

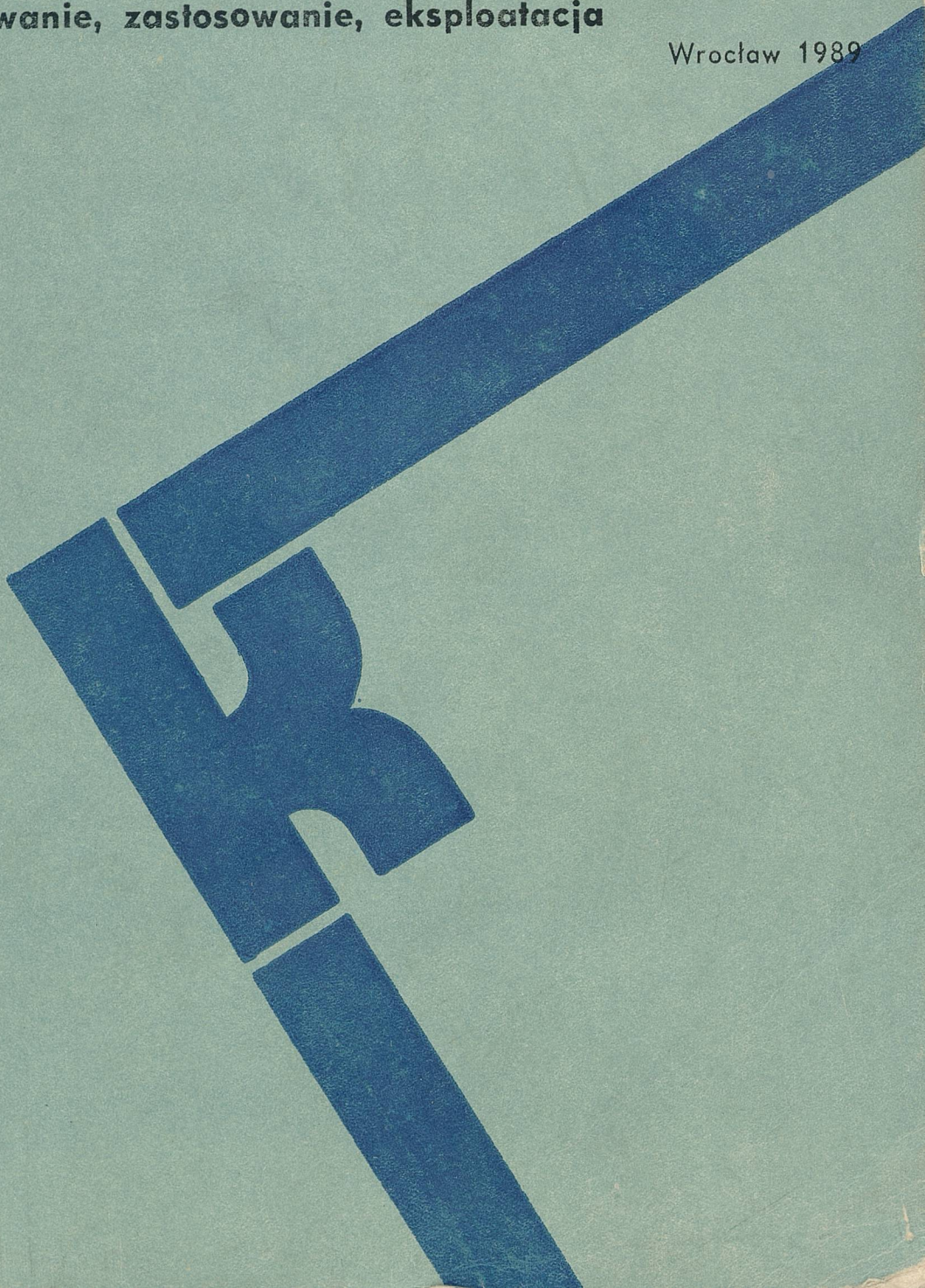
Prace Naukowe Centrum Obliczeniowego
Politechniki Wrocławskiej

Seria: Konferencje

Mag. 9
4

Sieci komputerowe '89
Projektowanie, zastosowanie, eksploatacja

Wrocław 1989



PRACE NAUKOWE POLITECHNIKI WROCŁAWSKIEJ

No. 9	Scientific Papers of the Computer Centre of the Technical University of Wrocław	No. 9
Conferences	No. 4	1989

Computer Networks '89
Design, applications, exploitation

Prace Naukowe Centrum Obliczeniowego
Politechniki Wrocławskiej

9

Seria:
Konferencje

4



Sieci komputerowe '89

Projektowanie, zastosowania, eksploatacja

Wrocław, 27–30 czerwca 1989



Wydawnictwo Politechniki Wrocławskiej · Wrocław 1989

Komitet programowy konferencji

Jerzy BATTEK (przewodniczący), Jacek BAŃKOWSKI, Tadeusz BATYCKI, Daniel BEM, Eugeniusz BILSKI, Jerzy BROMIRSKI, Wojciech CELLARY, Manfred FRANK, Zbigniew HUZAR, Andrzej KALIŚ, Wacław KASPRZAK, Piotr KOCIATKIEWICZ, Witold KOMOROWSKI (redaktor naukowy), Jerzy MARSZAŁEK, Dominik RUTKOWSKI, Wojciech SOBCZAK, Janusz SOCHACKI, Norbert WULST, Wojciech ZAMOJSKI, Andrzej ZIELIŃSKI

Recenzenci

Andrzej BABORSKI, Jacek BAŃKOWSKI, Jerzy BATTEK, Tadeusz BATYCKI, Daniel BEM, Eugeniusz BILSKI, Wojciech CELLARY, Iwona DUBIELEWICZ, Jacek GRUBER, Zbigniew HUZAR, Andrzej KALIŚ, Piotr KOCIATKIEWICZ, Witold KOMOROWSKI, Elżbieta KOSMULSKA-BOCHENEK, Ludwik KUŹNIARZ, Zygmunt MAZUR, Jacek MOŚCIŃSKI, Mieczysław MURASZKIEWICZ, Janusz RATAJCZAK, Dominik RUTKOWSKI, Henryk RYBIŃSKI, Janusz SOCHACKI, Maciej SYSŁO, Adam WOLISZ, Jan ZARZYCKI, Krzysztof ZIEMBA

Zawarte w tym zeszycie artykuły stanowią teksty referatów przyjętych na konferencję „Sieci komputerowe '89”.

Jest to już trzecia konferencja na temat sieci komputerowych organizowana przez Centrum Obliczeniowe Politechniki Wrocławskiej, tym razem pod hasłem *projektowanie – zastosowania – eksploatacja*. Ponieważ nasze poprzednie konferencje (w latach 1985 i 1987) spotkały się ze znacznym zainteresowaniem, w tym roku zaprosiliśmy więc do udziału również uczestników zagranicznych, czego rezultatem jest kilka referatów z Czechosłowacji, Holandii, Jugosławii, NRD, Wielkiej Brytanii i Związku Radzieckiego.

Referaty są publikowane w postaci nadesłanej przez Autorów; jedynie w kilku przypadkach wprowadzono drobne poprawki techniczne.

Selekcji nadesłanych materiałów dokonał Komitet programowy na podstawie opinii Recenzentów, którym w tym miejscu wyrażam podziękowanie.

Dziękuję zwłaszcza Autorom referatów przygotowanych na zamówienie Komitetu organizacyjnego, Panom: Eugeniuszowi Bilskiemu, Wojciechowi Cellaremu, Zbigniewowi Huzarowi, Piotrowi Kociatkiewiczowi, Nigelowi Linge, Krzysztofowi Odolakowi, Andrzejowi Pękałskiemu, Witoldowi Staniszkisowi, Zdzisławowi Spławskiemu i Józefowi Woźniakowi.

Witold Komorowski
(Redaktor naukowy)

© Copyright by Wydawnictwo Politechniki Wrocławskiej, Wrocław 1989

WYDAWNICTWO POLITECHNIKI WROCŁAWSKIEJ

Wybrzeże Wyspiańskiego 27, 50-370 Wrocław

ISSN 0860-1615

Nakład 300+90 egz. Ark. wyd. 18,75. Ark. druk. 16^{3/4}. Papier offset. kl. V, 70 g, B1.

Oddano do druku w maju 1989 r. Druk ukończono w czerwcu 1989 r.

Zakład Graficzny Politechniki Wrocławskiej. Zam. nr 757/89 – Cena zł 1000,—

Profesor
Jerzy Bromirski
1915-1989

14 marca 1989 r. zmarł nagle Profesor Jerzy Bromirski - pionier informatyki wrocławskiej, wychowawca wielu pokoleń studentów.

Swą pracę na Politechnice Wrocławskiej rozpoczął w 1948 r. jeszcze jako student Wydziału Elektromechanicznego. Te studia wrocławskie były kontynuacją rozpoczętych w 1938 r. w Warszawie a przerwanych przez udział w kampanii wrześniowej i pobyt w niemieckich oflagach. Po uzyskaniu stopnia inżyniera elektryka i magistra nauk technicznych został w 1950 r. asystentem w kierowanej przez Prof. Zygmunta Szparkowskiego Katedrze Teletechniki (później Automatyki i Telemechaniki).

W latach 1959-1961, już jako docent, oprócz pracy na Politechnice, zajmował stanowisko głównego konstruktora d.s. maszyn matematycznych w nowo powstałych Wrocławskich Zakładach Elektronicznych "Elwro". Był to okres budowy pierwszych komputerów Odra 1001, Odra 1002 i Odra 1003.

Dzięki staraniom Profesora w 1962 r. powołano specjalność "maszyny matematyczne" na Wydz. Elektroniki, rok później natomiast utworzono pierwszą w Polsce, oprócz Politechniki Warszawskiej, Katedrę Konstrukcji Maszyn Cyfrowych. Od 1968 r., kiedy powstał Instytut Cybernetyki Technicznej, Profesor kierował w nim Zakładem Automatów (później Systemów Cyfrowych), pełniąc jednocześnie funkcje dziekana Wydziału Elektroniki (1971-1978), następnie zaś Wydziału Informatyki i Zarządzania (do 1981).

W 1978 r. wraz ze swym zespołem przeniósł się do Centrum Obliczeniowego, gdzie utworzył grupę inżynierii oprogramowania i włączył ją w prace nad sieciami komputerowymi.

Po przejściu na emeryturę w 1985 r., mimo przebytej operacji oczu, nie zerwał kontaktów z Politechniką - był przewodniczącym pierwszej konferencji "Sieci komputerowe" w 1985 r. oraz członkiem komitetu programowego tych konferencji w 1987 r. i 1989 r.

W czasie swej wieloletniej pracy naukowej i dydaktycznej Profesor przygotował ok. 70 prac naukowych, kilka podręczników akademickich (wśród nich "Teoria automatów" była podstawową książką studentów informatyki na wszystkich polskich politechnikach), promował 33 doktorów, wychował wiele pokoleń studentów, przez których był szanowany i lubiany.

Profesor uczestniczył w pracy wielu stowarzyszeń, rad naukowych, pełnił różne funkcje konsultacyjne; m.in. był członkiem Państwowej Rady Informatyki, członkiem Zespołu Elektroniczacji i Automatyzacji Rady Głównej Nauki, Szkolnictwa Wyższego i Techniki, członkiem Centralnej Komisji Kwalifikacyjnej, zastępcą Redaktora naczelnego "Podstaw Sterowania" od chwili założenia pisma, członkiem-założycielem Polskiego Towarzystwa Informatycznego.

Jego otwartość na przedstawiane Mu problemy, życzliwość i wyrozumiałość w kontaktach z innymi zjednywały Mu powszechny szacunek i sympatię.

Odchodząc pozostawił wdzięczną pamięć o sobie i przeświadczenie, że zakończył się pewien okres polskiej informatyki - czas jej narodzin i dojrzewania.

information policy,
computer network

J. Van Arkel *

NETWORKS AND POPULATION INFORMATION SYSTEMS

The use of networks by government is increasing sharply. This is evident in various fields of application. This paper examines the use of networks from the point of view of developments in the field of personal data filing in the Netherlands. It also describes the network conception that is used to realise a decentralised structure for the national population registration system. This decentralised structure asks for standardisation of data elements as well as network protocols, enforced by strict regulations.

1. NEED FOR A POLICY

In recent years, the importance of personal data has increased owing to the growing number of government tasks. The government carries out its tasks for the benefit of individuals, groups of people and the population as a whole. In order to carry out these tasks satisfactorily, it has to have access to certain information on people. People's lives can be directly affected by a poorly-designed system. Important requirements for a coherent policy are:

- the wish to improve services by means of making optimal use of technical aids;
- attention for careful protection of privacy which includes the use of administrative numbers and the linking up of various data files;
- the need to use personal data more effectively and efficiently;
- the desire for more and improved information for policymaking;
- the tracking down and combating of fraud, for example social security and tax fraud;
- the desire to establish a cost allocation formula for certain flows of information;
- the need for good security.

*Ministry of Home Affairs, Netherlands

These are some reasons for having a good personal information systems policy. As yet such policy has not been enshrined in legislation.

2. ORGANISATIONAL CHART

Government information systems can be divided into areas of attention, such as education and public health, items to be registered, such as persons and property, and management systems, such as finance and employees. Population registration policy cares only about persons as objectives of registration.

In the Netherlands there is a minister responsible for each area and for each item to be registered. A basic data filing system is also being developed for the most important items to be registered. Basic data filing systems contain certain core facts on each registered item for general use within government. The Minister of Home Affairs is the politician responsible for coordinating government personal information systems.

His responsibility is directed primarily at the flows of information between the various sectors and the more or less comparable activities taking place in various sectors in order to prevent work being repeated unnecessarily. A Provisional Council has been set up in order to advise him in this respect.¹⁾

3. VARIOUS ASPECTS OF PERSONAL INFORMATION SYSTEMS POLICY

The various interests which are served by a good personal information system should at each occasion be weighed up carefully. These are:

- the task of the organisation or the part of the organisation concerned;
- the effectiveness and the efficiency of the information systems;
- the privacy of the individual.

Neither the task which a government body carries out nor the organisational form chosen for its implementation are part of personal information systems policy. The tasks of a government body are often laid down by law and should be considered as given for the purposes of personal information systems policy. Personal data may be necessary for the implementation of such tasks. What data, when and in what form are necessary is up to the organisation which is responsible for the implementation of the task in question. The protection of privacy is in the Netherlands a limiting condition for personal information systems policy. Late in 1988, an act²⁾ was passed in the Netherlands, which deals with the protection of privacy.

If we take the task and the privacy aspects more or less as facts of life then the emphasis in personal systems policy automatically comes to lie on effectiveness and efficiency of the way the government as a whole is handling its personal data.

In concrete terms this means for the lines of policy:

- not collecting and storing more data than is really necessary;
- collecting data only once (not collecting data which are already available elsewhere in government);
- storing data only once if possible;
- storing data as close to the individual as possible;
- having as few large central data bases as possible;
- making use of personal data that is already available in government.

The practical implementation of this decentralised approach means that more and more communication will have to take place between all the decentralized information systems. In other words: nation-wide networking.

4. BASIC POPULATION REGISTRATION SYSTEMS

By tradition, municipalities in the Netherlands play an important role in registering personal data. This is in general registration for social and judicial purposes. It takes the form of a card system and it is subject to stringent government regulations. At this moment a new electronic system is build for the above mentioned purpose. The main points of the new system are:

- updating of the regulations; the municipalities remain responsible for implementation;
- the municipalities may choose their own hardware and develop their own software; only the specifications are determined by the regulations;
- the government develops and maintains the communications network;
- this network carries out the standardized communication between all 700 municipalities and 500 data users;
- there is no question of interactive real time data exchange, but of an electronic mail system.

The choice of an electronic mail system is based on considerations of privacy, both individual and organisational. Organisations would object to other organisations being authorised to search their data bases and select information from them. Organisations object far less to answering queries from other organisations by using their own programs.

Conformity with international standards was sought in setting up the system which means that for the bottom three levels of the OSI model, the Dutch PTT's X.25 is used and for the higher levels, the X.400 standard.

5. FURTHER DEVELOPMENTS

We are working on further expansion of policy in a policy document. This document will be presented to the Lower House in 1989³. It contains recommendation in the following four areas: Data, Functioning, Technical aspects, Management and organisation.

5.1 Data

Particular attention will be given to the definition of personal data and the format in which they are recorded for the purposes of data exchange. Standardised regulations are to be drawn up for personal data which are needed frequently.

5.2 Functioning

Requirements regarding the functioning of the various systems can be laid down only by the organisations in question. General principles, however, such as collecting and storing data only once, should be followed. Another general principle is that policy information should not be collected separately but, where possible, should be compiled on the basis of data already in the basic data filing system or in executive files.

5.3 Technical aspects

Government departments are free in their choice of hardware and software, as they are with the basic data filing systems. The technical realisation of functional requirements is a matter for the organisations in the various sectors. If however communication is involved, then general regulations have to be adhered to. As for the network it has already been decided that the PTT's X.25 data net is to be used for government networks.⁴

The next regulation should be that the X.400 standard for personal data exchange should be adhered to unless this is not practical. Because an organisation, like for example the police, needs a very quick answer to certain questions.

The plan that in the future, data exchange between the various basic data filing systems will be possible through a standardised infrastructure (an interdepartmental/intergovernmental network) is also playing a part. There will therefore be a functional and physical network for the government as a whole on personal data, property data and social objects for example.

5.4 Management and organisation

The problem of implementation is not the only factor determining the choice of organisational form. The main issues are centralisation and decentralisation, i.e. the balance of power in public administration.

In addition, the cost aspect is important. In view of the amounts involved in large systems and networks, every party with an interest in an information system should be able to discuss and make decisions on its operation. At present, an organisational form for the basic data filing system is being sought in which all interested parties are represented in a balanced way and which establishes a cost allocation formula. This formula will be laid down in legislation.

6. CONCLUSION

Developments in the Netherlands in the field of personal information systems are highly decentralised. Recently, municipalities in particular have been strong supporters of this, the argument being that anyone who has more or less exclusive access to information also wields power. Parliament too has opted for decentralised small-scale personal data files in order to protect privacy.

As central government needs personal data and in view of the progress in automation, the use of networks for the exchange of personal data will increase considerably. The need for standardization in order to facilitate data exchange and the need for joint operation in order to cut costs mean that cooperation becomes more and more essential.

Experience has shown that cooperation does not always take place if it is voluntary. Mutual interest is not enough! The allocation of responsibilities is too fragmented and there is too little coordination. Central control in the form of strict regulations in this respect is therefore unavoidable.

References:

- 1) Royal Decree of Institution, 24/10/1985
- 2) Wet Persoonsregistraties (Dutch Privacy Act), 28/12/1988
- 3) Draft document "Care for data", 01/03/1989
- 4) Datanet 1 regulation, Stat 1982 nr. 82, Decree Information in Central Government

weryfikacja protokołów,
logika niezmienników,
CSF

Janusz BARCZYŃSKI[✉]

MODULARNA WERYFIKACJA PROTOKOŁÓW^{✉✉}

W poprzedniej pracy [1] przedstawiono aksjomatyczne podejście do formalnego opisu i weryfikacji protokołów. Również scharakteryzowano technikę szczegółową zgodną z tym podejściem. Język opisu protokołów jest uogólnieniem notacji CSP. Weryfikacja zaś, dotyczy dowodzenia własności niezmienniczych protokołów. Ten artykuł zawiera szerszą charakterystykę techniki. Dla ilustracji podano formalny opis protokołu z literatury i przeprowadzono jego weryfikację.

1. WPROWADZENIE

Oprogramowanie komunikacyjne sieci komputerowej, w tym implementacja protokołów komunikacyjnych, tworzy rozproszony przestrzennie system programów równoległych (współbieżnych), którego składowe komunikują się przez wymianę komunikatów. Gwarancję poprawności oprogramowania sieciowego, na akceptowalnym poziomie ufności, może dać jedynie konsekwentne stosowanie odpowiedniej metodologii konstruowania, której integralnym fragmentem jest część dotycząca formalnego opisu oraz weryfikacji [2]. Metody testowe, jak wiadomo, nie mogą zagwarantować poprawności. Wśród formalizmów, wykorzystywanych do opisu i analizy systemów ze współbieżnością, istotne znaczenie mają dwa, a mianowicie notacja CSP Hoare'a [6] oraz CCS Milnera [7]. Warianty obu notacji znalazły zastosowanie w dziedzinie protokołów komunikacyjnych. Propozycja Hoare'a dotycząca CSP, doprowadziła do powstania wielu odmian i uogólnień tej notacji. Między innymi powstała wersja teoretyczna zbliżona do CCS. Opracowano różne semantyki CSI oraz systemy dowodowe dla weryfikacji CSP-programów. Drugi nurt zaowocował pewną liczbą języków współbieżnych opartych o CSP, między innymi opracowano takie implementacje jak [8]: transputerowy OCCAM a także CSP-S, CSP/8C, CSP-i oraz inne. W [3] przedstawiono technikę formalnego opisu i weryfikacji protokołów komunikacyjnych sieci komputerowej o architekturze

✉/ Centrum Obliczeniowe Politechniki Wrocławskiej, Wrocław

✉✉/ Praca zrealizowana w ramach CPER 8.13, cel nr 52

zgodnej z modelem OSI. Język opisu protokołów stanowi tu uogólnienie notacji CSP. Krótką charakterystykę techniki podano w [1] oraz [4]. Natomiast w [5] podano szerszy kontekst teoretyczny dla techniki a także zaproponowano nowe metody weryfikacji. Mówiąc krótko, protokoły komunikacyjne zapisywane są jako uogólnione CSP-programy, natomiast weryfikacja polega na dowodzeniu własności niezmienniczych takich programów-protokołów. Inne własności mogą być wyrażane przez język logiki temporalnej. System wnioskowania jest kompozycyjny, w sensie składania dowodów. Ten artykuł zawiera szerszą, w stosunku do [1], charakterystykę techniki. Dla ilustracji podano fragment formalnego opisu protokołu wymiany danych, z [4], i przeprowadzono jego weryfikację.

2. O TECHNICĘ

Protokoły komunikacyjne opisywane są jako programy równoległe o następującej ogólnej postaci $Q \equiv [Q_1 // \dots // Q_n]$, gdzie Q_i jest procesem sekwencyjnym $A::P$ lub programem równoległym. Instrukcje komunikacyjne mają postać:

- $\langle A!T(\bar{e});S \rangle$, $\langle B?T(\bar{x});R \rangle$, (dla komunikatów)
- $\langle A!T;\text{skip} \rangle$, $\langle B?T;\text{skip} \rangle$. (dla sygnałów)

Programy S oraz R mogą mieć postać $S \equiv h.out := h.out \circ [\bar{e}]$ oraz $R \equiv h.in := h.in \circ [\bar{x}]$, gdzie $h.out$ oraz $h.in$ są zmiennymi historii komunikatów odpowiednio wyjściowej oraz wejściowej; $\circ$ jest operatorem konkatenacji. Formuły poprawnościowe, wyrażające twierdzenia o własnościach niezmienniczych programów (protokołów), mają postać $\langle Q, \varphi \rangle \text{ inv } I$, gdzie φ oraz I są formułami pierwszego rzędu; I jest formułą-niezmiennikiem a φ jest warunkiem początkowym programu Q . Przy konstruowaniu formuł-niezmienników wykorzystywane są klasyczne spójniki logiczne oraz następujące operatory na historiach komunikatów: $|h|$ - długość historii h ; $h \circ h'$ - konkatenacja historii h i h' ; $h(i)$ - i -ty element historii h , $1 \leq i \leq |h|$; $m \in h$ - " m jest elementem historii h "; $h = h'$ - "historia h jest równa h' "; $h \leq h'$ - " h jest początkową podhistorią historii h' " itp.

System dowodowy umożliwia dowodzenie formuł poprawnościowych. Zawiera, między innymi, reguły komponowania dowodów, które pozwalają na modularną budowę dowodów. Mówiąc dokładniej, dowód własności niezmienniczych programu równoległego uzyskiwany jest z odpowiedniego złożenia dowodów dla programów-składowych tego programu. Dowody dla składowych sekwencyjnych uzyskiwane są przy tym w oparciu o teksty odpowiednich procesów. System dowodowy podano w [3], natomiast niżej przedstawimy szkic dowodu w tym systemie.

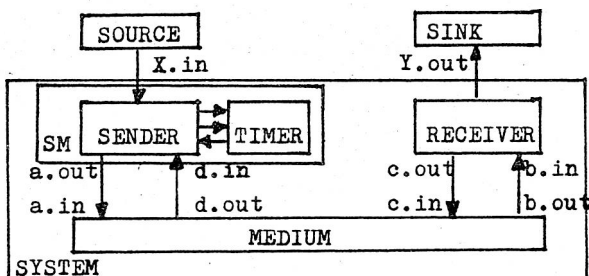
3. PRZYKŁAD

Przykład pozwala zapoznać się z niektórymi konstrukcjami języka opisu protokołów, z postacią formuł-niezmienników oraz ze sposobem ich dowodzenia. Rozważany będzie następujący program równoległy

$$DTP \equiv [SOURCE::SO//SYSTEM//SINK::SI],$$

gdzie: $SYSTEM \equiv [SM//MEDIUM//RECEIVER::R]$ jest modelem systemu komunikacyjnego umożliwiającego jednokierunkowy transfer danych pomiędzy użytkownikami (procesy SOURCE oraz SINK są modelami użytkowników); programy $SM \equiv [SENDER::S//TIMER::T]$ oraz RECEIVER są modelami modułów protokołu - formalny opis protokołu polega na zdefiniowaniu tych programów; program MEDIUM jest modelem medium komunikacyjnego. Rys. 1 przedstawia graf komunikacyjny programu DTP, z zaznaczonymi zmiennymi historii. Proces SENDER zdefiniowano w Dodatku, natomiast procesy TIMER oraz RECEIVER znajdują się w [4] - pominięto je aby nie zwiększać objętości artykułu.

Rys. 1



Formuły poprawnościowe dla MEDIUM oraz SYSTEM mogą mieć postać:

$$(a) \quad \langle MEDIUM, \varphi_{MEDIUM} \rangle \text{ inv Ext}_{MEDIUM}$$

gdzie $Ext_{MEDIUM} \equiv (\forall m)(m \in b.out \rightarrow m \in a.in) \wedge (\forall m')(m' \in d.out \rightarrow m' \in c.in)$. Medium komunikacyjne jest zawodne: może gubić, zmieniać kolejność oraz generować duplikaty przesyłanych komunikatów.

$$(b) \quad \langle SYSTEM, \varphi_{SYSTEM} \rangle \text{ inv } Y.out \leq X.in,$$

gdzie $\varphi_{SYSTEM} \equiv \omega_{SENDER} \wedge \omega_{TIMER} \wedge \varphi_{MEDIUM} \wedge \omega_{RECEIVER}$.

System komunikacyjny jest niezawodny: protokół wymiany danych zabezpiecza przed błędami, które mogą być wprowadzane przez medium.

Weryfikacja protokołu polega na udowodnieniu formuły (b) przy założeniu, że zachodzi (a). Oto szkic dowodu w systemie z [3]. Wprowadzimy następujące oznaczenia:

$$mseq(h) = \begin{cases} 0 & \text{dla } h = \emptyset \text{ (historia pusta)} \\ \max\{\{j: (\exists d) [j, d] \in h\}\} & \text{dla } h \neq \emptyset, \end{cases}$$

gdzie $h \in \{a.out, a.in, b.out, b.in\}$,

- $Ext_{SENDER} \equiv mseq(a.out) \leq |X.in| \wedge$
 $(\forall m) \{m \in a.out \rightarrow (\exists i) (1 \leq i \leq |X.in| \wedge m = [i, X.in(i)])\},$
- $Ext_{RECEIVER} \equiv |Y.out| \leq mseq(b.in) \wedge$
 $(\forall k) \{1 \leq k \leq |Y.out| \rightarrow [k, Y.out(k)] \in b.in\}.$

Początkowo, biorąc teksty odpowiednich procesów dowodzimy formuły:

- (1) $\langle SENDER, \omega_{SENDER} \rangle \text{ inv } Ext_{SENDER},$
- (2) $\langle TIMER, \omega_{TIMER} \rangle \text{ inv true}.$

Ze złożenia dowodów (1) i (2) uzyskiwany jest dowód formuły:

- (3) $\langle SM, \varphi_{SM} \rangle \text{ inv } Ext_{SENDER},$ gdzie $\varphi_{SM} \equiv \omega_{SENDER} \wedge \omega_{TIMER}.$

Biorąc tekst procesu RECEIVER, dowodzimy formułę:

- (4) $\langle RECEIVER, \omega_{RECEIVER} \rangle \text{ inv } Ext_{RECEIVER}.$

Ze złożenia dowodów formuł (3), (4), (a) uzyskiwany jest dowód formuły:

- (5) $\langle SYSTEM, \varphi_{SYSTEM} \rangle \text{ inv } r_{SYSTEM},$

gdzie $r_{SYSTEM} \equiv Ext_{SENDER} \wedge Ext_{MEDIUM} \wedge Ext_{RECEIVER}.$

Ponieważ prawdziwa jest formuła

- (6) $(r_{SYSTEM} \wedge GI) \rightarrow Y.out \leq X.in,$

gdzie $GI \equiv a.out = a.in \wedge b.out = b.in \wedge c.out = c.in \wedge d.out = d.in$ jest globalnym niezmiennikiem, więc z dowodu (5) uzyskiwany jest dowód formuły (b).

BIBLIOGRAFIA

- [1] Barczyński J.: Opis i weryfikacja protokołów - podejście aksjomatyczne. W: Sieci komputerowe. Teoria, technika, zastosowania. Prace Naukowe Centr. Oblicz. Polit. Wrocł. nr 5, seria: Konf. nr 3, ss. 174-181, Wrocław 1987.
- [2] Barczyński J., Huzar Z., Spławski Z.: Wstępna analiza problemów specyfikacji i weryfikacji oprogramowania sieciowego. Polit. Wrocł., Centr. Oblicz., raport spr 7/87.
- [3] Barczyński J.: Formalny opis oraz weryfikacja protokołów komunikacyjnych sieci komputerowej. Polit. Wrocł., Centr. Oblicz., raport pre 6/87.
- [4] Barczyński J.: Proving Invariance Properties of Protocols. Polit. Wrocł., Centr. Oblicz., raport pre 7/87 - przyjęto do publikacji.
- [5] Barczyński J.: Weryfikacja programów równoległych. Uogólnione logiki Hoare'a. Polit. Wrocł., Centr. Oblicz., raport spr 2/88.
- [6] Hoare C.A.R.: Communicating Sequential Processes. CACM, vol. 21, no. 8, 1978.
- [7] Milner R.: A Calculus of Communicating Systems. LNCS 92, 1980.

- [8] Wrench K.I.: CSP-i: An Implementation of CSP. Software-Practice and Experience, vol. 18/6/, ss. 545-560, 1988.

MODULAR VERIFICATION OF PROTOCOLS

An axiomatic approach to a formal description and verification of computer network protocols is presented in [1]. In the same paper a detailed technique consistent with this approach is also characterized. The protocol description language is a generalization of CSP notation, while the verification consists in proving invariance properties of protocols. A more extensive characteristic of this technique is presented in this paper. As an illustration a formal description of protocol from the literature is given and its verification made.

МОДУЛЬНАЯ ПРОВЕРКА КОРРЕКТНОСТИ ПРОТОКОЛОВ

Представлена техника формального описания и проверки корректности протоколов компьютерной сети. Язык описания протоколов является обобщением нотации CSP. Проверка корректности заключается в доказании инвариантных свойств протоколов. Для иллюстрации дано формальное описание протокола, помещенного в предметной литературе и проведена его проверка корректности.

DODATEK. Definicja procesu SENDER z warunkiem początkowym

```

SENDER:: type data = ...; sequence_number = 0..infinity;
      packet = [sequence_number, data]; ack = [sequence_number];
var S_data: data; highest_sent, ackno, waiting_for_ack: sequence_number;
      X.in: history of data; a.out: history of packet; d.in: history of ack;
*[highest_sent=waiting_for_ack-1; skip → {przesłanie pakietu}
  highest_sent:=highest_sent+1;
  <SOURCE?data(S_data); X.in:=X.in o [S_data]>;
  <MEDIUM!packet(highest_sent, S_data); a.out:=a.out o [highest_sent,
                                                             S_data]>;
  <TIMER!start; skip>
  [true; <MEDIUM?ack(ackno); d.in:=d.in o [ackno]> → {obsługa potwierdzenia}
    [ackno ≠ waiting_for_ack; skip → skip [
      ackno = waiting_for_ack; skip → <TIMER!cancel: skip>;
      waiting_for_ack:=waiting_for_ack+1]
    [true; <TIMER?timeout; skip> → {obsługa timeout'u}
      <MEDIUM!packet(highest_sent, S_data);
        a.out:=a.out o [highest_sent, S_data]>;
      <TIMER!start; skip>
    ]
  ]

```

Warunek początkowy :

$\omega_{SENDER} = \text{highest_sent} = 0 \wedge \text{waiting_for_ack} = 1 \wedge X.in = \emptyset \wedge a.out = \emptyset \wedge d.in = \emptyset$

weryfikacja protokołów,
Estelle,
Alternating Bit

Janusz BARCZYŃSKI[✉]

ANALIZA SEMANTYCZNA SPECYFIKACJI EXAMPLE^{✉✉}

Example jest specyfikacją protokołu Alternating Bit, występującą w dokumentach ISO dotyczących techniki ESTELLE. W artykule pokazano, że protokół ten może osiągnąć tzw. żywy uścisk, którego efektem zewnętrznym jest brak postępu w świadczeniu usług. Dokonano poprawy specyfikacji.

1. WPROWADZENIE

Język ESTELLE i jego semantyka są przedmiotem prac standaryzacyjnych w ISO [1,2]. Istnienie formalnej semantyki jest warunkiem koniecznym każdej techniki weryfikacji. Jednym z potencjalnych kandydatów do weryfikacji jest protokół Alternating Bit (AB) ze specyfikacji Example. Specyfikacja ta przytaczana jest w dokumentach ISO prezentujących technikę ESTELLE [1] (1st DP 9074, 2nd DP, DIS). Analiza semantyczna specyfikacji Example wskazuje, że protokół AB nie jest poprawny. Mianowicie, może łatwo wystąpić tzw. żywy uścisk (livelock). Ogólnie, żywy uścisk występuje w systemie komunikacyjnym jeżeli moduły protokołu realizują cykl operacji C taki, że:

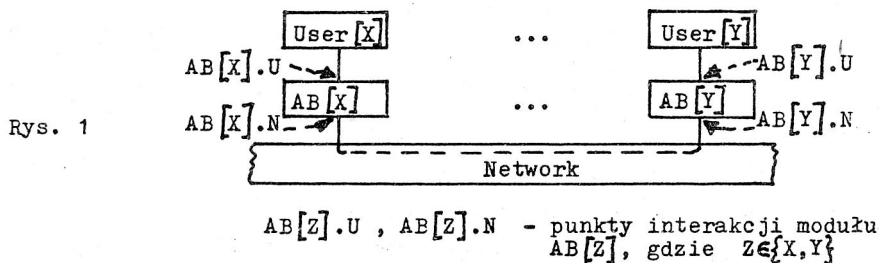
- (i) z cyklu C nie ma wyjścia, chociaż istnieją przejścia z każdego stanu cyklu C,
- (ii) realizacja cyklu C nie powoduje wykonywania "użytecznej pracy".

Z punktu widzenia użytkowników systemu komunikacyjnego, warunek (ii) oznacza, że nie ma postępu w świadczeniu usług protokołu. Podobny efekt zewnętrzny obserwuje się w przypadku osiągnięcia przez system komunikacyjny stanu blokady (deadlock - śmiertelny uścisk), tzn. stanu, który nie jest stanem końcowym i z którego nie ma dalszych przejść.

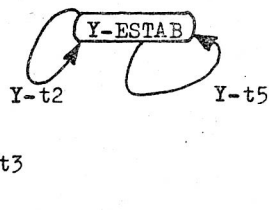
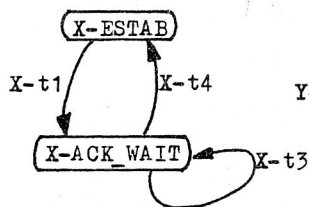
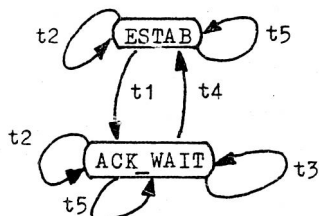
2. ANALIZA SPECYFIKACJI

✉/ Centrum Obliczeniowe Politechniki Wrocławskiej, Wrocław
✉✉/ Praca zrealizowana w ramach CPBR 8.13, cel nr 52

W Example występuje definicja modułu protokołu AB oraz niekompletne definicje modułu użytkownika (User) i modułu sieci (Network) - podano tylko nagłówki tych ostatnich modułów (patrz np. [1] - DIS 9074). Na rys. 1 przedstawiono fragment systemu egzemplarzy modułów generowanego przez Example oraz strukturę połączeń ich punktów interakcji.



W Dodatku podano część deklarującą przejścia definicji ciała modułu protokołu AB. Ta ostatnia pozwala nam przedstawić strukturę sterowania modułu AB jak na rys. 2. W dalszym ciągu przyjmiemy, bez utraty ogólności rozważań, że wymiana danych jest jednokierunkowa - od użytkownika User[X] do User[Y]. Wtedy aktualne struktury sterowania egzemplarzy AB[X] oraz AB[Y] mają postać podaną odpowiednio na rys. 3 oraz rys. 4.



Z punktami interakcji $AB[Z].N$, gdzie $Z \in \{X, Y\}$ (por. rys. 1), skojarzone są nieograniczone kolejki FIFO, które oznaczmy przez $Q(AB[Z].N)$. Na koniec kolejki $Q(AB[Y].N)$ dołączane są komunikaty sieciowe z danymi nadawane, via sieć, przez egzemplarz $AB[X]$ - przez wykonywanie przejść $X-t1$ i $X-t3$, w których występuje instrukcja output N.DATA_request(B) (por. Dodatek). Natomiast na koniec kolejki $Q(AB[X].N)$ dołączane są komunikaty sieciowe z potwierdzeniami nadawane, via sieć, przez egzemplarz $AB[Y]$. Komunikaty te mogą być zdjęte z początków kolejek $Q(AB[Y].N)$ oraz $Q(AB[X].N)$ przez wykonanie przejść odpowiednio $Y-t5$, egzemplarza $AB[Y]$, oraz $X-t4$, egzemplarza $AB[X]$. W przejściach tych występują klauzule when N.DATA_response(Ndata) (por. Dodatek). Komunikaty sieciowe są wartościami zmiennych rekordowych B oraz Ndata, zadeklarowanych niżej.

```

var B, Ndata : record
  Id : (DATA, ACK); { rodzaj komunikatu: DATA - komunikat z danymi,
                     ACK - komunikat z potwierdzeniem }
  Conn: Cep_type;   { identyfikator nadawcy - w naszym przypadku X
                     lub Y }
  Data: U_Data_type; { dane użytkownika; w komunikatach z potwierdze-
                     niami pole to jest puste }
  Seq : 0..1       end { liczba sekwencyjna }

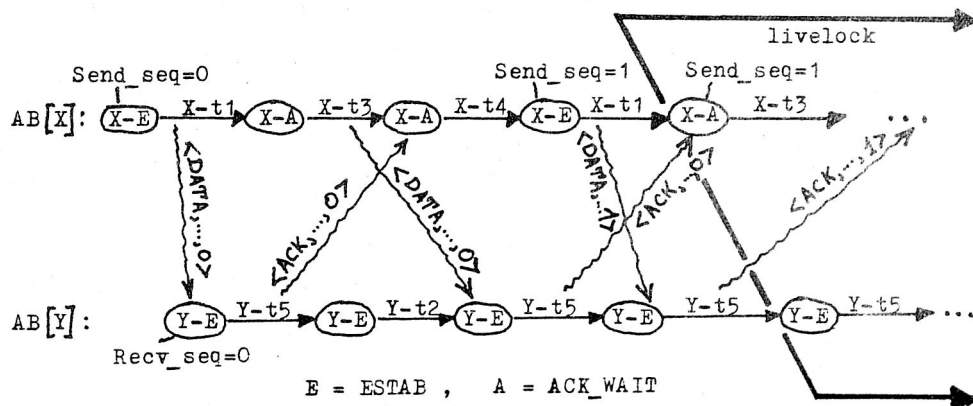
```

W dalszym ciągu będą nas interesowały pola Id oraz Seq, stąd komunikaty sieciowe będą oznaczane następująco :

- $\langle \text{DATA}, \dots, \text{ls} \rangle$ - komunikat z danymi,
- $\langle \text{ACK}, \dots, \text{ls} \rangle$ - komunikat z potwierdzeniem, gdzie $\text{ls} \in \{0, 1\}$.

Założenie 1. Założmy, że $\text{AB}[X]$ znajduje się w stanie sterowania X-ACK_WAIT oraz, że na wyjściu kolejki $Q(\text{AB}[X].N)$ znajduje się komunikat $\langle \text{ACK}, \dots, \text{ls} \rangle$ taki, że $\text{ls} \neq \text{Send_seq}$. ■

Przy takim założeniu przejście X-t4 nie będzie nigdy możliwe, ponieważ zachodzi $(\text{Ndata.Id} = \text{ACK}) \text{ and } (\text{Ndata.Seq} \neq \text{Send_seq})$ (oczywiście $\text{ls} = \text{Ndata.Seq}$) i klauzula provided nie jest spełniona. Tylko wykonanie X-t4 może zdjąć komunikat $\langle \text{ACK}, \dots, \text{ls} \rangle$ z wyjścia kolejki $Q(\text{AB}[X].N)$. Stąd dalej $\text{AB}[X]$ może wykonywać jedynie przejście spontaniczne X-t3 powodujące retransmisję komunikatu z danymi. W odpowiedzi na takie zachowanie $\text{AB}[X]$, egzemplarz $\text{AB}[Y]$ może jedynie wykonywać przejście Y-t5 tzn. odbierać retransmitowane komunikaty, nie przekazując ich do bufora odbiorczego, i nadawać potwierdzenia. Z powyższego widać, że gdy $\text{AB}[X]$ znajdzie się w stanie spełniającym założenie (1) to wówczas zostanie przerwane przekazywanie danych od $\text{User}[X]$ do $\text{User}[Y]$. Na zakończenie wystarczy pokazać, że stan wymieniony w zał. (1) może być łatwo osiągnięty. To ostatnie można zobaczyć analizując scenariusz komunikacji z rys. 5. Po wykonaniu, pokazanego tam, drugiego przejścia X-t1 zostaje osiągnięty stan z założenia (1).



Rys. 5

3. POPRAWA SPECYFIKACJI

Aby wyeliminować, omawiany wyżej, błąd specyfikacji Example wystarczy następująco zmodyfikować przejście t4 (por. Dodatek):

```

from ACK WAIT to ESTAB                                { przejście t4' }
when N.DATA_response(Ndata)
provided Ndata.Id = ACK
begin if Ndata.Seq = Send_seq then
    begin Remove(Send_buffer);
        Send_seq := (Send_seq + 1) mod 2 end end

```

BIBLIOGRAFIA

- [1] ESTELLE - A Formal Description Technique Based on an Extended State Transition Model. ISC, 1st DP 9074 (1985), 2nd DP (1986), DIS (1987).
- [2] Barczyński J.: Technika formalnego opisu ESTELLE - wprowadzenie do języka i semantyki. Polit. Wrocław. Centr. Oblicz., raport spr 13/87.

DODATEK. Deklaracja przejść modułu protokołu AB

```

from ESTAB to ACK WAIT
when U.SEND_request(Udata)
begin
    copy(P.Msgdata,Udata);
    P.Msgseq:=Send_seq;
    Store(Send_buffer, P);
    Format_data(P, B);
    output N.DATA_request(B)
end;

```

{t1}

```

from ACK WAIT to ESTAB
when N.DATA_response(Ndata)
provided (Ndata.Id=ACK) and
        (Ndata.Seq=Send_seq)
begin
    Remove(Send_buffer);
    Send_seq:=(Send_seq + 1) mod 2
end;

```

{t4}

```

from EITHER to same
when U.RECEIVE_request
provided not buffer_empty(Recv_buffer)
begin
    Q:=Retrieve(Recv_buffer);
    output U.RECEIVE_response(Q.Msgdata);
    Remove(Recv_buffer)
end;

```

{t2}

```

from EITHER to same
when N.DATA_response(Ndata)
provided Ndata.Id = DATA
begin
    copy(Q.Msgdata,Ndata.Data);
    Q.Msgseq:=Ndata.Seq;
    Format_ack(Q,B);
    output N.DATA_request(B);
    if Ndata.Seq=Recv_seq then
        begin
            Store(Recv_buffer, Q);
            Recv_seq:=(Recv_seq + 1) mod 2
        end ;
    end ;

```

{t5}

```

from ACK_WAIT to ACK_WAIT
delay(Retran_time)
begin
    P:=Retrieve(Send_buffer);
    Format_data(P,B);
    output N.DATA_request(B)
end;

```

{t3}

SEMANTICAL ANALYSIS OF EXAMPLE SPECIFICATION

An Example is a specification of the Alternating Bit protocol which can be found in the ISO documents concerning the ESTELLE technique. The protocol mentioned can reach the so-called livelock. Lack of progress in the service providing is an external effect of the livelock. In the work the improvement of specification is made.

СЕМАНТИЧЕСКИЙ АНАЛИЗ СПЕЦИФИКАЦИИ EXAMPLE

Example является спецификацией протокола Alternating Bit, выступающей в документах ISO, касающихся техники ESTELLE. В статье было показано, что этот протокол может достигнуть т.наз. живого заклинивания (livelock), внешним эффектом которого является отличие прогресса в оказании услуг. Сделано исправление спецификации.

sieć komputerowa,
architektura,
standard

Eugeniusz BILSKI*

ARCHITEKTURA WYŻSZYCH WARSTW MODELU ODNIESIENIA OSI/ISO

W referacie przedstawiono aktualny stan prac ISO nad architekturą warstw sesji, prezentacji i aplikacji, modelu OSI. W referacie powołane są najważniejsze dokumenty ISO, dotyczące modelu odniesienia jako całości oraz dotyczące ww. warstw, które ukazały się w okresie od 1985 r. do połowy 1988 r.

1. Wprowadzenie

Rosnące ciągle potrzeby w zakresie komunikacji międzynarodowej oraz rosnące możliwości ich realizacji, oparte na osiągnięciach mikroelektroniki, powodują coraz szybszy rozwój sieci komputerowych. Funkcjonowanie sieci w skali międzynarodowej jest możliwe tylko wtedy, gdy w sieciach krajowych będą obowiązywać te same standardy międzynarodowe. Bliska już wizja powstania Zjednoczonej Europy, spowodowała gwałtowny rozwój prac w zakresie standardów sieci komputerowych. Jako bazę dla tych prac przyjęto model odniesienia OSI/ISO. Do modelu odniesienia skłania się również coraz więcej firm amerykańskich i japońskich. Standaryzacja czterech dolnych warstw modelu jest już daleko zaawansowana i to zarówno dla sieci lokalnych jak i rozległych [1]. Ukazały się już na rynku układy scalone realizujące protokoły dolnych warstw. Natomiast dla wyższych warstw, a zwłaszcza dla warstwy aplikacji, opracowano standardy dla wybranych zastosowań. Istotnym hamulcem prac był brak rozwiniętego modelu funkcjonowania wyższych warstw. Model taki jest opracowywany w ISO od kilku lat; obecnie osiągnął stan, który pozwala wnioskować, że w dalszym jego rozwoju nie powinno już być zmian rewolucyjnych.

Niniejszy referat ma na celu przybliżenie, krajowym użytkownikom sieci komputerowych, podstawowych elementów tego modelu. Sprawa jest ważna dlatego, że dotyczy wszystkich obecnych i potencjalnych użytkowników sieci komputerowych. Włączenie do sieci procesów aplikacyjnych użytkownika, wymaga dobudowania do nich modułów sieciowych, specyficznych dla każdego typu procesu aplikacyjnego.

* Instytut Cybernetyki Technicznej Politechniki Wrocławskiej

2. Opis ogólny górnych warstw modelu OSI

W sieci komputerowej współdziałają ze sobą procesy aplikacyjne, rezydujące w komputerowych systemach rzeczywistych. Przykładami procesów aplikacyjnych są: programy użytkowe, bazy danych, zbiory danych, kompilatory języków programowania, terminale, sterowniki procesów technologicznych, roboty. Współdziałanie procesów aplikacyjnych odbywa się w celu wykonania zadania (sieciowego) i polega na komunikacji pomiędzy nimi, według uzgodnionych:

- reguł zarządzania transferem danych;
- semantyk związanych z przekazywanymi danymi;
- abstrakcyjnej syntaktyki użytej do transferu danych.

W środowisku OSI komunikacja taka odbywa się za pomocą jednostek aplikacyjnych (AE - Application Entity), rys. 1, stanowiących część modelu OSI. Rzeczywisty system otwarty jest to system, którego proces aplikacyjny jest wyposażony w jednostkę aplikacyjną. Jeden rzeczywisty system otwarty może zawierać jeden lub więcej procesów aplikacyjnych. Wykonanie zadania wymaga komunikacji pomiędzy dwoma lub więcej procesami aplikacyjnymi. Jeden proces aplikacyjny może zawierać jedną lub więcej jednostek aplikacyjnych. Jednostka aplikacyjna obsługuje tylko jeden proces aplikacyjny. Komunikacja pomiędzy jednostkami aplikacji odbywa się za pośrednictwem jednostek funkcjonalnych w dolnych warstwach modelu, są to odpowiednio: jednostka prezentacji (PE - Presentation Entity), jednostka sesji (SE - Session Entity) i następnie, nie pokazanych już na rys. 1, jednostek transportowej, sieciowej i liniowej. Zgodnie z ogólną konwencją usług, obowiązującą w modelu OSI (opisaną w dokumencie ISO TR 8509), warstwy sąsiednio niższe świadczą usługi warstwie sąsiednio wyższej. Usługi te dostępne są w punktach dostępu do usług (SAP - Service Access Point); na rys. 1 pokazane są punkty SAP dla warstwy prezentacji (PSAP), sesji (SSAP), transportowej (TSAP).

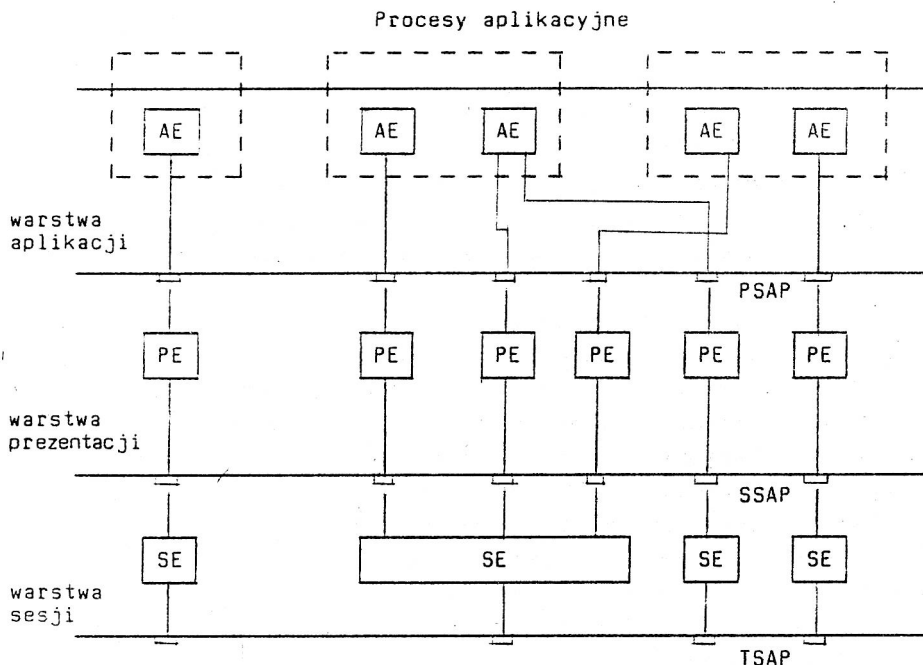
3. Charakterystyki wyższych warstw modelu OSI

3.1. Warstwa sesji

Poniżej przedstawione zostaną podstawowe własności użytkowe warstwy sesji, opracowane w oparciu o [3] i [4].

Warstwa sesji świadczy usługi warstwie prezentacji. Usługi te są realizowane w oparciu o usługi warstwy transportowej oraz funkcje realizowane w warstwie sesji. W środowisku OSI, zorientowanym połączeniowo, usługi warstwy sesji są następujące:

1. Ustanawianie połączenia sesji z jednoczesną negocjacją parametrów połączenia.



AE - jednostka aplikacji
PE - jednostka prezentacji

SE - jednostka sesji
SAP - punkt dostępu do usług

Rys. 1. Ogólne zasady współdziałania procesów aplikacyjnych w środowisku OSI

Fig. 1. Interconnection of application processes in OSI environment

2. Rozłączenie połączenia sesji, które może być uporządkowane - po przeniesieniu wszystkich danych lub nieuporządkowane (nagłe) - z możliwością utraty danych.
3. Transfer danych normalnych; pozwalający na wymianę danych, połączoną zwykle ze sterowaniem przepływem i dialogu.
4. Przesyłanie danych przyspieszonych; nie stosuje się wtedy sterowania przepływem i dialogiem; dane te mogą wyprzedzać dane normalne.
5. Zarządzanie tokenem, pozwalające sterować prawem wyłączności realizacji pewnych funkcji.
6. Sterowanie dialogiem, pozwalające na ustawienie pracy półduplexowej (z użyciem tokenu) lub duplexowej.
7. Synchronizację, pozwalającą (z użyciem tokenu) na umieszczanie w danych użytkownika, znaków synchronizujących, stanowiących potwierdzone punkty identyfikacyjne. Główne punkty synchronizacyjne dzielą dialog na tzw. jednostki dialogu (rys. 2). Brak potwierdzenia wysłanego znaku, umożliwia cofnięcie dialogu do ostatniej prawidłowo zrealizowanej jednostki dialogu.

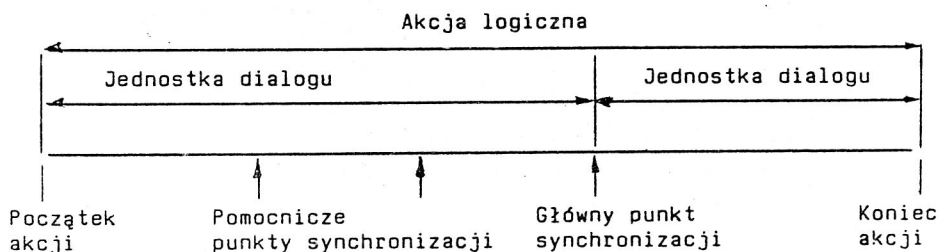
8. Resynchronizację, pozwalającą użytkownikom na zmianę synchronizacji i cofnięcie dialogu do stanu istniejącego w chwili wskazanej przez punkt resynchronizacji.
9. Sterowanie akcjami, pozwalające użytkownikom podział dialogu na tzw. akcje logiczne; sterowanie akcjami umożliwia przerwanie akcji i ich wznowienie w późniejszym czasie. Realizacja tej funkcji wymaga posiadania tokenu.
10. Informowanie o stanach wyjątkowych, pozwala warstwie sesji informować o nieoczekiwanych sytuacjach, nie wykrywanych przez inne usługi, np. błąd protokołu.
11. Niezależna wymiana danych, umożliwia przesłanie informacji niezależnie od posiadania tokenu.
12. Wymiana danych uzgadniania, umożliwia przesłanie ograniczonej ilości danych poza akcją użytkownika, dla celów specjalnych.

Usługa sesji jest realizowana przez jednostki sesji (SE - session entity); jednostki te są ściśle związane z funkcjonowaniem protokołu sesji. Jak widać na rys. 1, jednostka może być związana z więcej niż jednym punktem dostępu do usług sesji; użytkownik usług sieciowych - jednostka prezentacji (PE - presentation entity) może korzystać z więcej niż jednego punktu dostępu do usług sesji.

W praktyce można korzystać z trzech podzbiorów protokołu sesji:

- podzbiór podstawowy - BSC;
- podzbiór podstawowy z synchronizacją - BSS;
- podzbiór podstawowy ze sterowaniem akcjami - BAS.

Obszerne informacje na temat warstwy sesji można znaleźć w [2]; usługi sesji opisuje standard ISO DIS 8326, a protokół - ISO DIS 8327. Oba dokumenty zostały przetłumaczone na język polski i wraz z uzupełnieniami stanowią projekty Polskich Norm [6] i [7].



Rys. 2. Przykład struktury jednostki dialogu

Fig. 2. Example of a Structured Dialogue Unit

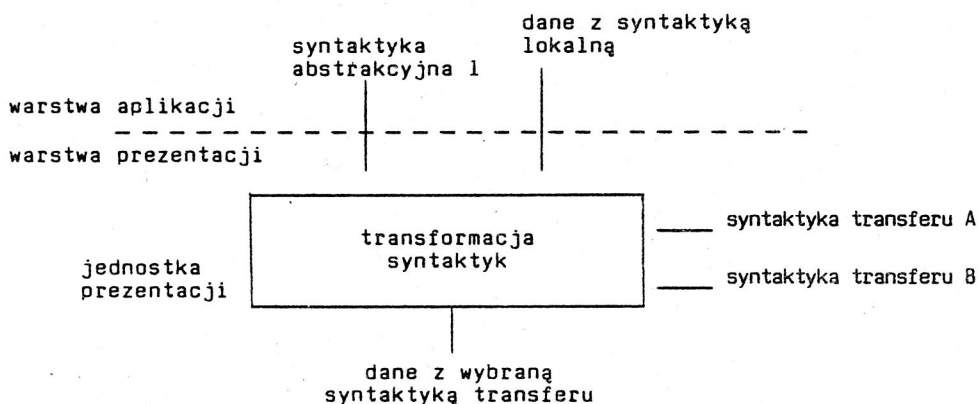
3.2. Warstwa prezentacji

Warstwa prezentacji świadczy usługi warstwie aplikacji. Usługi realizowane są w oparciu o usługi warstwy sesji oraz funkcje realizowane w warstwie prezentacji.

Podstawową funkcją warstwy prezentacji jest transformacja syntaktyk. Podstawowymi pojęciami pozwalającymi zrozumieć tą funkcję są: syntaktyka abstrakcyjna, syntaktyka transferu i kontekst prezentacji. Syntaktyka abstrakcyjna jest zbiorem reguł używanych do formalnej specyfikacji danych, stosowanych w warstwie aplikacji; reguł niezależnych od technik kodowania tych danych. Syntaktyka abstrakcyjna jest zdefiniowana w terminach określających zbiór typów danych. Syntaktyka transferu jest konkretną reprezentacją danych stosowaną w trakcie przesyłania danych pomiędzy dwoma systemami otwartymi. Kontekstami prezentacji określa się szczegółowe kombinacje par, syntaktyka abstrakcyjna - syntaktyka transferu (rys. 3). Kombinacja jest poprawna jeżeli syntaktyka transferu jest w stanie, korzystając z usług prezentacji, odwzorować wszystkie własności syntaktyki abstrakcyjnej.

Zwykle stosuje się jedną syntaktykę abstrakcyjną dla jednego procesu aplikacyjnego. Jeżeli jest jedna syntaktyka abstrakcyjna i jedna syntaktyka lokalna zgodna z syntaktyką transferu, to nie ma praktycznie warstwy prezentacji. Jednym z rodzajów stosowanych syntaktyk transferu jest syntaktyka zawierająca kompresję danych, stosowana dla zmniejszenia kosztów transmisji.

Protokół prezentacji zawiera mechanizmy wyboru syntaktyki transferu akceptowanej przez oba komunikujące się systemy otwarte; wybór ten odbywa się w drodze negocjacji w trakcie nawiązywania połączenia prezentacji.



Rys. 3. Ilustracja odwzorowania syntaktyki abstrakcyjnej na dwie syntaktyki transferu

Fig. 3. Illustration of a Possible Mapping of an Abstract Syntax onto Multiple Transfer Syntaxes

W środowisku OSI, zorientowanym połączeniowo, usługi warstwy prezentacji są następujące:

1. Ustanawianie połączenia prezentacji oraz wybór syntaktyki transferu.
2. Rozłączanie połączenia prezentacji, które może być uporządkowane - z zachowaniem danych lub nieuporządkowane - z możliwością utraty danych.
3. Zarządzanie kontekstem, które umożliwia określanie kontekstów prezentacji.
4. Transfer danych umożliwiający pięć form przesyłania danych:
 - transfer danych ze sterowaniem tokenu;
 - transfer danych bez sterowania tokenem;
 - transfer danych przyspieszonych;
 - wymiana danych uzgodnienia;
 - transfer danych w polu "user data".
5. Sterowanie dialogiem, umożliwiające dostęp do sterowania dialogiem w warstwie sesji.

Usługi prezentacji są realizowane przez jednostki prezentacji (PE - presentation entity). Standardy ISO dotyczące warstwy prezentacji są następujące:

DIS8822 - usługi;

DIS8823 - protokół;

DIS8824 - syntaktyka abstrakcyjna ASN.1.

DIS8825 - reguły kodowania dla ASN.1.

Obszerną informację nt. warstwy prezentacji w języku polskim, można znaleźć w [2].

3.3. Warstwa aplikacji

Rozproszone przetwarzanie informacji wymaga współdziałania dwóch lub więcej procesów aplikacyjnych. Warstwa aplikacji zawiera wszystkie niezbędne do tego celu funkcje, nie realizowane w niższych sześciu warstwach modelu odniesienia OSI. Jest to jedyna warstwa dostarczająca bezpośrednich usług procesom aplikacyjnym.

Powstało już kilka standardów dotyczących warstwy aplikacji, ale ogólny model całej warstwy jest dopiero w stadium rozwoju. Przedstawiony w referacie model warstwy, oparty jest na dokumentach roboczych ISO [3, 4] i dokumencie ECMA [5].

Jak już wspomniano w rozdziale 2, proces aplikacyjny jest elementem rzeczywistego systemu otwartego, biorącym udział w wykonaniu jednego lub więcej, zadań przetwarzania informacji. Procesy aplikacyjne biorące udział w wykonaniu zadania, mogą w tym uczestniczyć równocześnie lub szeregowo. Współdziałanie procesów dla wykonania zadania, może być synchroniczne lub asynchroniczne. Pojedynczy proces aplikacyjny może brać

udział w realizacji jednego lub kilku zadań jednocześnie. Współdziałanie z różnymi procesami aplikacyjnymi może wymagać różnych mechanizmów interakcyjnych.

W środowisku OSI, proces aplikacyjny jest reprezentowany przez jeden lub więcej jednostek aplikacyjnych (AE - application entity), rys. 1. Każda jednostka aplikacyjna reprezentuje jeden i tylko jeden proces aplikacyjny. Jednostka aplikacyjna realizująca funkcje określone dla konkretnej komunikacji nazywa się powołaną jednostką aplikacyjną (Application Entity Invocation). Innymi słowy, jednostka powołana istnieje na czas realizacji określonego zadania sieciowego.

Każda jednostka aplikacyjna składa się ze zbioru możliwości komunikacyjnych (procesu aplikacyjnego) zwanych elementami usług aplikacyjnych (ASE - Application Service Element). Element ASE jest zbiorem powiązanych funkcji, które mogą realizować komunikację pomiędzy jednostkami aplikacyjnymi. Przykładami elementów ASE są następujące typy elementów: FTAM, JTM, VT, CCR i sterowanie asocjacją. Każdy typ elementu ASE jest określony przez własne usługi i protokoły.

Jednostka aplikacyjna korzysta z usług warstwy prezentacji.

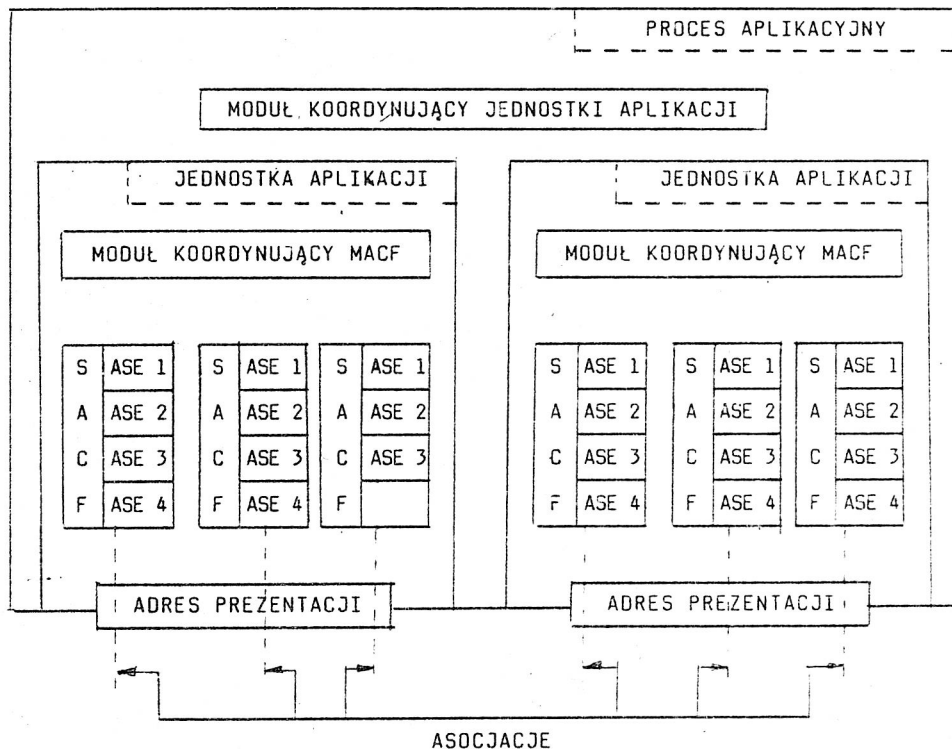
Asocjacja aplikacyjna jest to współdziałanie jednostek aplikacyjnych. Ustanawianie i zwalnianie asocjacji realizuje specjalny element ASE zwany elementem sterującym usługą asocjacji (ACSE - Association Control Service Element). ACSE inicjowany jest przez proces aplikacyjny albo przez inny element ASE. ACSE z kolei inicjuje połączenie prezentacji. Elementy ASE, które korzystają bezpośrednio z usług prezentacji i świadczą usługi innym elementom ASE, nazywają się wspólnymi elementami usług aplikacji (CASE - Common Application Service Element). Inne elementy ASE (zwane SASE - Specific Application Service Element), mogą korzystać z usług elementów ASE lub bezpośrednio z warstwy prezentacji.

Kontekst aplikacyjny jest to jednoznacznie określony zbiór elementów ASE zapewniających współdziałanie jednostek aplikacyjnych, za pomocą asocjacji. Może być powołanych kilka kontekstów aplikacyjnych na jednej asocjacji. Jednostka aplikacji może utrzymywać asocjacje pojedyncze lub wiele równoległych asocjacji. Asocjacją pojedynczą steruje moduł SACF (Single Association Controlling Function), rys. 4. W drugim przypadku jednostka aplikacji posiada dodatkowo moduł MACF (Multiple Association Controlling Function), rys. 4.

W zależności od rodzaju procesu aplikacyjnego stosuje się kilka jednostek aplikacyjnych, każdą z pojedynczymi asocjacjami (i co najmniej jednym kontekstem) lub jedną jednostką aplikacyjną z wieloma modułami SACF i modułem MACF.

Jeżeli semantyka procesu aplikacyjnego ma charakter komunikacyjny to lepiej stosować jedną jednostkę aplikacyjną i koordynację wewnątrz niej;

w przypadku semantyki o charakterze przetwarzającym wygodniej jest stosować kilka jednostek aplikacyjnych i koordynację pomiędzy nimi, np. przy obsłudze baz danych stosuje się dwie jednostki aplikacyjne (a więc dwa protokoły) - jedną do obsługi dostępu a drugą do wyszukiwania - działające niezależnie.



Rys. 4. Proponowany model warstwy aplikacji

Fig. 4. Proposed Application Layer Model

Elementy usług aplikacyjnych (ASE)

Dotychczas opracowane zostały lub znajdują się w opracowaniu następujące elementy:

1. Element sterujący usługą asocjacji (ACSE); realizuje on ustanawianie i zwalnianie asocjacji aplikacyjnych, identyfikuje kontekst aplikacyjny stosowany w asocjacji, przesyła informacje użytkownika pomiędzy jednostkami aplikacji, zarządza kontekstem aplikacyjnym. Opis ACSE zawarty jest w dokumentach: ISO 8649/1 i 2 - usługi. ISO 8650/1 i 2 - protokół.

2. CCR (Commitment, Concurrency and Recovery); utrzymuje aplikację rozproszoną wtedy gdy w przetwarzaniu uczestniczy więcej niż dwie jednostki aplikacyjne i więcej niż jedna asocjacja; zapewnia koordynację działań na odrębnych asocjacjach. Opis CCR zawarty jest w dokumentach: DIS9804 - usługi (poprzednie oznaczenie - DIS8649/3) oraz DIS9805 - protokół (poprzednie oznaczenie - DIS8650/3).

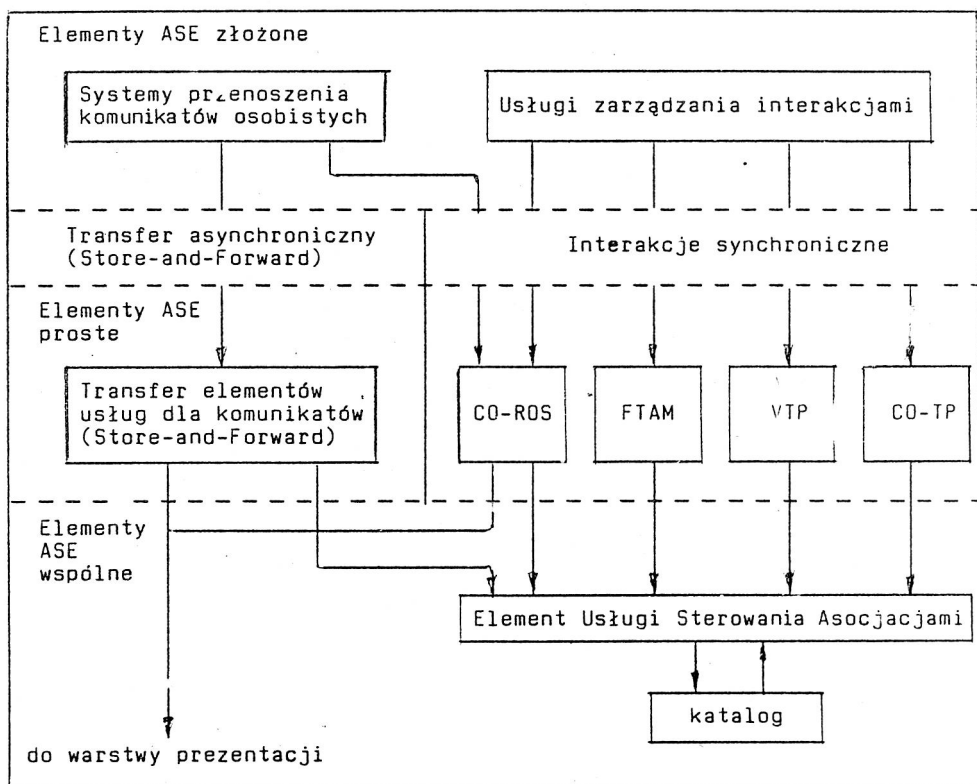
Elementy dla konkretnych zastosowań (SASE):

3. Transfer zbiorów (FTAM); umożliwia użytkownikom zbiorów w systemach otwartych, transfer, dostęp oraz zarządzanie zbiorami. Opis FTAM zawarty jest w dokumentach: IS08571/1+4.
4. Transfer zadań (JTM); umożliwia realizację zadań w sieci połączonych systemów otwartych. Opis JTM zawarty jest w dokumentach: DIS8831 - usługi i DIS8832 - protokół.
5. Wirtualny terminal (VTP); pozwala na transfer i manipulację danymi, niezależnie od sposobu wewnętrznej reprezentacji danych stosowanej przez użytkowników. Opis VTP zawarty jest w dokumentach: IS09040 - usługi i IS09041 - protokół.
6. Transfer komunikatów (MT); utrzymuje transfer komunikatów w kategorii "store-and-forward" (poczta elektroniczna); jest częścią systemu MUTIS (Message Oriented Text Interchange System). Dostarcza komunikaty do jednego lub więcej odbiorców w określonym czasie i, jeżeli potrzeba, dokonuje transformacji syntaktyk. Z systemem MOTIS związane są następujące standardy: DIS8505, DP9065, DIS8883, DP9073, DP9066, DIS10021/1+7.
7. System dla prac biurowych (TOS); zawiera procedury związane z automatyzacją prac biurowych. Z systemem TOS związane są następujące standardy: DIS8613/1+7, DP8879, DP8884, DIS9069, DP9070, DP9063.

Ponadto opracowywane są elementy usług dla grafiki komputerowej (DP9592), systemów bankowych (IS08583), systemów automatyzacji procesów produkcyjnych (IS09506), autoryzacji dostępu (DIS9834) oraz kryptografii.

W praktyce, nie wszystkie elementy konkretne, wymagają stosowania elementów wspólnych. Rys. 5. przedstawia wersję roboczą (wg [3]) rozmieszczenia elementów w jednostce aplikacji.

Protokół warstwy aplikacji jest realizowany przy użyciu pewnej liczby elementów ASE. Elementy te mogą być użyte sekwencyjnie lub w pewnych kombinacjach; np. transfer zbiorów wymaga zastosowania elementów FTAM (SASE) i ACSE (CASE). Typowa struktura jednostki danych protokołu aplikacyjnego (APDU) jest określona za pomocą pojedynczej syntaktyki abstrakcyjnej. Jeżeli wymagana jest kombinacja APDU z różnych elementów ASE, to wymagane jest odniesienie do kilku syntaktyk abstrakcyjnych.



Rys. 5. Trójpoziomowy model elementów ASE wg [3]

Fig. 5. Three Levels of Application Service Elements

W środowisku OSI rozważane są trzy sposoby komunikacji pomiędzy systemami otwartymi:

- połączeniowy;
- bezpołączeniowy;
- store-and-forward.

Objaśnienia wymaga trzeci sposób, którego reprezentantem jest poczta elektroniczna, stanowiąca obecnie jedną z najważniejszych usług sieci komputerowych. Komunikacja taka realizowana jest następująco:

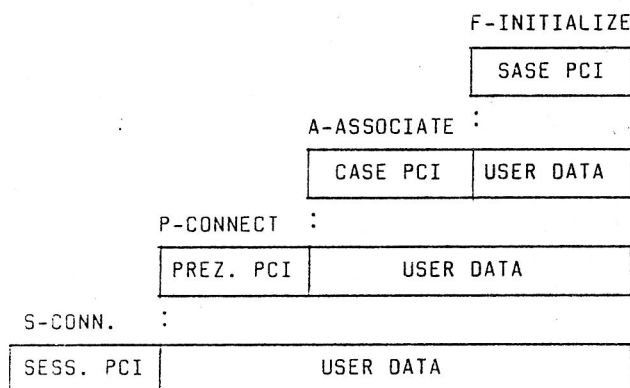
- protokół najwyższego poziomu warstwy aplikacji realizuje komunikację bezpołączeniową;
- protokoły niższego poziomu warstwy aplikacji oraz pozostałych warstw modelu (1-6), realizują komunikację połączeniową.

Umożliwia to komunikację wtedy, gdy system końcowy jest aktualnie niedostępny, np. wyłączony terminal użytkownika.

Adresowanie w OSI jest przedmiotem standardu ISO7498/3. Proces ustanawiania asocjacji przebiega następująco:

- użytkownik podaje nazwę jednostki aplikacyjnej, z którą chce ustanowić asocjację;
- nazwa wysyłana jest do katalogu (Directory), w którym szukany jest adres prezentacji związany z nazwą;
- w innym katalogu w warstwie sieciowej dokonywane jest odwzorowanie na adres sieciowy jednostki aplikacyjnej z którą nawiązywana jest asocjacja.

Schemat blokowy jednostek poszczególnych protokołów przedstawia rys. 6. Ogólne zasady organizacji nazw i adresacji zawarte są w dokumencie ISO7498/3.



Rys. 6. Jednostki protokołów w czasie ustanawiania asocjacji

Fig. 6. Protocol-Data-Units in Connection Control

W środowisku OSI wymagane jest planowanie, organizacja, zarządzanie i sterowanie zasobami systemów otwartych. Istnieją trzy kategorie tych funkcji:

- zarządzanie systemem;
- zarządzanie (N)-warstwą;
- operacyjne sterowanie (N)-warstwą.

Zarządzanie systemem odnosi się do wszystkich warstw modelu, kilku warstw lub pojedynczej warstwy; jest ono usytuowane w warstwie aplikacji. System jest opisany w dokumentach ISO DP9595 (usługi) oraz ISO DP9596 (protokół). Ogólne zasady zarządzania zawarte są w ISO7498/4. Dwie pozostałe kategorie są przedmiotem standardyzacji w poszczególnych warstwach modelu odniesienia. Ważną rolę w modelu spełnia wspomniany już wyżej katalog (Directory). Pozwala on klientom (użytkownikom i aplikacjom) korzystać tylko z nazw jednostek aplikacyjnych, dokonując odpo-

wiedniego odwzorowania adresów. Organizacja katalogu jest taka, że dodawanie, usuwanie i zmiana lokalizacji fizycznej obiektów, nie oddziałuje na pracę sieci; umożliwia on dołączanie do listy nazw, nowych nazw lub ich atrybutów oraz umożliwia ich wyszukiwanie. Katalog jest opisany w dokumencie ISO DIS9594. Informacje zawarte w katalogu są rozproszone pośród rzeczywistych systemów otwartych. Docelowo przewiduje się zastosowanie dwóch protokołów: dostępu użytkownika do katalogu oraz protokołu zarządzania interakcjami pomiędzy rozproszonymi częściami katalogu. W modelu wyższych warstw, oprócz elementów usług, rozważane są następujące zagrożenia:

1. System zabezpieczeń (ISO7498/2) obejmujący:
 - legalizację,
 - kontrolę dostępu,
 - zachowanie poufności danych,
 - integralność danych,
 - mechanizmy zabezpieczeń i ich rozmieszczenie w poszczególnych warstwach,
 - ocena jakości usług.
2. Techniki opisów formalnych (FDT); opisy usług i protokołów przedstawiane w języku naturalnym, są wprawdzie łatwiej zrozumiałe, ale zawierają dwuznaczności, są niekompletne i mogą prowadzić do błędów w implementacji oraz prowadzić do niekompatybilności implementacji opartych na tych samych standardach. W celu uniknięcia tego wprowadza się techniki opisów formalnych. ISO rozwija dwie techniki:
 - LOTOS - ISO8807,
 - ESTELLE - ISO9074.
3. Notacja syntaktyki abstrakcyjnej (ASN.1). Warstwa aplikacji wymaga mechanizmu do opisu danych w sposób abstrakcyjny, nie określający sposobu kodowania. Mechanizm ten musi być wystarczający do opisu szerokiego i różnorodnego wachlarza struktur danych i powinien umożliwiać łatwe kodowanie w warstwie prezentacji. Oba te warunki spełnia notacja syntaktyki abstrakcyjnej (ASN.1). Określa ona pewną liczbę bazowych typów danych i związanych z nimi wartości oraz dostarcza konstrukcji, które pozwalają budować złożone struktury danych, zwane typami strukturalnymi. Typy bazowe to: Boole'owskie, całkowite, ciągi bitów i ciągi oktetów. Natomiast konstrukcje to: sekwencja, zbiór i wybór (choice). Notacja ASN.1 stosowana jest przy tworzeniu jednostek danych protokołów aplikacji (APDU) i prezentacji (PPDU). Możliwe jest stosowanie makr określających nową notację, z możliwością powoływania typów ASN.1. ASN.1 opisana jest w dokumencie ISO8824, a reguły kodowania w ISO8825.

4. Metody testowania zgodności. Każda konkretna implementacja usług i protokołów powinna być poddana testowaniu na zgodność ze standardami. Metody testowania muszą być kompletne i zapewniać bezkolizyjne łączenie sieci po ich testowaniu; dlatego są one przedmiotem standaryzacji. Standard ISO opisujący metody testowania zgodności zawarty jest w dokumencie DP9464/1+7. Dokument zawiera między innymi opis języka definiowania testów.

Przyjęty uniwersalizm modelu, a jednocześnie konieczność ścisłego sformułowania procedur dla szerokiego wachlarza różnorodnych zastosowań, przy zachowaniu kanonicznego założenia OSI o nieograniczaniu implementatorów modelu, doprowadziły do znacznej złożoności modelu wyższych warstw. Mimo przedłużania się prac nad modelem, zostaną one w niedługim czasie doprowadzone do końca; zostawiając w nim miejsca na dalszy rozwój zgodnie z filozofią OSI. Główny rozwój prowadzony będzie w kierunku dołączania nowych rzeczywistych systemów otwartych, nowych procesów aplikacyjnych oraz dalszych elementów usług ASE.

Literatura

- [1] Bilski E., Usługi i standardy sieci transmisji danych. Materiały konferencji nt. Sieci komputerowe. Teoria, technika, zastosowania. Politechnika Wrocławska, Wrocław 1987.
- [2] Stroiński M., Wietrzyk J., Warstwa sesji i prezentacji modelu odniesienia ISO/OSI - Funkcje, Usługi i Protokoły. Materiały Konferencji nt. Sieci komputerowe. Politechnika Wrocławska, Wrocław 1985, str. 61-80.
- [3] Application Layer Struktury. ISO/TC97/SC21/N1165. April 18, 1986.
- [4] Technical Report for Guide to Developing Application Layer Standards (ANSI), ISO/IEC JTC1/SC21/N2715. 1988-05-09.
- [5] TP Model and Application Layer Struktury. ECMA/TC32-TG2-September 1987.
- [6] Projekt PN: Sieci komputerowe. Usługi i protokoły. Usługi sesji zorientowane połączeniowo. Politechnika Wrocławska i IKSAiP, 1988.
- [7] Projekt PN: Sieci komputerowe. Usługi i protokoły. Protokół sesji zorientowany połączeniowo. Politechnika Wrocławska i IKSAiP, 1988.

Upper Layers Architecture of the RM OSI

In the paper, results of work concerning ULA, leading by ISO and ECMA, are presented. Also the standards of the Application Layer, including their short descriptions, are specified. The paper is based on the documents which have been published in the period from 1985 to half of 1988.

system SKOT w sieci KASK,
ewidencja zasobów systemu
rozliczanie systemu SKOT

D.BOROWSKA, W.BOROWSKI, R.FILIPEK,
A.GOSPODAROWICZ, Z.ŁEKAWA, S.ZAJAC

EWIDENCJA I ROZLICZANIE UŻYTKOWNIKÓW SYSTEMU SKOT W SIECI KASK

Referat podejmuje problem rozliczania zadań wielu użytkowników pracujących pod kontrolą systemu SKOT w sieci KASK. Przy ewidencjonowaniu zasobów systemu proponuje się wykorzystać własności systemu do automatycznego tworzenia statystyki. W niniejszym opracowaniu proponujemy metodę, która rozdziela wykorzystane zasoby systemu na te transakcje, które je wykorzystywały i z uwzględnieniem ich współbieżnego czasu pracy.

1. WSTĘP

System kontroli i obsługi terminali SKOT jest uniwersalnym systemem bazy danych/transmisji danych uruchamianym na komputerach Jednolitego Systemu. System umożliwia transmisję danych z terminala do komputera, przetwarzanie danych, dostęp do zbioru danych/baz danych oraz retransmisję danych do odpowiedniego terminala. Skot działa pod kontrolą systemu operacyjnego. Rozliczany jest jako jedno z zadań tego systemu zgodnie z obowiązującym w danym ośrodku komputerowym systemem rozliczania zadań.

Aby umożliwić użytkownikom jednocześnie korzystanie z systemów współpracujących ze SKOT-em oraz przetwarzanie własnych programów (transakcji), system SKOT w sieci komputerowej KASK musi udostępnić w wyznaczonym czasie wszystkim użytkownikom swoje zasoby. Stąd przed administracją ośrodka komputerowego staje zadanie określenia i rozliczenia w jakim stopniu poszczególni użytkownicy wykorzystali informacyjno-obliczeniowe zasoby ośrodka. Koszty, charakteryzujące stopień wykorzystania zasobów ośrodka komputerowego przez poszczególnych użytkowników systemu SKOT w sieci KASK, powinny być skorelowane ze złożonością przetwarzanych zadań. Użytkownik powinien też znać jak poszczególne jego działania wpływają na koszt jego pracy i z jakich składników ten koszt się składa.

Koszt pracy użytkownika przy tym samym scenariuszu dialogu na jego terminalu nie powinien zależeć od aktywności innych użytkowników.

W niniejszym referacie podejmuje się problem rozliczania zadań wielu użytkowników pracujących pod kontrolą systemu SKOT w sieci KASK oraz proponuje się sposób rozwiązania tego problemu.

2. ZMIANY W SYSTEMIE SKOT Z PUNKTU WIDZENIA SYSTEMU ROZLICZAJACEGO

System rozliczający wykorzysta informacje statystyczne dostarczane przez moduły SKOT-a, co pozwoli ograniczyć do minimum zmiany w tych modułach. Jest to podyktowane tym, aby nie wydłużać czasu pracy systemu SKOT przez rozbudowę jego modułów i uprościć adaptację systemu rozliczającego w poszczególnych ośrodkach komputerowych. Z punktu widzenia systemu rozliczającego, z systemu SKOT będą pobierane dane identyfikujące użytkownika oraz dane dotyczące uruchamianych przez niego transakcji.

W systemie rozliczającym, użytkownicy systemu SKOT będą identyfikowani za pomocą symbolu konta (określonego w Tablicy Znaku Włączenia SNT), którym będą posługiwać się w chwili zgłoszenia do systemu.

Podstawą rozliczania poszczególnych użytkowników sesji systemu SKOT będzie ewidencja czasów startu i zakończenia transakcji uruchamianych z określonego terminala, wraz ze stanem liczników dotyczących częstości korzystania z zasobów systemu (terminala, zbiorów, modułów programowych). Pod uwagę będzie brany również priorytet transakcji. Informacje te można uzyskać dzięki temu, że SKOT zezwala na dostęp do tablic, w których te dane są zawarte:

- ilość operacji wejścia/wyjścia na/z terminal - Tablica Sterowania Terminalami (TCT),
- ilość odwołań do zbiorów, oraz określonych operacji - Tablica Sterowania Zbiorami (FCT),
- ilość odwołań do modułów programowych - Tablica Programów Przetwarzających (PPT),
- ilość wykorzystania przeznaczeń - Tablica Sterująca Przeznaczeniem (DCT),
- priorytet transakcji - zawarty w Obszarze Sterowania Transakcjami (TCA), przydzielonym przez SKOT dynamicznie w czasie inicjacji transakcji.

Przy ewidencjonowaniu w/w informacji, wykorzystane zostaną własności systemu SKOT, dotyczące automatycznego tworzenia statystyki. Kod źródłowy Programu Zarządzania Zadaniem (KCP) zostanie rozszerzony o procedurę użytkownika, która w czasie trwania sesji systemu SKOT, będzie sukcesywnie rejestrować te dane, a po jej zakończeniu przekaże je w formie zbioru statystyk dla celów rozliczania.

3. KONCEPCJA SPOSOBU EWIDENCJONOWANIA I ROZLICZANIA UŻYTKOWNIKÓW SYSTEMU SKOT

Rekordy zbioru statystyk będą zawierały dane o wykorzystaniu zasobów systemu SKOT w momencie startu i końca każdej transakcji. Na podstawie tych danych należy określić, w jakim stopniu użytkownicy systemu SKOT wykorzystywali jego zasoby. Proponujemy metodę, która pozwoli rozdzielić wielkość wykorzystywanych zasobów systemu na poszczególnych użytkowników pracujących podczas sesji, z uwzględnieniem stopnia wykorzystania danego zasobu przez poszczególną transakcję oraz współbieżnego czasu pracy tych transakcji. W tym celu sesję systemu SKOT traktujemy jako nieciągły zbiór zdarzeń, gdzie elementarnym zdarzeniem jest start lub koniec transakcji. Przedział czasu między zdarzeniami oraz wartość przyrostu liczników rejestrowanych zasobów, dotyczą więc transakcji aktywnych w danym przedziale czasu. Wielkość wykorzystania danego zasobu systemu zostanie rozdzielona na współbieżnie pracujące transakcje proporcjonalnie do ich priorytetu, jako że priorytet jest parametrem jednoznacznie określonym dla każdej transakcji. Każde elementarne zdarzenie (start lub koniec transakcji) powoduje zmianę ilości aktywnych transakcji (w); start kolejnej transakcji zwiększa tę ilość o 1, a koniec zmniejsza o 1. Dla każdego elementarnego zdarzenia liczymy przyrost wartości licznika rejestrującego i -ty zasób systemu SKOT, oraz dzielimy go przez sumę priorytetów ($PRTY$) transakcji aktywnych w badanym przedziale czasu, otrzymując tym samym jednostkowy wskaźnik (z_i) wykorzystania i -tego zasobu systemu, co można zapisać następująco:

$$z_i = \frac{\Delta L_i}{\sum_{j=1}^w PRTY_j}$$

Mnożąc wskaźnik (z_i) przez priorytet transakcji aktywnej w badanym przedziale czasu uzyskamy wielkość i -tego zasobu systemu, przypadającą na j -tą transakcję.

Dane o pracy użytkownika w ramach sesji systemu SKOT, dostarczane przez statystykę systemu, nie uwzględniają zasadniczych zasobów systemu. Przy występujących ograniczeniach pamięci operacyjnej, istotnym zasobem z punktu widzenia systemu rozliczającego jest zajętość pamięci operacyjnej. Zasobem, który charakteryzuje stopień złożoności przetwarzanego zadania jest czas pracy arytmometru. Dane dotyczące wykorzystania tych zasobów systemu, będą ewidencjonowane na podstawie informacji dostarczanych przez standardowy pakiet statystyki systemu kontroli i rozliczania zadań SMF.

System SMF ewidencjonuje sesję systemu SKOT jako jedno zadanie,

które pracowało pod kontrolą systemu OS/JS, a więc i dane dotyczące pamięci operacyjnej oraz czasu pracy arytmometru będą globalne dla całej sesji systemu SKOT. Aby określić, w jakim stopniu dana transakcja korzystała z omawianych tu zasobów systemu, wykorzystamy ich zależność od zasobów, które można odnieść do danej transakcji. Zależność tę wyznaczymy w oparciu o dane badań empirycznych, na podstawie których wyznaczymy funkcję zależności czasu pracy arytmometru (Y_1) oraz wielkość zajętości pamięci (Y_2), od danych o wykorzystaniu zasobów systemu, rejestrowanych przez SKOT. Zmiennymi objaśniającymi dla zmiennej Y_1 i Y_2 będzie czas pracy transakcji (x_1), licznik wejścia/wyjścia na ekran (x_2), licznik wywołań modułów (x_3), licznik odwołań do zbiorów typu DBD (x_4), licznik odwołań do zbiorów typu sekwencyjnego (x_5) oraz priorytet transakcji (x_6). Posługując się metodami statystycznymi, wyznaczymy funkcje zależności:

$$Y_1 = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \dots + \alpha_k x_k + \varepsilon$$

$$Y_2 = \alpha_0 + \alpha_1 x_1 + \alpha_2 x_2 + \dots + \alpha_l x_l + \varepsilon$$

gdzie: α - parametry funkcji,

ε - element losowy, oznaczający że jest to funkcja przybliżona.

Przedstawiona metoda, pozwoli przewidywać wykorzystanie omawianych tu zasobów systemu SKOT w warunkach rzeczywistej eksploatacji systemu. Mając dane o wykorzystaniu zasobów, które dostarczy statystyka systemu SKOT, oraz wyznaczoną funkcję zależności, można ustalić wielkość zasobów $\hat{Y}_1$ i $\hat{Y}_2$ dla każdego użytkownika systemu.

Uzyskane w wyniku rozliczania, wielkości każdego zasobu systemu wykorzystywanego przez daną transakcję, wyrażone będą w postaci kwotowej na podstawie cennika obowiązującego w danym ośrodku komputerowym i wprowadzone w formie zestawienia uwzględniającego identyfikator/konto użytkownika.

REGISTRY AND ACCOUNT FOR USERS OF THE SKOT SYSTEM FROM KASK NETWORK

The problem of account the task of many users working under SKOT system from KASK network control is considered. Then the solution of this problem is presented.

УЧЕТ И РАСЧЕТ ПОЛЬЗОВАТЕЛЕЙ СИСТЕМЫ SKOT В СЕТИ KASK

В докладе рассматривается проблема расчета задач многих пользователей работающих под контролем системы SKOT в сети KASK, а также предлагается способ решения этой проблемы.

Andrzej BRODNIK *

Marian SPEGEL **

Barbara KOROUSIC *

Local area network,

industrial network

A LOW - COST INDUSTRIAL NETWORK FOR REAL - TIME APPLICATIONS

The article describes a new low-cost general-purpose network for industrial applications. It defines the data link and the network layers of the seven-layer OSI model. The network layer access method of the protocol is a combination of the polling and the token passing principles. The results of the throughput analysis of the protocol are also presented. Two practical examples of application of this network are also discussed.

1. INTRODUCTION

In the transition from the industrial to the information society, the value of information is raising. Its value is often in direct relation with the time needed to get it. This rule can be observed in almost every human activity, including in control of industrial processes.

An manufacturing process generally consists of primary storage, of production (with possible intermediate storages of semi-products), and of storage of final products. The amounts of stocks vary with time, depending on a number of parameters (such as input material characteristics) as well as on number of stochastic variables (such as production volume demand). Generally speaking, bigger stocks also imply more expensive production, because capital is idly tied to stocks rather than to its active use for business. Consequently, modern production methods tend to keep the stocks as low as possible; their latest and much publicized addition is the so called "just-in-time" method of efficient managing of stocks of production materials. However, these methods basically rely on some form of dynamic (demand-driven) planning of production.

Efficient planning of a production process-and in particular, efficient dynamic planning of production-cannot be done without true and

* Department of Computer Science, "Josef Stefan" Institute, Ljubljana, Yugoslavia

timely information on the state of the production process. In essence, effective monitoring of a production process can only be achieved by acquiring information on the status of production process directly from the process itself and in real-time. Real-time data acquisition (RTDA) means that data about production events are collected at the time of their occurrence, so that can be immediately transmitted to monitoring processor and that proper action can be promptly taken in response to collected data.

In general, the biggest problem with RTDA systems is their distribution over large production areas, covering possibly several square kilometers; the multiplicity of control points (possibly several hundreds or even thousands) where data are generated or control information needed is no lesser problem. An adequate answer to these problems today are local area networks (LANs), providing efficient way of connecting large number of information sources and/or destinations commonly referred to as terminals.

Terminals are used to enter the data, to transmit the data to the host computer, and to receive return information from the computer and/or from other terminals in order to display it to the user and/or to use it for the process control.

Nowadays, there exist a number of local area networks for industrial purposes, such as Ethernet [1], MAP [2], and CSMA/CD MAC [3] LAN's. In industry, the most widely used is the MAP network, developed by General Motors to monitor their production lines.

In this article, we describe a low-cost industrial network and the corresponding industrial terminals, designed to suite a specific application with relatively low rate of information processing. The network is defined according to the ISO standards; only the second (data link) and the third (network) layer of seven layer (according to the OSI model) are described, and we concentrate on the network layer of our protocol. Contrary to the Ethernet LAN, our network is relatively inexpensive and therefore more suitable for low-cost industrial applications. It is also cheaper than the MAP architecture, but the price for inexpensiveness is relatively low speed of our LAN. For example, the response time provided by our network is measured in seconds rather than milliseconds as in MAP. Nevertheless, our LAN proved to be quite suitable for industrial processes with small flow of data, for example in garment industry.

2. PROTOCOLS

The main purpose of connecting two separate devices is to enable them to exchange data. Just as natural languages are used for human communication, computers also require some (but definitely simpler) kind of languages for communication among devices. These communication lan-

guages are called protocols.

To serve well in a broad community of users and applications, protocols must be consistent, layered and standardized as much as possible. Whereas the first demand is obvious, the second one is dictated by the economy. To support these requirements, the International Organization for Standardization developed a number of International standards describing different layers of OSI Model [1].

The described seven layers of the OSI Model from the first (physical) upwards to the last-seventh (i.e., the application) layer all conform to the same principal idea: each layer presents the demands to the next lower layer and serves the demands from the next upper layer [1].

Furthermore, each layer works as a separate process or better, as a monitor [4]. This ensures the consistency of each layer.

The purpose of this article is present the definition of our new protocol on the network layer, to describe the hardware and software implementation of our LAN, as well as to present early results on the timing analysis of our network. Finally, we discuss some experimental results from the first two installations of our LAN in industrial applications.

2.1 Network Layer Protocol

The semantics of the protocol for our network is designed specifically for the purpose of data acquisition in a given industrial application. Yet our protocol supports essential characteristics of a general purpose LAN. Most of the time, the data are collected in one terminal point which is the master at that time. All remaining terminals are slaves and answer the master terminal according to the customary polling protocol.

Sometimes it is necessary to send data to another terminal point. This means that the location of the master on the LAN must be changed. Such a mechanism is usual in general purpose LANs using tokens such as Ethernet [1]. Therefore, tokens were introduced in our network layer protocol as well. Accordingly, our protocol consists of both the polling and the token passing principles. We designed it according to several standards [5,6,7,8,9]. Its messages can be divided, according to its double nature, into two main groups: the control group and the data group. Their basic structure is shown in Figure 1.

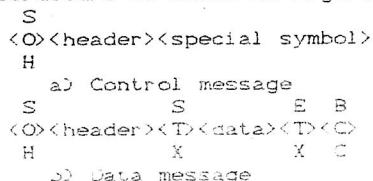


Figure 1. Message format

The data group is very simple and need not be discussed further. It is used only as the data carrier. It is the first group that shows the real flavour of our new protocol.

As mentioned previously, our LAN defines the lower three layers of the OSI Model. As the structure of the network layer will be discussed in details later, we begin with brief description of the physical and the data link layers. The physical layer is defined with a twisted pair of wires or any other kind of low-cost electrical signal carrier. In order to keep the cost down, the standard RS485 [10] was employed as well. On the data link layer, we employ the SDLC [10] standard supporting both point-to-point and broadcast communication.

The topology of our network follows from the characteristics of the described three layers of OSI model. It is basically a bus structure (Figure 2); but it can be extended to a general graph as shown in Figure 2 since some terminals can be master stations and since two branches can be connected to each of these stations. The role of these master stations connecting two branches is obvious.

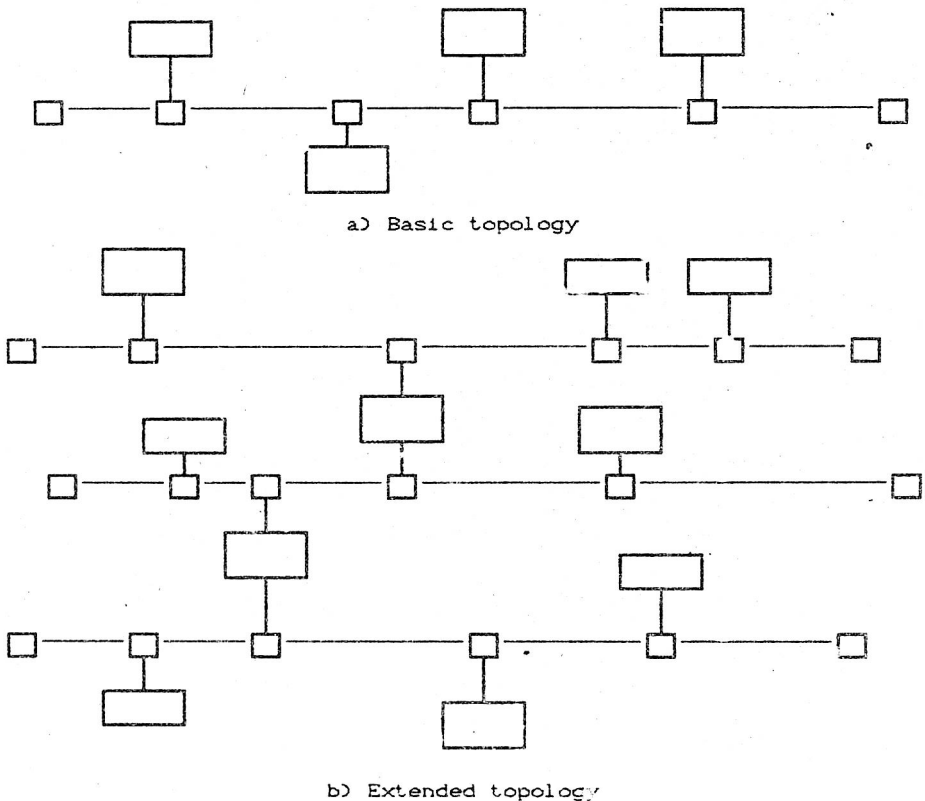


Figure 2. Network topology

According to time analysis [11] and experimental results achieved so far, the new network with 32 terminals can support the frequencies up to 100 events per minute. Considering that in the contemporary manufacturing control systems the terminals are intelligent and therefore the exchanged information is terse, the described capacity of our network satisfies real-time requirements as expressed by J.A. Stankovic [12].

2.2 Enquiry Message

This message is used in normal operation by the master to request in sequence each terminal to release its data. The Figure 3 shows the structure of the message as well as its ENQ form as its special instance. The ENQ is used to divide time in quanta. Each terminal is polled only when the master has information that the terminal is attached to the network. In this way, the unused time on the network can be significantly reduced.

S	E
<O><receiver address><sender address><N>	
H	Q

a) General message (enq)

S	E
<O><FF><sender address><N>	
H	Q

b) Specific message (ENQ)

Figure 3. Format of enquiry message

Our network works in the following way. At the very beginning of the network life, the master marks the first time quantum by sending the ENQ form of the message. Then it starts interrogating all terminals known to be connected, and ends with a cycle sending another ENQ message. This procedure proceeds indefinitely (Figure 4).

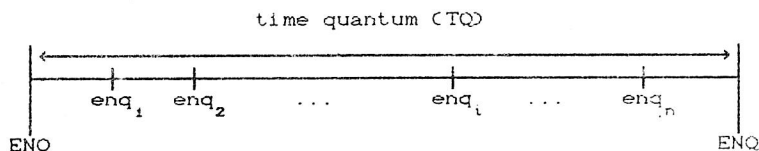


Figure 4. Time quantum

On the other end, the slave terminals must all the time listen to the network. If they are not asked in one time quantum they simply send after the ENQ message an acknowledge message to signal master that they are connected to the network (Figure 5).

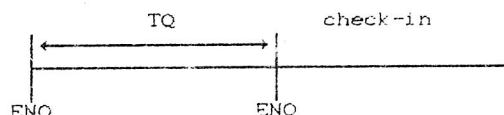


Figure 5. Check-in procedure

The nature of terminals (slave or potential master) is defined at the startup time of the network. If there is only one master on the network, there is no problem. However, when there are two of them or more, they must make an agreement who is a true master. To achieve this agreement, a simple trick known from the Ethernet protocol is used [1]. Each potential (according to its internal data) master terminal waits for some time and then sends the ENQ message. If collision occurs on the line, the procedure is repeated, but with double waiting time (Figure 6). This procedure is finite as the number of terminals on network is finite [11] and the real master is the one who succeeded to send its ENQ message properly.

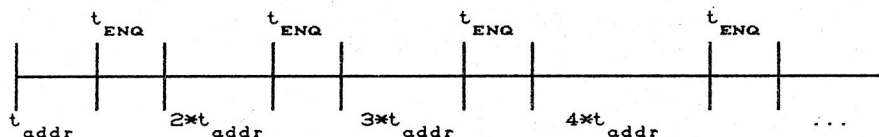


Figure 6. Start-up procedure

2.3 Acknowledge and Not-Acknowledge Messages

These messages (Figure 7) also serve double role. As it has been already mentioned, they are used by network terminals to check-in after ENQ message. This, however, is their minor role; their main purpose is to confirm the correctness of received data. They are used according to the so called window mechanism [1] meaning that all messages previous to the last confirmed are understood to be correct (Figure 8).

2.4 Token Messages

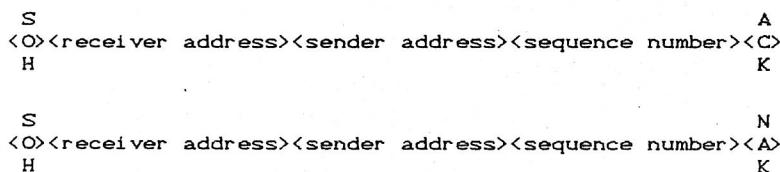


Figure 7. Format of acknowledge and not-acknowledge message

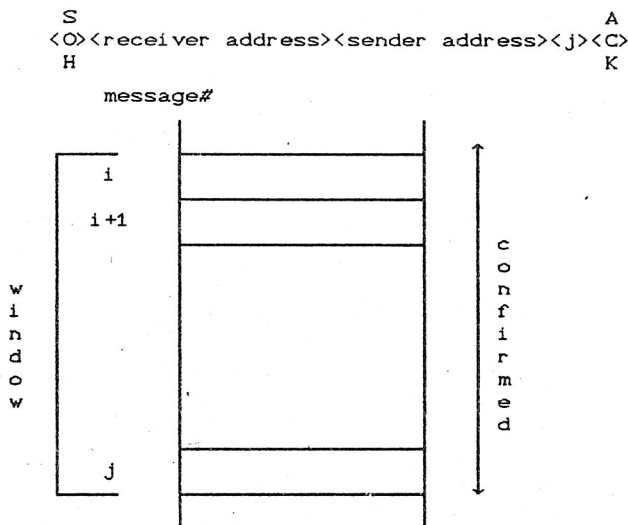


Figure 8. Data confirmation

There are two such messages (Figure 9). The first one is used by the slave terminal to ask the master for the token and is sent after the regular ENQ message. The second message is in fact the true token and is sent after the token request. The reception of the token must be confirmed with an ACK message.

3. NETWORK USAGE

The entire network development was divided into two phases. The first one was to develop the support for the usual polling principle. Its installation was made at the UNIS company, manufacturer of internal transportation systems in Ljubljana, and another installation is in preparation for late June at the IKA garment industry at Ajdovscina.



Figure 9. Token message

The first implementation of our network provides for connecting of sixteen terminals. The terminals are intelligent CCD cameras for recognition of bar codes written on articles presented to the cameras by conveyer belts. The data are sent directly to the master terminal which is connected to an IBM PC. The computer collects the data and stores

them on the disk. All software was written in Modula-2 using the IN-BED tool [13,14] for the development of embedded applications. The time spent for coding of the network software was about six man-months.

The next installation is planned for the end of June in the IKA clothing industry. This installation consists of our new general purpose 18088 - based data collection terminals, each providing one serial and five parallel eight-bit wide communication ports. Furthermore, two network branches can be connected to each terminal. The peripheral devices connected to terminals are keyboards, displays, light pens and bar-code readers. The distributed application under design will acquire from the described network the exact status of production down to each workplace and each working operation.

4. CONCLUSION

In this article, we have described main characteristics of a new low-cost local-area network intended for industrial use in manufacturing processes with low rate of information flow. We have demonstrated how a theoretically interesting combination of the standard low-cost solution (polling) and of standard token passing principle can be advantageously used to solve practical networking needs in real-time monitoring of production processes.

In the first phase of the development of our network project, we have successfully implemented the single-master network with the polling principle. In the next phase of our work, we plan to introduce the token principle and generalize the network topology.

Furthermore, we also plan to improve on the data terminal by adding D/A and A/D converters, magnetic-card readers as well as by providing the connection for other devices using the Centronics protocol. It seems important to us to emphasize that according to our favourable past experience, these extensions will be easily accomplished because of our reliance on Modula-2 as our standard high-level language for software development.

5. REFERENCES

- [1] "Computer communications and network", "E. Kardelj" University, Faculty of Electrical Engineering, Ljubljana 1988. (in Slovene)
- [2] J. Strom, "Matter of protocols", Systems International, Vol.16, No.12, December 1988.
- [3] J.M. Catalfo and G.I.E. Cencelo, "Lan node for industrial environments", Microprocessors and Microsystems, Vol.12, No.7, September 1988.
- [4] P.B. Hansen, "Operating System Principles", Prentice-Hall, 1973.
- [5] ISO International standard 2111, "Data communication-Basic mode control procedures-Code independent information transfer", ISO 1972.
- [6] ISO International standard 646, "7-bit coded character set for information processing interchange", ISO 1973.

- [7] ISO International standard 2628, "Basic mode control procedures-Complements", ISO 1973.
- [8] ISO International standard 2629, "Basic mode control procedures-Conversational information message transfer", ISO 1973.
- [9] ISO International standard 1745, "Information processing- Basic mode control procedures for data communication systems", ISO 1975.
- [10] "Z8030/Z8530 SCC Serial Communications Controller", Zilog, 1983.
- [11] B.Korousic, "An Industrial Terminals Network in Real-Time" (diploma work), "E.Kardelj" University, Ljubljana 1989 (in Slovene).
- [12] J.A.Stankovic, "Misconceptions About Real-Time Computing", Computer, Vol.21, No.10, October 1989.
- [13] Real Time Associates, "The IN - BED System Handbook", London - Croydon, 1988.
- [14] A.Brodnik, J.Ferbezar, S.Mavric, M.Spegel, V.Vouk, " System for Rapid Prototyping of Embedded Applications Using Modula-2", to be published in Software 1989.

Промышленная сеть низкой стоимости для работы в реальном масштабе времени

В статье рассматривается новая дешевая сеть общего назначения для промышленных применений. Определяются линейный и сетевой уровни семиуровневой модели взаимодействия открытых систем. Метод доступа к сетевому уровню является комбинацией принципов отброса и передачи опознавательного знака. Приводятся результаты анализа пропускной способности протокола. Обсуждаются также два практических примера применения этой сети.

TANIA PRZEMYSŁOWA SIĘĆ KOMPUTEROWA DLA ZASTOSOWAŃ W CZASIE RZECZYWISTYM

W pracy przedstawiono sieć komputerową ogólnego przeznaczenia dla zastosowań przemysłowych. Zdefiniowano warstwy liniową i sieciową siedmio warstwowego modelu połączeń systemów otwartych. Metoda dostępu w protokole warstwy sieciowej jest kombinacją strategii wywoływania i przekazywania znacznika. Zaprezentowano wyniki analizy przepustowości protokołu. W zakończeniu przedyskutowano dwa praktyczne przykłady za - stosowania omawianej sieci.

conformance test,
test system,
OSI,
TEKOS

Uwe BUHRAND *
Jochen HINKELMANN *

THE DESIGN OF THE CONFORMANCE TEST SYSTEM "TEKOS"

To ensure the correct work of communication software, beside a normal software test there was found a need for testing the conformance of an implementation to its specification. In our institute the test system TEKOS for testing upper layer protocols of the OSI reference model was developed. The main characteristic of TEKOS is its protocol independence. In this paper the architecture and the implementation of TEKOS are described.

1. INTRODUCTION

To ensure the correct work of communication software, beside a normal software test there was found a need for testing the conformance of an implementation to its specification [2].

In our institute the test system TEKOS [3] for testing upper layer protocols of the OSI reference model [1] was developed.

The implementation under test (IUT) which has to be studied by testing is a part of a real open system. The IUT consists of one or more OSI layers and is considered as a "black box". For testing the IUT, points of control and observation (PCO) are used. One PCO is located at the upper service boundary of the IUT and another below the IUT.

The PCOs are used by two test facilities, the upper tester (UT) and the lower tester (LT). The rules for the cooperation between LT and UT have to be described in test coordination procedures.

The main goal for the design of the test system was to obtain the independence of the test system from the concrete IUT.

* Academy of Sciences of the GDR Institute of Informatics and Com
Computing Technique

THE ARCHITECTURE OF THE TEST SYSTEM "TEKOS"

For our test system we chose the external distributed test method. This abstract method is characterized in the following way (Fig. 1):

- The UT is located together with the IUT and the underlying transmission service (TS) at one site, the system under test (SUT).
- The LT and the TS form the test system at another site.
- The connection between the LT and the lower IUT interface is realized by the TS through a connection under test (CUT).
- The UT has access to the IUT service.

The external distributed test method allows the implementation and operation of the test system in a separate test center. The IUT may work in its normal environment.

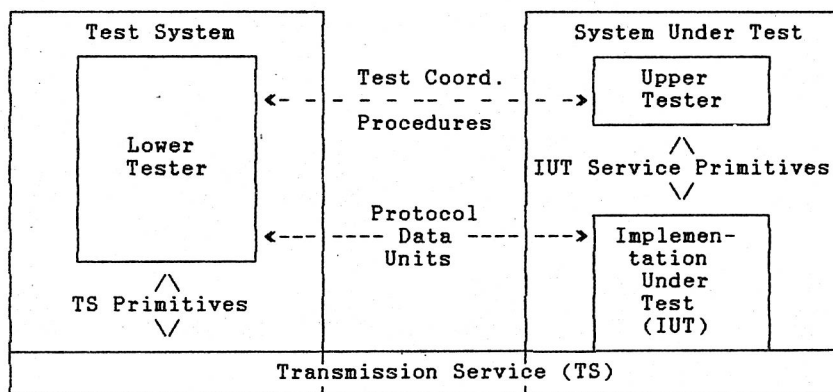


Fig.1 The External Distributed Test Method

For the realization of TEKOS, some additional arrangements and modifications were made (Fig. 2):

- The main part of the UT is located at the test system too. Only a small part of the UT without own intelligence remains at the SUT. This so called **test responder (TR)** is driven by the UT part in the test system through a second connection between the test system and the SUT, the **test connection (TC)**. The TC may be routed through the IUT ("ferry concept", [5]) or not ("astride technique", [4]). If it is possible the first variant is avoided because it assumes the correct operation of several IUT functions and may influence the CUT. The second variant demands the direct access to the TS beside the IUT.

The arrangement of UT and LT within one system simplifies the coordination between them; there is no need for a protocol between UT and LT.

The statements for the control and observation of the IUT service are transmitted via the TC in the form of abstract service primitives (ASP). Their mapping onto concrete IUT directives and vice versa is performed by the TR.

By exchanging ASPs, the interactions of the UT with the IUT may be described independent from the concrete service interface of the IUT.

- The protocol data units (PDU) exchanged between LT and IUT are built and analyzed by the LT with an encoder and decoder (E/D) respectively.

In comparison with a reference implementation of the appropriate protocol an E/D allows more flexibility. Any corrupted PDUs may be sent and received.

- For the work with ASPs in the UT within the test system the E/D is used too.

This simplifies the operation of the UT.

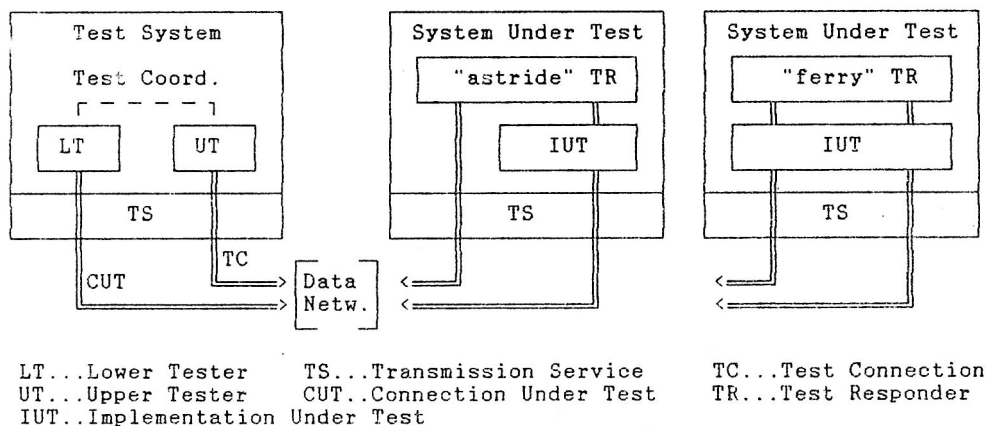


Fig.2 Architecture of the Test System and the System Under Test in "astride" and "ferry" Version

3. DESCRIPTION OF THE CONCRETE STRUCTURE OF "TEKOS"

The test system TEKOS consists of several components. These components were defined with the intention to minimize

- the dependence from the concrete layer under consideration,

- the dependence from the concrete TS and
- the complexity of the interactions between the components.

The functions of the UT and the LT in TEKOS are divided into an invariant part, the test handler (TH), and variant parts, the interface handlers (IH) (Fig. 3).

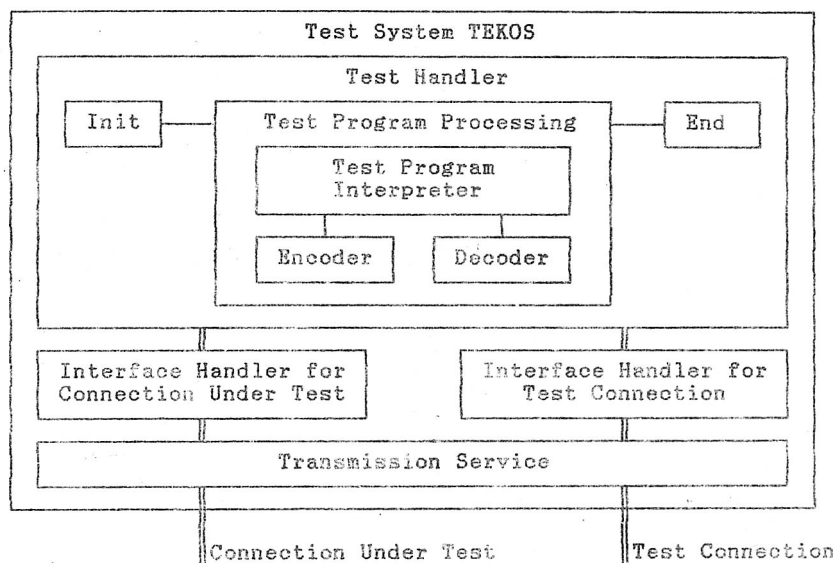


Fig.3 Detailed Structure of the Test System TEKOS

The main part of the TH is the test program processing component. A test program is a parameterized executable test suite. The test program processing component is subdivided into several modules.

The test program interpreter interprets a test program, which consists of a message type description part and an action part. The most important statements within the action part concern the message transfer with the IUT via the IHs.

When a PDU or ASP is to be sent the interpreter builds up a parameter list first with values given in the action part. Then the encoder is called which constructs the final PDU or ASP on the base of the appropriate type description and the parameter list.

On receiving a PDU or ASP the interpreter calls the decoder which checks the plausibility of the message and extracts the parameters with the help of the type description. The parameter list built up by the decoder may be used for further analysis by the interpreter.

Encoder and decoder are not bound by rules of a protocol. The price paid

for the flexibility of the E/D concept is that the test must be specified in minute detail and becomes quite long.

The IHs map the messages into real service data units for the underlying TS.

The IH for the CUT sends PDUs to the IUT and receives PDUs from the IUT. This component provides a generalized service independent from the specific implementation of the TS.

The IH for the TC sends ASPs to the TR and receives ASPs from the TR using a TC protocol. It provides a service interface for the exchange of ASPs which is completely independent from the layer that is used for the transmission.

The TR simulates the user of the service provided by the IUT. This component has to be portable and small in order to ensure the implementation at the SUT with little effort and also for systems with limited resources.

4. IMPLEMENTATION AND RESULTS

The described test system TEKOS is implemented in MODULA-2 on a VAX-compatible minicomputer. It uses a subprocess supervisor developed in our institute. This supervisor allows the quasi-parallel operation of the TH and the IHs which are implemented in the form of subprocesses. The chosen architecture of TEKOS allows to build up a flexible system for testing different implementations of several layers. The first tested IUT was an implementation of the session layer.

REFERENCES

- [1] Information Processing Systems - Open Systems Interconnection - Basic Reference Model
ISO / IS 7498, Oct. 1984
- [2] OSI Conformance Testing Methodology and Framework
ISO / DP 9646, July 1987
- [3] N. N.
Konformitätsbewertung von Rechnernetzkomponenten
Inst. of Inform. and Comp. Techn., Berlin, 1989 (Internal Report)
- [4] Rafiq,D., Castanet,R., Chraibi,C.
Towards an Environment for Testing OSI Protocols
Protocol Specification, Testing and Verification VI
North-Holland, Amsterdam, 1986
- [5] Zeng,H.X., Li,D., Du,X.F.
New Advances in Ferry Testing Approaches
Computer Networks and ISDN-Systems, 14(1987)

protocol conversion,
heterogenous network,
temporal logic

Otakar BUZEK*

ACHIEVING INTEROPERABILITY VIA PROTOCOL CONVERSION

Consider the problem of achieving interoperability between two processes in different protocol systems. Protocol entities communicate by exchanging messages; we discuss the possibility of construction of a protocol converter based on common image protocol system. The approach - to combine the projection method and the temporal logic formalism - is based on recent papers [4], [6].

1. INTRODUCTION

Due to the steady proliferation of heterogeneous networking as a result of many factors it becomes increasingly difficult to solve the problem of the information exchange. The problem of achieving interoperability between entities in different network architectures is a permanent fact of life even if standard protocol architecture, say, Open System Interconnection [1], has been presented. It is already too late to get everyone to adhere to the same standard. There is installed a base of many thousands SNA networks, several thousands DECnet networks and so the role of OSI is much more promising as an intermediary protocol for conversion between two existing ones than it is a worldwide standard architecture towards which all existing architectures should converge.

In recent years in our institute there has been an intelligent terminal network developed and installed. To date due to the possibility to incorporate some functionally and economically

* General Computing Center, Czechoslovak Academy of Sciences, Praha

interesting products it is worth the expense of providing adequate conversion.

Up to now there have been some "ad hoc" protocol conversions attempted [2],[3] and some formal models of protocol converters have been proposed but still there is no general theory for the solution of the protocol conversion problem.

II. CONVERSION TO ACHIEVE INTEROPERABILITY

In our research in this field we have tried to use temporal logic and the projection method for the construction of a general protocol converter (related works [4],[5],[6]).

The projection approach is in use in the protocol verification and the protocol modelling and up to now it is quite well-sophisticated. This analysis avoids the characterisation of the reachability graph and the set of the reachable states of the protocol system. Instead of this brute-force state exploration an image protocol system for functions that interest us is constructed from a given protocol system. The main goal is to find an image protocol system common for both given protocol systems.

At any time the protocol system, say Π , is completely specified by the following sets [6]:

$$S_i, M_{ik}, T_i, E_k, g_0 \quad \text{for } i = 1, 2, \dots, I \\ k = 1, 2, \dots, K,$$

where S_i is the set of states of the entity P_i ,

M_{ik} is the set of messages that the entity P_i can send into the channel C_k ,

T_i denotes the set of events specified for P_i ,

E_k denotes the set of channel events specified for the channel C_k and

g_0 denotes the initial global state of the protocol system Π .

Now via aggregation of entity events, states and messages we can obtain an image computation tree which characterizes an image protocol system. For this image protocol system to be a common image protocol system for systems Π_1 and, say Π_2 it is necessary to make the projection of Π_1 with respect to characteristic behaviour of Π_2 and vice versa. A Protocol converter based on described image

protocol system can simply provide a mapping function (A message sent by an entity P_1 of the protocol system Π_1 e.g. $m_{ik} \in M_{ik}$ goes through the channel C_k and is transformed by converter into a message $n_{ik} \in N_{ik}$ with the image $n'_{ik} = m'_{ik}$ for delivery to the channel R_k and the entity Q_1 of the protocol system Π_2 . Similarly, messages in N_{ik} are mapped into those in M_{ik} , $i = 1, 2, \dots, I$, $k = 1, 2, \dots, K$.) or can be implemented as a finite state converter which provides conversion of sequences of messages.

The formalism of temporal logic is used for specifying structures of states and especially for investigation of assertions of liveness and safety properties.

There are some useful properties of an image protocol constructed this way [4]. First, any safety property that holds for an image protocol system must also hold for the original protocol system. Second, if any image protocol has a sufficient resolution so that its events satisfy a well-formed property, then it is faithful. (See [4], [5] for an excellent treatment of this subject).

A protocol converter between protocol systems Π_1 and Π_2 "speaking" common image protocol with the highest resolution can be constructed on these principles.

III. CONCLUSION

A protocol converter to achieve interoperability between two protocol systems is implemented in the physical path between the protocol systems. It is worthwhile noting that this protocol conversion is quite different from protocol complementing [2].

REFERENCES

- [1] Information Processing - Open System Interconnection - Basic Reference Model, International Standard 7498, first edition 1984-10-15
- [2] P. Green, "Protocol conversion," IEEE Trans. Commun., vol. COM-34, no. 3, pp. 257-268, Mar. 1986

[3] I. Groenback, "Conversion between TCP and ISO transport protocols as a method of achieving interoperability between data communications systems," IEEE J. Select. Areas Commun., vol. SAC-4, no. 2, pp. 288-296, Mar. 1986

[4] S. S. Lam, "Protocol conversion," IEEE Trans. on Software Engineering, vol. 14, no. 3, pp. 353-362, Mar. 1988

[5] A. Pnueli, "Applications of temporal logic to the specification and verification of reactive systems: A survey of current trends, " in Current Trends in Concurrency: Overview and Tutorials (Lecture Notes in Computer Science, vol. 224), J. W. de Bakker et al., Eds. New York: Springer-Verlag, 1986, pp. 510-584

[6] S. S. Lam and A. U. Shankar, "Protocol verification via projections," IEEE Trans. Software Eng., vol. SE-10, no. 4, pp. 325-342, July 1984

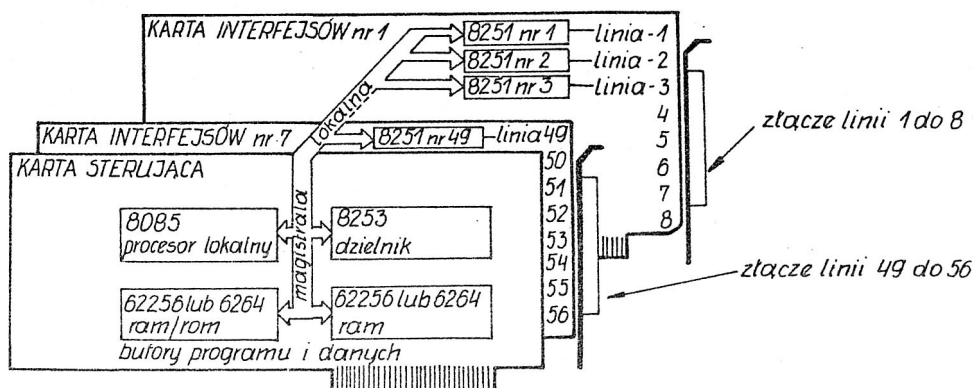
JG CABANSKI *
M LEWANDOWSKI *
St MARACH *
Z OCZKI *

sieci lokalne
sprzęt PC

A S T E R Autonomiczny STERownik TRANSMISJI SZEREGOWYCH DLA PC XT/AT

ASTER, autonomicznie steruje blokową transmisją danych komputera PC z/do maksymalnie 56 urządzeń/komputerów poprzez interfejsy napięciowe lub prądowe bez/z optoizolacją. Transmitowane bloki danych z PC do urządzeń lub odwrotnie - opcjonalnie przetworzone przez emulatory protokołów sieci MasterNET, udostępniane są wzajemnie we wspólnej pamięci RAM (ang. dual port memory). Zakończenie transmisji blokowych sygnalizowane jest poprzez system przerwań PC. Przeznaczenie: sterowanie, rejestracja, bazy danych, banki, handel, dydaktyka.

Komputery PC są wyposażone zwykle w jeden lub dwa interfejsy szeregowo RS232C, jako urządzenia COM1 i COM2 w systemie DOS. Ilość ta, zadowala wielu użytkowników komputera PC. Problem powstaje kiedy zachodzi konieczność zastosowania kilkunastu lub kilkudziesięciu interfejsów szeregowych. Wprawdzie dostępne są dodatkowe karty z 4 lub 8 interfejsami szeregowymi, jednakże użytkownik takiej karty musi zapewnić programową obsługę sprzętu dla nadania/odebrania każdego pojedynczego znaku, indywidualnie dla każdego interfejsu szeregowego karty. Czas zaangażowania procesora PC obsługą wielu interfejsów szeregowych już przy średnich szybkościach transmisji poważnie ogranicza możliwości komputera PC. Narzuca się konieczność uwolnienia procesora komputera PC od czasochłonnych obowiązków obsługi wielu interfejsów szeregowych. Oczekiwania te, realizuje prezentowany poniżej sterownik ASTER.



Rys.1 Struktura logiczna i fizyczna sterownika ASTER.

Sterownik ASTER ma budowę modułową i składa się z:

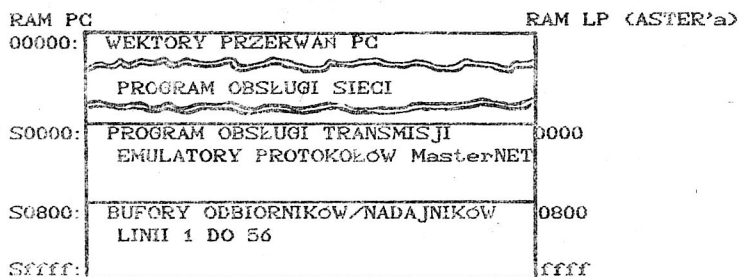
- jednej karty sterującej,
 - jednej do siedmiu kart interfejsowych po 8 interfejsów każda.
- Stwarza to możliwość dołączenia do komputera PC do 56 urządzeń wyposażonych w interfejs szeregowy. Możliwa jest również instalacja w komputerze PC drugiego sterownika transmisji szeregowych ASTER.

K a r t a s t e r u j ą c a zawiera:

- procesor lokalny Intel 8085 dla autonomicznej obsługi interfejsów,
- programowany dzielnik częstotliwości Intel 8253 dla programowej konfiguracji bazowych szybkości transmisji X,Y i czasu względnego,
- jedną lub dwie pamięci typu 62256 lub 6264 jako RAM o dwustronnym dostępie w przestrzeni adresowej komputera PC dla buforowania programu, oraz danych (bufor programu i bufor danych).

Pojemność pamięci RAM jak i jej adres w przestrzeni adresowej PC dostosowywane są do wymagań użytkownika. Ponadto RAM bufora programu może być zastąpiona pamięcią ROM typu 27256 lub 27128 lub 2764.

Karta sterująca współpracuje z kartami interfejsowymi przez własną magistralę lokalną.



Rys. 2 Przestrzeń adresowa PC i ASTER'a

K a r t y i n t e r f e j s o w e zawierają po 8 interfejsów szeregowych napięciowych lub prądowych bez/z optoizolacją. Każdy z interfejsów karty może transmitować dane z jedną z dwóch bazowych X,Y lub z trzech połówkowych Y/2, X/4, X/8 szybkości transmisji. Bazowe szybkości ustalane są programowo, połówkowe wydzielane sprzętowo. Przyporządkowanie interfejsów do poszczególnych szybkości jest realizowane sprzętowo.

Obsługą interfejsów na poziomie sprzętu i protokołu transmisji zajmuje się sterownik. Użytkownik PC widzi każdą linię jako bufor odbiornika i nadajnika we wspólnej pamięci RAM. Wysłanie komunikatu w linię ogranicza się do wpisania jego treści do odpowiedniego bufora.

Autonomiczny sterownik transmisji szeregowych ASTER dzięki swym walorom koncepcyjnym i ekonomicznym znajduje zastosowanie w przemysłowych systemach sterowania (cukrownie), rejestracji (zakłady przemysłu mięsnego) złożonych systemach pomiarowych, systemach dydaktycznych, sieciach lokalnych obsługi klienta (banki, handel, dworce, hurtownie, magazyny).

ASTER - Autonomous serial transmission controller for PC XT/AT

ASTER - controls autonomously block data transmission of the PC from/to maximum of 56 devices/computers via voltage/current loop interfaces with/no optocouplers. Data records transmitted from the PC to devices or in the opposite direction, optionally transformed by MasterNET protocol emulators, are interchanged via dual port memory. The end of the data transmission is signalled by the interrupt system of the PC. Application: control, data acquisition, data bases, banks, teaching.

ASTER - Автономический контроллер последовательной передачи для PC XT/AT.

ASTER автономически управляет блоковой передачей данных компьютера PC с/до максимально 56 устройств /компьютеров через интерфейсы типа напряжение или ток с или без оптической изоляцией. Блоки данных передаваемые из PC в устройства или обратно обработанные эмуляторами протоколов сети MasterNET, можно взаимно использовать в общедоступной памяти типа ОЗУ. Окончание блоковой передачи сигнализируется при помощи системы прерываний ЭВМ. Сеть предназначена для использования в системах управления, регистрации данных, баз данных, банках, торговли, дидактике.

JG CABANSKI*
M LEWANDOWSKI*
St MARACH*
Z OCZKI*

sieci lokalne

M a s t e r N E T

SIEĆ LOKALNA DLA IBM PC XT/AT

MasterNET - sieć lokalna o topografii gwiazdистой z centralnym węzłem w komputerze PC XT/AT łącząca do 56 sterowników, rejestratorów, terminali, monitorów współpracujących z otoczeniem poprzez interfejs szeregowy według różnorodnych protokołów transmisji danych na bazie autonomicznego sterownika transmisji szeregowych ASTER.

MasterNET jest siecią otwartą - w pewnym sensie jest tylko strukturą sieci wypełnianą przez określoną aplikację. Jedynymi niezmiennikami tej sieci jest jej topografia, uwarunkowania techniczne sterownika transmisji szeregowych ASTER oraz dwuczłonowość oprogramowania - a mianowicie:

- programu obsługi sieci (POS) wykonywanego przez procesor PC,
- programu obsługi transmisji (POT) wykonywanego przez procesor lokalny LP (ang. local processor).

Kształt i funkcje programu POS zależą od konkretnej aplikacji sieci. Program POT ma natomiast sprecyzowane zadanie: obsłużyć transmisje blokowe zlecane przez program POS oraz ewentualne transmisje międzyliniowe. Sposób zlecenia transmisji może być różnorodny. Problem zlecenia transmisji blokowych sprowadza się do podania kierunku transmisji, adresu i długości bloku oraz protokołu i parametrów fizycznych transmisji. Najbardziej przejrzystą metodą uzyskuje się przy wspólnie ograniczonej długości bloku oraz stałych adresach buforów odbiorników i nadajników. Koncepcję tę, już zastosowaną, przedstawiają rysunki 1 do 4.

P r o g r a m (podprogramy) o b s ł u g i s i e c i :

- ładuje do bufora programu karty sterującej sterownika program POT,
- definiuje/modyfikuje parametry sieci,
- inicjuje zerowanie sprzętowe sterownika,

* CBR Elektronicznych Układów Specjalizowanych, TORUŃ, tel. 33045-47

- aktywizuje procesor sterownika,
- zleca sterownikowi realizację transmisji szeregowych dla poszczególnych interfejsów wg wcześniej ustalonych parametrów (szybkość, protokół),
- obsługuje przerwania od sterownika,
- zgłasza przerwania do sterownika.

Program obsługi transmisji

po otrzymaniu sterowania - aktywizacji procesora sterownika:

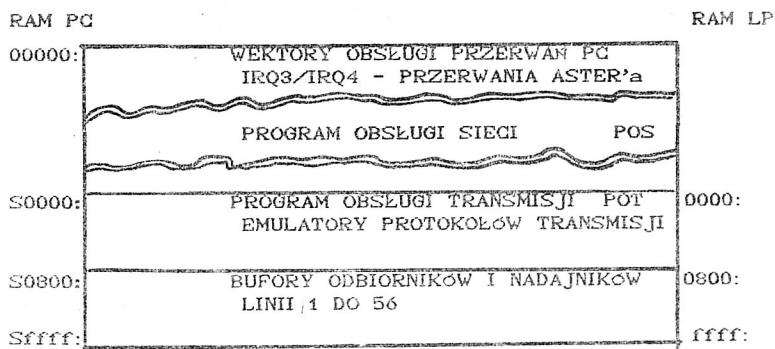
- przygotowuje interfejsy szeregowo do pracy w sieci,
- nadaje w linie/odbiera z linii komunikaty z/do buforów zawartych we wspólnej pamięci RAM wg zleceń programu obsługi sieci, zgodnie z wybranym dla danej linii protokołem transmisji,
- informuje program obsługi sieci o zaistniałych zdarzeniach i nieprawidłowościach.

Programy: obsługi sieci i obsługi transmisji komunikują się poprzez bufory komunikatów (rys. 2). Dodatkową możliwość komunikacji stanowi system przerwań. Bufor komunikatów linii (odbiornika, nadajnika) ma strukturę (rys. 3) pozwalającą elastycznie definiować zlecenia dla programu obsługi transmisji. Rys. 4 prezentuje przykłady zleceń realizowanych przez sterownik na poziomie sprzętu.

Program obsługi sieci korzysta ze zbioru podprogramów sieci MasterNET dla zlecenia transmisji. Możliwe są zlecenia bezpośrednio przez RAM.

Program obsługi transmisji składa się z:

- bloku obsługi przerwań,
- emulatorów protokołów (IBM2740, IBM3270, BSC, BPPM, ...),
- wektora konfiguracji sieci definiującego który z emulatorów obsługuje dany interfejs szeregowy - linię.



S= 8,9,...,E nr segmentu pamięci RAM dwustronnie dostępnej

Rys. 1 Pamięci procesora PC i procesora LP (local processor)

Linia nr sxx00:	00	01	02	...	FF	00	01	02	...	FF
Linia 01 s0800:	lb	hb	buf. odb. L01			lb	hb	buf. nad. L01		
Linia 02 s0a00:	lb	hb	buf. odb. L02			lb	hb	buf. nad. L02		
	...	...	...			...	...	...		
Linia n sxx00:	lb	hb	buf. odb. Ln			lb	hb	buf. nad. Ln		
	...	...	...			...	...	...		
Linia 56 s7e00:	lb	hb	buf. odb. L56			lb	hb	buf. nad. L56		

Rys. 2 Bufory komunikatów

Linia n sxx00:	lb	hb	buf. odbiornika Ln			lb	hb	buf. nadajnika Ln		
----------------	----	----	--------------------	--	--	----	----	-------------------	--	--

hb - starszy bajt adresu tekstu nadawanego/odbieranego
(wybór bufora nadajnika/odbiornika linii od 8 do 7f lub status),

lb - 00 = koniec transmisji, wolny nadajnik/odbiornik linii,
01 = zerowanie i programowanie linii,
02...ff = młodszy bajt adresu tekstu nadawanego/odbieranego
w wyżej wybranym buforze hb
(tekst nadawany/odbierany mieści się w prawej części bufora nadajnika/odbiornika, a jego długość równa się 256-lb < 255)

xx - starszy bajt adresu bufora komunikatów równy 8+2*n.

Rys. 3 Bufor komunikatów linii

	00	01	02	FF	00	01	02	FF
s0800:L01	00	00		ED	09			M a s t e r N E T
s0a00:L02	00	00		EF	0B			Głowackiego 48/30
s0c00:L03	00	00		01	09			33045-33047w311
s0e00:L04	00	00		FE	09			
s1000:L05	FB	10		REPLY	ED	09		
s1200:L06	FB	12		TORUN	F6	13		TELEPHON ?
s1400:L07	FE	17		ED	09			

W linii 1 nadaj komunikat: M a s t e r N E T z bufora linii 1 (09).

W linii 2 nadaj komunikat: Głowackiego 48/30 z własnego bufora nr 0B.

Linia 3 zakończyła poprawnie nadawanie z bufora 09 linii 1.

Linia 4 nadaje komunikat z bufora 09 tj. z bufora nadajnika linii 1.

W linii 5 nadaj komunikat z bufora 09 oraz odbierz pięć znaków odpowiedzi do własnego bufora odbiornika.

W linii 6 nadaj zapytanie: TELEPHON ? oraz odbierz pięcioletnią odpowiedź np. TORUN do własnego bufora odbiornika.

W linii 7 nadaj komunikat z bufora nadajnika linii 1

oraz odbierz odpowiedź dwuznakową: OK do bufora nadajnika linii 8.

Rys. 4 Przykład wypełnienia buforów komunikatów

MasterNET - local network for IBM PC XT/AT

MasterNET - is a local net of star topography with the IBM PC/XT/AT in its central node built on the basis of ASTER - the autonomous serial transmission controller. It can connect up to 56 controllers, recorders, terminals, and CRT monitors cooperating with environment via serial interface according to various data transmission protocols.

MasterNET - локальная сеть ЭВМ типа IBM ПЦ XT/AT

MasterNET - локальная сеть ЭВМ типа "звезда" с центральным узлом типа IBM XT/AT соединяющая до 56 контроллеров, регистраторов, терминалов, дисплеев сопряженных при помощи последовательных интерфейсов с различными протоколами передачи данных основанных на автономном контроллере последовательной передачи ASTER.

Wojciech CELLARY*

Distributed Databases

Some Thoughts on Distributed Database Research

Abstract. The basic technology for creating distributed database systems, namely computer networks and database systems, is available now. However, distributed database systems, in comparison with their centralized counterparts, pose new design and management problems. This paper presents the rationale for and against distributed databases and, then, it presents some thoughts on main research topics in the field of distributed database systems design and management.

A Distributed Database System (DDBS) may be defined as an integrated database system composed of autonomous local databases interconnected by a computer network and communicating with one another to perform a common user task.

Research in the area of DDBSs has been experiencing rapid growth in recent years, and at present several commercial DDBSs are already available: Encompass from Tandem, Ingres-Star from Relational Tech., ORACLE from Oracle Corp., Teradata and other.

There are several organizational and technical reasons for developing DDSSs [Bernstein 87, Ceri 84, Cellary 88, Gray 86].

The main organizational reasons for DDBSs are as follows:

- * organizational autonomy : many organizations are decentralized : banks, airlines, travel agencies, large enterprises and corporations, etc. For such decentralized organizations distributed computer systems - in particular distributed database systems - are more adequate than centralized ones since they better reflect their decentralized structure. Managers at different levels of the organization structure need administrative control over facilities critical to their effectiveness particularly of their computers and their databases. Moreover, DDBSs are flexible in their capacity to grow and modify their functions and more maintainable in the ability of each of their elements to change independently of the others;
- * integration of pre-existing database systems: a distributed database system is a natural solution when several databases already exist in an organization and the necessity to provide common global applications arises.

There are several technical and economical reasons for developing distributed database systems. First, it is sometimes not feasible to

* Technical University of Poznań, 60-965 Poznań

build and maintain a centralized database system with the required capacity. Second, distributed database systems can improve most of the important indices characterizing the quality of a computer system such as: availability, reliability, performance, security, flexibility and modularity.

Typical issues are as follows:

- * increased availability and reliability of the system due to the physical distribution and replication of computer resources such as data, computing power, etc. A crash of a single site does not necessarily affect the other sites. In addition, the remaining sites may be able to take over of the functions of the crashed site,
- * better system performance as a consequence of the increased level of parallel processing also obtained by putting the system resources closer to data sources and users,
- * increased capacity and flexibility of the system resulting from the open and modular structure of the database system which allows growth and smoother change of functions and capacity,
- * increased data security since managers have physical control over the databases that store their data.

The basic technology for creating distributed database systems, namely computer networks and database systems, is available now. However, distributed database systems, in comparison with their centralized counterparts, pose new design and management problems.

A DDBS is managed by a Distributed Database Management System (DDBMS) whose main task is to give the users a "transparent" view of the distributed structure of the database, i.e. an illusion of having a monolithic and centralized database at their disposal. Distribution transparency, i.e. location and replication transparency, implies that the conceptual and external-level problems (using the ANSI/SPARC terminology) of distributed databases do not essentially differ from similar issues in centralized database systems. On the other hand, the internal-level problems concerning physical distributed database design and DDBS management are specific and qualitatively new.

Let us consider these new problems in more detail. First, we discuss the problem of distributed database design. It is clear that it is more difficult to design a distributed database than a centralized one since many technical and organizational issues which are crucial in the design of a single-site database become more difficult in a multiple-site system. The term "distributed database design" has a very broad and unprecise meaning. We will concentrate on those problems which are peculiar to distributed database, namely data fragmentation, allocation and replication.

Data (e.g. relations) may be subdivided into horizontal, vertical, and mixed fragments and then partitioned among local databases of the distributed system based on some predicates.

The following are simple fragmentation and partitioning predicates.

Global relation: Student (Number, Name, Dept)

Fragmentation predicate:

relation Student₁ = Student for which Dept = 'Electrical Eng';
 relation Student₂ = Student for which Dept = 'Computer Science';
 relation Student₃ = Student for which Dept = 'Robotics';
 (we assume that 'Electrical Eng', 'Computer Science' and 'Robotics' are only values for Dept).

Partitioning predicate:

relation Student₁ stored at local database LB₁ at site 1
 relation Student₂ stored at local database LB₂ at site 2
 relation Student₃ stored at local database LB₃ at site 3

Relations are fragmented and partitioned for two main reasons:

- * autonomy: allowing local control and management of each fragment of a relation,
- * locality: placing relations closer to their consumers thus improving local availability and performance.

Fragments of relations may be replicated at several local databases. Replication of relation fragments is used for two reasons: to increase availability of these fragments and to improve performance by eliminating longhaul message delays. However, replication of fragments poses several new problems if updates must be performed on all the replicas. We will return to this problem later when we discuss the issue of concurrency control.

In spite of limited experience in the design of distributed databases systems, the problem of data fragmentation and allocation is intensively studied as an advanced research area [Ceri 84, Coan 86, Cornell 87, Cornell 88, Wah 85].

The main issues in DDBS management can be classified in load balancing, query optimization, concurrency control and reliability. Solutions to these problems in a centralized environment, if they exist, are inappropriate for a distributed environment because of the differences in the internal level structure of the databases. Their effective solution conditions the possibility of taking full advantages of DDBS applications. The load balancing problem has been intensively studied in the field of distributed systems during the past few years mostly from the theoretical viewpoint. Load balancing is a process of effective sharing computational resources by transparently distributing the system workload, or, in other words, it is a process of dynamic

scheduling of user tasks to optimize the utilization of distributed system resources. Research concerning load balancing techniques was concentrated mainly on communication network, distributed file systems and distributed operating systems. A number of approaches and solutions to this problem have been presented in the literature [Casavants 86, Tantawi 85, Wang 85].

However, very little has been done in the field of DDBSs [Reuter 86]. DDBSs currently in use provide only very rudimentary means for load balancing. It is intuitively clear that further improvement of distributed database system performance may be obtained by effective balancing of the workload on different sites of the DDBS.

The next important issue in DDBS management is the query optimization problem. This problem has been an area of active research ever since the beginning of the development of relational database systems. It has received considerable attention in DDBS as well. A good survey on query optimization and other related issues can be found in the survey articles by Jarke and Koch [Jarke 84], Yu et al [Yu 84] and the book by Kim, Reiner and Batory [Kim 86]. If a database is distributed the processing cost of a query is determined by two factors: the cost of data transfer (communication costs) and local processing costs. The trade_off between communication costs and local processing costs generally depends on the transmission speed of the computer network.

Initially, the query optimization problem was considered under the assumption that communication costs dominate the costs of local processing and that the later may be ignored [Jarke 84, Kim 86, Yu 84].

The results of this research are applicable to DDBS implemented on longhaul point-to-point networks. However, local area networks exhibit quite different performance behaviors and therefore for DDBS implemented on local networks local processing costs cannot be ignored. Several heuristic algorithms for query optimization for DDBS on local networks have been proposed [Chen 87, Hevner 85, Wah 85, Yu 87].

As was proved by Chu and Hurley [Chu 82] the local preprocessing of unary relational operations (i.e. selection, projection) minimizes both communication and local processing costs. Therefore, many researchers have concentrated on the optimal implementation and scheduling of join operations in distributed systems. The main tool used to optimize the execution of a join operation is a semijoin operation [Bernstein 81, Ceri 84, Egyhazy 88, Jarke 84, Kim 84, Yu 84]. Based on the semijoin technique several query optimization procedures were developed [Ceri 84, Egyhazy 88, Chen 87].

Several aspects of the query optimization problem change when one

studies this problem in systems that are geared towards some of the newest database application domains such as artificial intelligence, expert database systems and deductive database systems. The most important changes are listed below:

- the unit of optimization changes from single query to a set of queries [Chakravarthy 86B, Sellis 88],
- queries may become recursive,
- queries may be nested [Ganski 87, Lohman 84],
- the number of relations expected to participate in a query increases significantly [Ioannidis 87],
- the knowledge about relations, domains of their instances and various constraints associated with them may be incorporated in the query optimization procedures [Chakravarthy 86A, Ioannidis 87, Shenoy 87].

Each of the above points represents an interesting research topic.

The next fundamental problem facing the designers of DDBSs is that of the correct control of concurrent access to the distributed databases by many users - i.e. the concurrency control problem.

The general aim of a concurrency control algorithm is to ensure consistency of the database and the correct completion of each transaction initiated in the system. An obvious additional requirement is to minimize overhead and transaction response time, and to maximize DDBS throughput.

Three successive phases can be distinguished in the study of concurrency control in DDBSs. Initially, there was an attempt to adapt concurrency control algorithms designed for multiaccess but centralized database systems. This attempt was not successful for one main reason: in DDBS no computer site will in general hold full information on the global state of the whole system, and, hence, all control decisions taken at a site of DDBS have to be made on the basis of incomplete and not entirely up-to-date information on the activities of the remaining sites. This fact must be taken into account in every concurrency control method.

In a more recent past three basic methods were designed in relation to the syntactic model of concurrency control, i.e. the model in which no semantic information on transactions and data is assumed. These are locking, timestamp ordering and validation (or certification). In the next phase of research on concurrency control problems a multiversion data model was assumed. Multiversion DDBSs are attractive to DDBS designers for several reasons. They allow a higher degree of concurrency, they can be combined with reliability mechanism in a natural way, and they can be easily designed so that no queries are delayed or rejected [Bernstein 87, Cellary 88, Herlihy 87]. A practical

designers for several reasons. They allow a higher degree of concurrency, they can be combined with reliability mechanism in a natural way, and they can be easily designed so that no queries are delayed or rejected [Bernstein 87, Cellary 88, Herlihy 87]. A practical difficulty with all multiversion concurrency control algorithms is the apparent need to keep a potentially unlimited volume of data. Therefore, most current multiversion database systems (e.g. ORACLE) are in fact K-version database systems, where K denotes the maximal number of versions of one data item. The development of concurrency control theory and algorithms for K-version distributed database systems is a new attractive topic of research in the field of concurrency control [Morzy 89].

In recent years three important open issues in transaction management in DDBSs have been identified. The first problem intensively studied recently, is the problem of managing replicated DDBSs in the face of network partitioning due to site or communication link failure. This problem is closely related to the problem of data availability in DDBSs. Although several techniques and algorithms for managing replicated DDBSs have been recently proposed [Bernstein 87, Davidson 85, El Abbadi 86, Garcia-Molina 88, Herlihy 86, Jajodia 87], additional work remains to be done to obtain satisfactory results in this field. The second problem concerns the concurrency control theory for nested transactions [Beerl 86, Haerder 87B, Lynch 86, Weikum 86]. The third one concerns the semantic model of concurrency control [Cellary 88, Herlihy 86, Herlihy 87, Schwarz 84]. The semantic concurrency control model tries to exploit the knowledge about the data (e.g. physical structure of the database, consistency constraints, etc) and transactions to improve the performance of a DDBS.

Some additional interesting problems for future investigation are:

- the management of hot-spots [Gawlick 85, O'Neil 86],
- hardware implementation of synchronization mechanisms [Robinson 85],
- the design of the lock managers so they do not become bottlenecks,
- the synchronization mechanisms in large heterogeneous distributed database systems [Elmagarmid 88, Pu 88].

Finally, we briefly discuss the reliability issues that arise in DDBSs. The problem of reliability is how to process transactions in a fault-tolerant manner. As the dependence on computer services grows, the design of fault-tolerant, highly-available distributed systems has become increasingly important over the last few years. There are two main related aspects of reliability: correctness and availability. This means that it is important not only that a DDBS behaves correctly, but also that data are available when necessary.

In the general reliability problem the following specific subproblems can be distinguished:

- the atomic commitment of transactions and termination protocols [Ceri 84, Bernstein 87],
- the database recovery algorithms, particularly for nested transactions [Bernstein 87, Haerder 87A],
- the logging techniques for distributed transaction processing [Daniels 87],
- network partition problem [Davidson 85, Coan 86],
- the problem of masking system crash in application database programs [Freitag 87].

Bibliography

- [Beeri 86] Beeri, C., Bernstein, P.A., Goodman, N. A Model for Concurrency in Nested Transaction Systems, Tech. Rep Wang Inst. TR-86-03, 1986
- [Bernstein 81] Bernstein, P.A., Chin, D., Using Semijoins to Solve Relational Queries, Journal of ACM, Vol. 28, No 1, 1981, pp. 25-40
- [Bernstein 87] Bernstein, P.A., Goodman, N., Hadzilacos, V., Concurrency Control and Recovery in Database Systems, Addison-Wesley Publ. Co. 1987
- [Casavant 86] Casavant, T.L., Kuhl, J.G., A Formal Model of Distributed Decision Making and Its Application to Distributed Load Balancing, Proc. 6th Int. Conf. on Distributed Computing Systems, 1986, pp. 232-239
- [Ceri 84] Ceri S., Pellasatti G., Distributed Databases: Principles and Systems, McGraw-Hill Book Company, 1984
- [Cellary 88] Cellary, W., Gelenbe, E., Morzy, T., Concurrency Control in Distributed Database Systems, North-Holland Publ. Co., 1988
- [Chakravorthy 86A] Chakravorthy, U.S., Fishman, D.H., Minker, J., Semantic Query Optimization in Expert Systems and Database Systems, Proc. 1st Int. Workshop on Expert Systems, 1986, pp. 659-674
- [Chakravorthy 86B] Chakravorthy, U.S., Minker, J., Multiple Query Processing in Deductive Databases Using Query Graphs, Proc. 12th Int. Conf. on Very Large Data Bases, 1986, pp. 384-393
- [Chen 87] Chen, J.S.J., Li, V.O.K., Optimizing Joins in Fragmented Database Systems on a Broadcast Computer Network, Proc. 7th Int. Conf. on Distributed Computing Systems, 1987
- [Coan 86] Coan, B.A., Oki, B.M., Kolodner, E.K., Limitations on Database Availability When Networks Partition, Proc. 5th ACM Symp. on Principles of Distributed Computing, 1986, pp. 187-194
- [Cornell 87] Cornell, D.W., Yu, P.S., A vertical Partitioning Algorithm for Relational Database, Proc. 3rd Int. Conf. on Data Eng., 1987, pp. 30-35
- [Cornell 88] Cornell, D.W., Yu, P.S., Site Assignment for Relations and Join Operations in the Distributed Transaction Processing Environment, Proc. 4th Int. Conf. on Data Eng., 1988, pp. 100-108
- [Chu 82] Chu, W.W., Hurley, P., Optimal Query Processing for Distributed Database Systems, IEEE Trans. on Computers, Vol. C-31, No. 9, 1982
- [Daniels 87] Daniels, D.S., Spector, A.Z., Thompson, D.S., Distributed Logging for Transaction Processing, Proc. ACM SIGMOD Conf. on Management of Data, 1987, pp. 82-96
- [Davidson 85] Davidson, S.B., Garcia-Molina, H., Skeen, P., Consistency in Partitioned Networks, ACM Computing Surveys, Vol. 17, No. 3, 1985, pp. 341-370
- [Egyhazy 88] Egyhazy, C., Trantis, K., A Query processing Algorithm For

- Distributed Relational Database Systems, *The Computer Journal*, Vol. 31, No. 1, 1988, pp. 34-40
- [El Abbadi 86] El Abbadi, A., Toueg, S., Availability in Partitioned Replicated Databases, *Proc. of 5th ACM Symp. on Principles of Database Systems*, 1986, pp. 240-251
- [Elmagarmid 88] Elmagarmid, A.K., Helal, A.A., Supporting Updates in Heterogeneous Distributed Database Systems, *Proc. 4th Int. Conf. on Data Eng.*, 1988, pp. 564-569
- [Freytag 87] Freytag, J.C., Cristian, F., Kaehler, B., Masking System Crashes in Database Application Programs, *Proc. 13th Conf. on Very Large Data Bases*, 1987, pp. 407-416
- [Ganski 87] Ganski, R.A., Wong, H.K.T., Optimization of Nested SQL Queries Revisited, *Proc. ACM SIGMOD Conf. on Management of Data*, 1987, pp. 23-33
- [Garcia-Molina 88] Garcia-Molina, H., Kogan, B., Achieving High Availability in Distributed Databases, *IEEE Trans. on Soft. Eng.*, Vol. 14, No. 7, 1988, pp. 886-896
- [Gawlick 85] Gawlick, D., Processing 'Hot Spots' in High Performance Systems, *Proc. IEEE CompCon*, 1985, pp.249-251
- [Gray 86] Gray, J.N., An Approach to Decentralized Computer Systems, *IEEE Trans. on Soft. Eng.*, Vol. SE-12, No.6, 1986, pp. 684-692
- [Haerder 83] Haerder, T., Reuter, A., Principles of Transaction-Oriented Database Recovery, *ACM Computing Surveys*, Vol. 15, No. 4, 1983, pp. 287-318
- [Haerder 87A] Haerder, T., Rothermel, K., Concepts for Transaction Recovery in Nested Transactions, *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1987, pp. 239-248
- [Haerder 87B] Haerder, T., Rothermel, K., Concurrency Control Issues in Nested Transactions, *Res. Rep. RJ 5803*, IBM Almaden Research Center, 1987, p. 25
- [Herlihy 86] Herlihy, M., A Quorum-Consensus Replication Method for Abstract Data Types, *ACM Trans. Computer Syst.*, Vol. 4, No.1, 1986, pp. 32-53
- [Herlihy 87] Herlihy, M., Extending Multiversion Time-Stamping Protocols To Exploit Type Information, *IEEE Trans. on Computers*, Vol. C-36, No. 4, 1987, pp. 443-448
- [Hevner 85] Hevner, A.R., Wu, O.Q., Yao, S.B., Query Optimization on Local Area Networks, *ACM Trans. on Office Information*, Vol. 3, No. 1, 1985, pp. 35-62
- [Ioannidis 87] Ioannidis, Y.E., Wong, E., Query Optimization by Simulated Annealing, *Proc. ACM-SIGMOD Conf. on Management of Data*, 1987, pp. 9-22
- [Jajodia 87] Jajodia, S., Mutchler, D., Dynamic Voting, *Proc. ACM SIGMOD Int. Conf. Management of Data*, 1987, pp. 227-238
- [Jarke 84] Jarke, M., Koch, J., Query Optimization in Database Systems, *ACM Computing Surveys*, Vol. 16, No. 2, 1984, pp. 111-152
- [Kim 84] Query Processing in Database Systems, W. Kim, D. Reiner and D. Batory, Eds., Springer-Verlag, 1984
- [Lohman 84] Lohman, G.M., Daniels, D., Haas, L.M., Kistler, R., Selinger, P.G., Optimization of Nested Queries in Distributed Relational Database, *Proc. of 10th Int. Conf. on Very Large Data Bases*, 1984, pp. 403-415.
- [Lynch 86] Lynch, N., Merritt, M., Introduction to the Theory of Nested Transactions, *Tech. Rep. MIT/LCS/TR-367*, MIT Laboratory, 1986
- [Morzy 89] Morzy, T., On Synchronization in K-version database systems, (in preparation)
- [O'Neil 86] O'Neil, P.E., The Escrow Transactional Method, *ACM Trans. on Database Systems*, Vol. 11, No. 4, 1986, pp. 485-430
- [Pu 88] Pu, C., Superdatabases for Composition of Heterogeneous Databases, *Proc. 4th Int. Conf. on Data Eng.* 1988, pp. 548-555
- [Reuter 86] Reuter, A., Load Control and Load Balancing in a Shared Database Management System, *Int. Conf. on Data Eng.* 1986, pp. 188-198

- [Robinson 85] Robinson, J.T., A Fast General Purpose Hardware Synchronization Mechanism, Proc. ACM SIGMOD Int. Conf. on Management of Data, 1985, pp. 122-130
- [Schwarz 84] Schwarz, P.M., Spector, A.Z., Synchronizing Shared Abstract Types, ACM Trans. on Computer Systems, Vol. 2,3, 1984, pp. 223-250
- [Sellis 88] Sellis, T.K., Multiple Query Optimization, ACM Trans. on Database Systems, Vol. 13, No. 1, 1988, pp. 23-52
- [Shenoy 87] Shenoy, S.T., Ozsoyoglu, Z.M., A System for Semantic Query Optimization, Proc. ACM SIGMOD Int. Conf. on Management of Data, 1987, pp. 181-195
- [Tantawi 85] Tantawi, A.N., Towsley, D., Optimal Static Load Balancing in Distributed Computer Systems, Journal of ACM, VOL. 32, No. 2, 1985, pp. 445-465
- [Wah 85] Wah, B., Lien, V., Design of Distributed Databases on Local Computer Systems with a Multiaccess Network, IEEE Trans. on Soft. Eng., Vol. SE-11, No. 7, 1985, pp. 606-619
- [Wang 85] Wang, Y.-T., Morris, R.J.T., Load Sharing in Distributed Systems, IEEE Trans. on Computer Systems, Vol. C-34, No. 3, 1985, pp. 204-217
- [Weikum 86] Weikum, G., A Theoretical Foundation of Multilevel Concurrency Control, Proc. of ACM Symp. on Principles of Database Systems, 1986, pp. 31-42
- [Yu 84] Yu, C.T., Chang, C.C., Distributed Query Processing, ACM Computing Surveys, Vol. 16, No. 4, 1984, pp. 399-433
- [Yu 87] Yu, C.T., Guh, K.C., Zhang, ., Templeton, M., Brill, D., Chen, A.L.P., Algorithms to Process Distributed Queries in Fast Local Networks, IEEE Trans. on Computers, Vol. C-36, No. 10, 1987, pp. 1153-1164

Streszczenie. Содержание.

Kilka uwag o badaniach prowadzonych w dziedzinie rozproszonych baz danych

Systemy rozproszonych baz danych, w porównaniu z systemami scentralizowanymi, umożliwiają, teoretycznie, uzyskanie istotnej poprawy większości parametrów charakteryzujących system bazy danych. Osiągnięcie potencjalnych korzyści wynikających z zastosowania tych systemów uwarunkowane jest efektywnym rozwiązaniem jakościowo nowych problemów dotyczących projektowania i zarządzania tymi systemami. Omówieniu tych problemów oraz przedstawieniu aktualnych kierunków badawczych w zakresie systemów rozproszonych baz danych poświęcona jest niniejsza praca.

Несколько замечок о исследованиях в области распределённых баз данных

Системы распределённых баз данных, по сравнению с системами централизованных баз данных, создают, теоретически, возможность значительного улучшения многих параметров системы баз данных. Для того чтобы получить на практике потенциальные преимущества ввиду использования этих систем надо эффективно решить ряд существенно новых проблем в отношении проектирования и управления этими системами. Обсуждению этих проблем и представлению текущих исследований в области систем распределённых баз данных посвящена настоящая работа.

sieć komputerowa,
testowanie zgodności protokołów,

Artur Dubielewicz *
Janusz Ratajczak *

STEROWANIE WYKONANIEM KOMPLETU TESTÓW W SYSTEMIE TESTOWANIA ZGODNOŚCI PROTOKOŁÓW W SIECI KASK

W prezentowanym referacie przedstawiono sposób sterowania przebiegiem wykonania kompletu testów zgodności protokołów, zastosowanym w, budowanym przez zespół Centrum Obliczeniowego Politechniki Wrocławskiej, testerze dla Krajowej Akademickiej Sieci Komputerowej. Pokazano zarówno sposób opisu kompletu testów (umożliwiający automatyzację procesu testowania), jak i zestaw działań realizowanych w czasie wykonania zadanego zestawu testów.

1. WSTĘP

Implementacja systemu testowania zgodności protokołów [1] proponowana w sieci KASK (Krajowa Akademicka Sieć Komputerowa) bazuje na języku wykorzystującym notację TTCN-MP [2].

Język TTCN nie daje możliwości prostego opisu kolejności wykonania poszczególnych egzemplarzy kompletu testów, stanowiącego pewną zamkniętą całość sprawdzającą poprawność implementacji protokołów. Dlatego też przyjęto, że taki opis będzie tworzony niezależnie w postaci zbioru tekstowego, nazywanego dalej Zbiorem Konfiguracyjnym Kompletu (ZKK).

2. ZBIÓR KONFIGURACYJNY KOMPLETU TESTÓW

ZKK składa się z dwóch głównych części:

- a) opisu parametrów formalnych testu (z podaniem zestawu wartości dla każdego parametru),
- b) opisu struktury kompletu testów (określającego kolejność wykonania poszczególnych egzemplarzy).

* Centrum Obliczeniowe Politechniki Wrocławskiej

Dodatkowo w opisie kompletu można umieścić komentarze dotyczące np. znaczenia parametrów kompletu testów, zestawu wartości dopuszczalnych dla każdego parametru, celu wykonania poszczególnych egzemplarzy testów itd. Ułatwi to wprowadzanie modyfikacji w ZKK oraz zwiększy czytelność opisu kompletu testów.

1.1. Parametry kompletu testów

Pierwsza część ZKK służy do określenia wartości wszystkich parametrów używanych w testach kompletu. Parametry wykorzystywane w danym egzemplarzu testu nazywamy jego parametrami aktywnymi. Wartość każdego parametru może być zadana albo pojedynczą wartością, albo zestawem wartości. Test będzie zawsze wykonywany dla wszystkich możliwych kombinacji wartości parametrów aktywnych.

1.2. Struktura kompletu testów

Druga część ZKK opisuje strukturę kompletu testów w postaci grafu określającego kolejność wykonania poszczególnych egzemplarzy testów. Z każdym węzłem grafu związany jest zestaw egzemplarzy testów. Graf opisany jest przez podanie wszystkich jego węzłów. Dla każdego węzła należy określić:

- nazwę węzła,
- zestaw egzemplarzy testów należących do węzła,
- węzły następne (następniki).

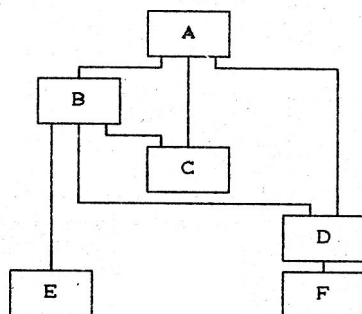
Kolejność zapisu węzłów nie jest istotna, natomiast egzemplarze testów będą wykonywane według kolejności występowania na liście. Podobnie kolejność węzłów na liście następników determinuje kolejność wykonania zestawów egzemplarzy testów.

3. WYKONANIE TESTÓW PRZEZ SYSTEM

Wykonanie testów będzie inicjowane komendą RUN-TEST. Parametry tej komendy muszą wskazywać Zbiór Konfiguracyjny Kompletu testów, oraz opcjonalnie węzeł w grafie struktury kompletu od którego ma się rozpocząć testowanie, egzemplarz testu należący do wybranego węzła oraz sposób wykonania testu.

Uruchomienie testowania od zadanego węzła spowoduje wykonanie egzemplarzy testów należących do wybranego węzła, a następnie kolejno egzemplarzy należących do jego następników.

Na przykład przy strukturze kompletu określonej grafem :



i wskazaniu węzła B jako początkowego, kolejność wykonania będzie następująca : egzemplarze węzłów B, E, D, F, C.

3.1. Komenda RUN-TEST

Format komendy:

`RUN-TEST [dev:] nazwa [\węzeł] [\egzempl] [tryb] [ALL]`

- `dev:` - nazwa urządzenia zawierającego ZKK,
- `nazwa` - nazwa zbioru zawierającego ZKK,
- `węzeł` - nazwa węzła od którego należy rozpocząć testowanie (jeśli zostanie opuszczona wykonany zostanie cały komplet testów),
- `egzempl` - jeśli jest podana nazwa egzemplarza to testowanie rozpocznie się od wybranego egzemplarza,
- `tryb` - docelowo przewiduje się dwa tryby testowania :
 - a) wykonanie zadanego zestawu testów bez ingerencji operatora
 - b) wykonanie krokowe (przejsięcie do wykonania następnego egzemplarza testu wymaga akceptacji operatora).
- `ALL` - oznacza żądanie wykonania wszystkich testów począwszy od wskazanego węzła, jeśli zostanie pominięty wykoną się tylko wybrany test (egzemplarz, grupa testów).

Przykłady wywołań:

`RUN-TEST nazwa lub RUN-TEST nazwa ALL`

- wykonaj wszystkie egzemplarze testów w komplecie

`RUN-TEST nazwa \węzeł ALL`

- wykonaj wszystkie egzemplarze testów począwszy od wybranego węzła

RUN-TEST nazwa\węzeł

- wykonaj tylko egzemplarze testów wybranego węzła

RUN-TEST nazwa\węzeł\egzempl

- wykonaj tylko wybrany egzemplarz testu.

3.2. Wykonanie komendy RUN-TEST w TESTERZE

W wyniku wydania komendy RUN-TEST realizowany jest następujący ciąg działań :

1. Wybór , na podstawie Zbioru Konfiguracyjnego Kompletu, kolejnego egzemplarza testu do wykonania.
2. Wywołanie kompilatora w celu translacji wybranego egzemplarza testu.
3. Ustalenie kolejnej kombinacji wartości parametrów i podstawienie ich w miejsce parametrów formalnych kompletu testów Dla każdego egzemplarza testu znany jest zestaw jego parametrów aktywnych. Tylko aktywne parametry danego egzemplarza będą brane pod uwagę przy generowaniu kombinacji wartości.
4. Wywołanie kontrolera testu w celu wykonania egzemplarza testu z ustalonymi wartościami parametrów.
5. Po pozytywnym zakończeniu testu następuje powtórzenie kroków 3 i 4. Jeśli test zakończył się błędem - następuje zakończenie testowania. W przypadku gdy wyczerpano wszystkie kombinacje parametrów zadanych komendą RUN-TEST nastąpi przejście do wykonania punktu 1. Wykonanie wszystkich egzemplarzy testów kończy obsługę komendy RUN-TEST.

W czasie testowania (tzn. po wydaniu komendy RUN-TEST) mogą wystąpić różne nieprawidłowe sytuacje. Zostaną one zasygnalizowane operatorowi przez wysłanie odpowiedniego komunikatu.

4. PODSUMOWANIE

Zaprezentowany w artykule sposób sterowania wykonaniem kompletu testów sprawdzającego zgodność implementacji ze standardem wydaje się być wygodny i elastyczny w użyciu. Z jednej strony projektant może napisać komplet testów, w języku TTCN, sprawdzający implementację w możliwie pełnym zakresie, z drugiej strony użytkownik może uruchamiać dowolny element kompletu (komplet, grupę, egzemplarz) zgodnie ze swoimi bieżącymi potrzebami. Dodatkowo wybrany element kompletu może być realizowany z dowolnie zadanymi zestawami parametrów, bez konieczności ich każdorazowego specyfikowania.

S. BIBLIOGRAFIA

- [1] Dubielewicz I., Komorowski W., Implementation of the conformance testing system, Proc. of Computer Networks'89.
- [2] Kolesnik K., Język TCN w systemie testowania zgodności sieci KASK, Proc. of Computer Networks'89.

THE TEST SUITE CONTROL IN THE KASK CONFORMANCE TESTING SYSTEM

The control of a suite test execution of the protocol conformance testing system for the Polish Academic Computer Network (KASK) is presented in the paper. It is shown both the description of a suite tests (automatizing the test execution process) and the set of activities realized during execution of the indicated suite test.

УПРАВЛЕНИЕ РЕАЛИЗАЦИЮ КОМПЛЕТА ТЕСТОВ В СИСТЕМЕ ТЕСТИРОВКИ СОГЛАСИЯ ПРОТОКОЛОВ В СЕТИ KASK.

В работе представляется метод управления выполнением комплекта тестов испытывающих согласие протоколов в Польской Академической Вычислительной Сети (KASK). Описывается принципы образования комплекта тестов, который дает возможность автоматизирования процесса тестирования, а также состав действий реализованных во время выполнения тестов.

computer network.
conformance testing

Iwona DUBIELEWICZ*
Witold KOMOROWSKI*

AN IMPLEMENTATION OF THE CONFORMANCE TESTING SYSTEM

In the paper a proposed implementation of the conformance testing system for the KASK network is presented. The hardware and software aspects of this realization are described.

1. INTRODUCTION

The works concerning the conformance testing undertaken in the KASK (National Academic Computer Network) follow the suggestions and recommendations given by ISO [5]. It was decided that the language described by TTCN-MP grammar and presented in the ISO/IEC DP 9646-2 Annex F (January 1988) would be used for the tests definition [3].

Considering actual architecture of the KASK network [7] the conformance testing system is limited and it is assumed it can only test the implementation of the transport protocols [1]. But it is expected that in the future the system will be developed in such a way that the conformance testing of the implementation of upper layers protocols will be possible.

2. CONFORMANCE TEST METHOD FOR KASK NETWORK

It was decided to apply the distributed test method [4] which requires direct access to the upper boundary of the implementation under test (IUT) and consequently the detailed specification of the control and observation requirements at both of the ends should be given.

The realization of the distributed method implies that much more work must be done to connect the upper tester to transport protocol layer in every tested system. To minimize these efforts, the upper tester is designed in such a manner that only one part of it, namely

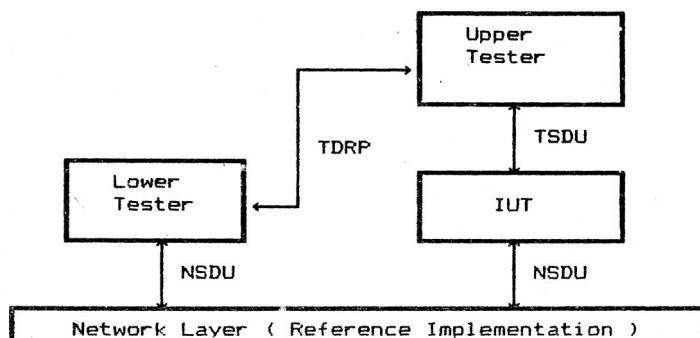
* Computer Center, Technical University of Wrocław

this one which communicates with the IUT, will be constructed and coded separately for every system. The rest of the upper tester is designed as a portable module, coded in the high level language, and it can be easily implemented in every system.

The upper tester (Responder) is designed to operate as an automaton controlled by the transport service primitives. According to [6] this type of upper testers have a good flexibility related to their design complexity.

It is assumed that the lower tester (Driver) is designed as an encoder/decoder of transport protocol data units. This kind of lower tester has the complicated structure, but it allows to test almost all the interesting features of IUT.

The communication between both parts of the testing system is achieved by means of the Tester Driver-Responder Protocol (TDRP), which was designed especially for this purpose. The figure below shows the architecture of the conformance testing system for the KASK network.



Architecture of conformance testing system for the KASK network

To verify this conception it was decided that the prototype of the upper tester would be connected to the transport layer of the microhost. The main reason of this decision was that the boundaries of every protocol layer implemented in the microhost are well defined and it is very easy to connect the upper tester to its transport layer. For that reason also the Driver is situated in the microhost programming environment.

3. IMPLEMENTATION OF THE SYSTEM

Lower tester implementation.

The lower tester (Driver) is located on the IBM PC/AT equipped with 4 MB RAM and SDLC Adapter card. Besides the DOS 3.3 operating system, the

computer has an additional special software system, which creates a virtual machine for the easy implementation of the protocol layers. The Driver operates in the environment created by this virtual machine and it communicates with the Network Layer by means of the Network Service Primitives.

The main modules of the Driver are:

- Command Interpreter,
- TTCN Translator,
- Code Generator,
- Test Controller,
- Registration Module.

Command Interpreter is the module which main function is to read and execute the commands of the operator. Hence, it performs:

- initialization of the Translator to compile the suite test definition program for the checking its completeness and correctness,
- activation of the Code Generation module to execute required test suite/group/case,
- activation of the Registration module to display registered data.

TTCN Translator performs the lexical and syntactic analysis of the required test case. The input data to the translator is the abstract test case and the output of the translator is the internal representation of the test case behaviour tree with references to the lists of constraints and parameters values. It is assumed that all the definitions of the test cases are collected in the Tests/Test-Steps Libraries. These libraries contain the tests written in the source (TTCN) code. Hence the translation of any test case must be done every time the test is to be executed (on demand of the Code Generator module).

Code Generator is a module which activates and controls the test suite/group/case execution. It creates the executable test case from the given internal form of test case and activates Test Controller module. To ensure the execution of the whole test suite or test group it investigates the tree of test suite/group execution structure. This structure is given in the separate text file as a graph [2]. Code Generator calls the Translator module to translate the subsequent abstract test case and informs the operator about the results obtained during the test execution.

Test Controller is a module which interpretes the executable test case. It communicates with the network layer by sending and receiving service primitives. Comparing the expected behaviour with actual one it generates the result of the test execution. It also logs all the sent and received service primitives into the Results file to enable the

off-line analysis.

Registration module performs the decoding/encoding of data gathered during the testing process to print/display them in the readable form.

All but one modules are proposed to be programmed in C language, only the Translator module - in TURBO Pascal.

Upper Tester implementation.

The upper tester (Responder) in his first implementation in the KASK will be located on the IBM PC/AC together with the microhost. The Responder consist of 3 parts: Controlling Responder (TRC) and two Testing Responders - Calling (TRG) and Called (TRD). It is assumed that the TRC is always accessible from Driver. Hence, the assumption that IUT has the ability to exchange the most simple transport services with the resident TRC must be fulfilled. The connection between Driver and TRC is used for TDRP command transport; connection between Driver and one of Testing Responders is used for test primitives exchange. Both TRD and TRG may be created and deleted by TRC as a result of suitable TDRP command issued from Driver.

Responder is designed as finite-state machine, with states changed according to the events and output depending on the operation mode programmed before. The TRC as well as every Testing Responder is programmed from Driver by the TDRP commands. The mode is characterized by several parameters describing the Responder's behaviour.

Every appearing event (e.g. receiving some transport primitive) causes the sequence of actions undertaken by Responder, (e.g. it sends some service primitive, stops the data generation etc.). Responder collects some data characterizing the test execution. This data are accessible to the Driver.

All Responder's modules will be programmed in C language.

4. CONCLUSIONS

It should be emphasized that presented conformance testing system is not planned to play the role of the testing centre or laboratory in the sense of ISO/IEC DP 9646-3,4. It was designed to facilitate and simplify the testing of protocol implementation in the KASK network. It also gives us some experiences in the conformance testing area.

REFERENCES

- [1] Dubielewicz I., Fryźlewicz Z., Kolesnik K., Komorowski W., Misiura L., Koncepcja testowania zgodności implementacji protokołów ze standardami. C.O.Pol.Wrocł. SPR 8/87.
- [2] Dubielewicz A., Ratajczak J., The test suite control in the KASK conformance testing system. Proc. of Computer Networks '89.
- [3] Dubielewicz I., Kolesnik K., Język TTCN w systemie testowania zgodności sieci KASK. Proc. of Computer Networks '89.
- [4] OSI Conformance Testing Methodology and Framework - Part 1: General Concepts, ISO/IEC DP 9646-1, 1988-03.
- [5] OSI Conformance Testing Methodology and Framework - Part 2: Abstract Test Suite Specification, ISO/IEC DP 9646-2, 1988-03.
- [6] Pavel R.J., Dwyer D.J., Some Experiences of Testing Protocol Implementations, in: Protocol Specification, Testing and Verification, IV IFIP, North Holland 1985 pp.657-677.
- [7] Protokoły w międzyuczelnianej sieci komputerowej (Red. E.Bilski). Wyd.Pol.Wrocł., Wrocław 1987.

REALIZACJA SYSTEMU TESTOWANIA ZGODNOŚCI W SIECI KOMPUTEROWEJ KASK

Przedstawiono koncepcję realizowanego obecnie na Politechnice Wrocławskiej systemu testowania zgodności implementowanych komponentów sieci komputerowej KASK z protokołami. Podano ogólną architekturę systemu i omówiono aspekty sprzętowe i programowe systemu.

testowanie zgodności,
język TTCN,
sieć KASK

Iwona DUBIELEWICZ*

Krystyna KOŁEŚNIK*

JĘZYK TTCN-MP W SYSTEMIE TESTOWANIA ZGODNOŚCI IMPLEMENTACJI PROTOKOŁÓW SIECI KASK.

W pracy podano krótką charakterystykę języka TTCN oraz omówiono zasady jego implementacji w systemie testowania zgodności dla sieci KASK. Język TTCN został przyjęty przez grupę WG16 ISO jako obowiązująca notacja dla zapisu kompletów standardowych testów zgodności. Uproszczona wersja tego języka, nazwana TTCN-MPO została przyjęta do realizacji w sieci KASK, co pozwoli na łatwe zastosowanie standardowych testów do badania zgodności realizacji protokołów w istniejących i nowych elementach tej sieci.

1. WPROWADZENIE.

Testowanie implementacji protokołów sieciowych ma na celu sprawdzenie ich zgodności ze standardami tych protokołów.

Standardowe testy zgodności tworzą tzw. komplety testów, składające się z grup egzemplarzy testów powiązanych celami testowania oraz z biblioteki zawierającej wspólne fragmenty egzemplarzy testów czyli tzw. kroki (mogą być wykorzystywane w wielu egzemplarzach) [4,5].

Egzemplarz testu opisuje zachowanie systemu testującego w postaci sekwencji zdarzeń określających obiekty wysyłane i odbierane, do i z testowanej implementacji. Zdarzeniami testu są prymitywy usługowe generowane i odbierane przez system realizujący test, zaś ich parametrami - jednostki danych testowanego protokołu. Ponieważ testowana implementacja może różnie odpowiedzieć na każdy prymityw nadany przez urządzenie testujące, w teście muszą być zapisane wszystkie możliwe reakcje badanej implementacji, czyli dla każdego zdarzenia wysyłanego muszą być przewidziane zdarzenia wariantowe. Stąd też zachowanie testu jest drzewem, którego węzłami są tzw. linie zdarzeń. Każda linia zdarzenia występuje w drzewie na określonym poziomie jednoznacznie opisującym jej położenie w sekwencji linii zdarzeń, do której należy.

* Centrum Obliczeniowe Politechniki Wrocławskiej, WROCŁAW

Sygnalizowane opublikowanie standardowych kompletów testów umożliwi wykorzystanie ich w wielu systemach testujących przy założeniu, że systemy te będą akceptować formę ich zapisu. Grupa WG16 Komitetu Standaryzacyjnego ISO rozpatrywała możliwość zapisu standardowych testów w różnych językach np. LOTOS, SDL, TDL i ESTELLE, a także przy użyciu notacji tablicowo-drzewowej TTCN (ang. Tabular-Tree Combined Notation). Po wielu dyskusjach zdecydowano się na przejęcie metody TTCN, dla której opracowano dwie wersje: publikacyjną - TTCN-GR i implementacyjną - TTCN-MP [5]. Wspomniane języki były rozpatrywane także przy opracowywaniu koncepcji systemu testującego dla sieci KASK [4]. Ostatecznie przyjęto, że zostanie opracowana i zrealizowana uproszczona wersja języka TTCN-MP, nazwana TTCN-MPO [2].

2. CHARAKTERYSTYKA JĘZYKA TTCN-MP.

Program kompletu testów zapisany w języku TTCN-MP składa się z czterech sekcji: nagłówka, deklaracji, ograniczeń i zachowania.

Sekcja nagłówek zawiera tekstowe informacje o przeznaczeniu kompletu oraz referencje do podanych (przez realizatora protokołu) informacji opisujących testowaną implementację.

W sekcji deklaracji opisuje się dwa typy obiektów używanych w sekcji zachowania: obiekty proste i obiekty złożone. Obiektami prostymi są:

- parametry kompletu testów,
- stałe i zmienne globalne,
- stopery,
- punkty kontroli i obserwacji.

Obiektami złożonymi są:

- abstrakcyjne prymitywy usługowe (ang. Abstract Service Primitives-ASP),
- jednostki danych protokołu (ang. Protocol Data Units - PDUD).

Dla obiektów prostych definiuje się nazwę, typ i zakres wartości jakie dany obiekt może przyjmować. Dla obiektów złożonych definiuje się strukturę i rodzaj jego składowych oraz ich atrybuty. Definicja atrybutów składowych obiektów złożonych jest taka sama jak dla obiektów prostych.

Język TTCN-MP zawiera kilkanaście typów predefiniowanych. Użytkownik może też definiować własne typy wyliczeniowe i ich podtypy. Obiekty omówionych typów można w teście porównywać używając standardowych operatorów relacji lub nadawać im wartość stosując operację przypisania.

Definicja punktów kontroli i obserwacji podaje nazwę i przeznaczenie tych punktów. Punkty te oznaczają symbolicznie miejsce

nadawania i odbioru prymitywów usługowych.

Definicje ASP zawierają opis prymitywów usługowych warstwy niższej (w stosunku do testowanego protokołu), a definicje PDU opisują jednostki testowanego protokołu.

W sekcji ograniczeń określa się zestawy wartości atrybutów obiektów złożonych. Jeden zestaw wartości składowych określonego PDU lub ASP stanowi jego listę ograniczeń identyfikowaną przez nazwę. W sekcji ograniczeń deklaruje się też tzw. zmienne wolne, które są przeznaczone do tymczasowego przechowywania wartości pól obiektów złożonych, których wartości są zależne od przebiegu transmisji.

W sekcji zachowania, która jest odpowiednikiem treści zwykłego programu, definiuje się kolejno grupy i egzemplarze testów do nich należące. Każdy egzemplarz składa się z nagłówka, umożliwiającego jego identyfikację, oraz z definicji zachowania. W definicji zachowania wymienia się sekwencje oczekiwanych zdarzeń aż do ich wyczerpania, czyli dojścia do zdarzenia, które ma określony werdykt (i tym samym końcowy poziom "i"). Następnie opisuje się zdarzenia alternatywne i ich sekwencje zdarzeń dla poziomu "i-1" (jeżeli takie istnieją) itd., aż do osiągnięcia poziomu początkowego - "1".

Do podstawowych typów linii zdarzeń testu należą linie "wyślij" lub "odbierz" wskazany prymityw usługowy. W teście mogą występować także inne typy linii zdarzeń tj.: odbierz dowolny prymityw usługowy, odbierz sygnał upływu czasu, opóźnij działanie o wskazany odcinek czasu, nadaj wartość obiektowi globalnemu, zbadaj wartość obiektu globalnego, wykonaj operację stoperową, skocz do linii wskazanej etykieta.

Niektóre linie zdarzeń mogą opisywać więcej niż jedno zdarzenie i w tym przypadku są one nazywane składowymi liniami lub zdarzeniami elementarnymi. Np. w liniach typu "wyślij" lub "odbierz" można nadawać wartości polom obiektów złożonych, używanym w danej linii, lub badać ich stan. Budowę linii zdarzeń omówiono szczegółowo w [2].

W opisie zachowania testu mogą też występować linie, które nie będą miały swoich odpowiedników w teście wynikowym. Są to tzw. pseudozdarzenia, które określają pewne działania dla translatora. Do pseudozdarzeń należą: dołącz drzewo kroku testu, powtarzaj drzewo, wykonuj drzewa równoległe.

Dla egzemplarzy testów mogą być też określone tzw. zachowania domyślne (zapisane również w postaci drzewa) wykonywane wówczas, gdy nie zachodzi żadne ze zdarzeń wariantowych danego poziomu drzewa testu.

Opis składni implementowanego języka w notacji BNF oraz semantykę operacyjną języka podano w [2].

3. IMPLEMENTACJA TTCN-MPO.

Uwzględniając sugestie zawarte w ocenie propozycji standardu TTCN [6] oraz możliwości realizacyjne zespołu, o implementacji translatora TTCN-MPO założono:

- podzbiór typów predefiniowanych zawiera typy : całkowity (Integer), boolowski (Boolean), łańcuch bitowy (BitString), łańcuch oktetów (OctetString), łańcuch znaków (CharacterString);
- operatory dla wartości typu Integer to operator dodawania, odejmowania i operator 'modulo';
- wyrażenia logiczne i arytmetyczne będą wyłącznie dwuargumentowe;
- nie wykorzystuje się parametrów list ograniczeń i zmiennych generycznych;
- dołączanie drzew domyślnych jest jawne przez podanie specjalnego operatora dla oznaczenia tego działania;
- istnieje dodatkowa operacja stoperowa umożliwiająca odczyt stanu stopera.

W implementacji TTCN-MPO dla sieci KASK przyjęto jednopoziomową strukturę grup testów w obrębie kompletu. Komplet testów przechowywany jest w postaci źródłowej tj. w postaci plików tekstowych, zawierających zapis testu w języku TTCN-MPO. Test ten to tzw. test abstrakcyjny (wzorzec testu), w którym wartości atrybutów obiektów w nim używanych nie są jeszcze określone. Przekształcanie testu abstrakcyjnego w test rzeczywisty (wykonywalny) jest dokonywane w urządzeniu testującym w trakcie wykonywania testu w oparciu o struktury danych wygenerowane podczas translacji.

W urządzeniu testującym dla sieci KASK, translacja kompletu testu przebiega dwustopniowo i składa się z tzw. translacji wstępnej i translacji właściwej. W czasie translacji wstępnej tworzona jest struktura kompletu testów tzn. katalogi kompletów, grup, egzemplarzy i kroków, oraz plik źródłowy dzielony jest na pliki egzemplarzy testów. Również na tym etapie jest badana pełność kompletu w sensie umieszczenia w bibliotece kroków, wszystkich kroków, które są wykorzystywane w różnych egzemplarzach testu.

Translacja właściwa dotyczy za każdym razem translacji części deklaracyjnej kompletu i translacji jednego egzemplarza testu abstrakcyjnego. Pseudozdarzenie "dołącz drzewo" wskazanego kroku powoduje, że translator zmienia analizowany plik źródłowy egzemplarza testu na plik źródłowy wskazanego kroku, i dopiero po jego wyczerpaniu wraca do analizy głównego pliku (począwszy od zdarzenia następnego po "dołącz drzewo").

Przyjęto, że uzyskane w wyniku translacji drzewo zachowania będzie

reprezentowane przez drzewo binarne, w którym każdy węzeł wskazuje na węzeł sekwencyjny i jeden z węzłów wariantowych.

System testujący dla sieci KASK, w którym ma być wykorzystywany język TTCN-MPO, jest przeznaczony do testowania warstwy transportowej. Stąd też w komplecie testów opisywać się będzie prymitywy usługowe warstwy sieciowej oraz jednostki danych protokołu transportowego.

4. ZAKOŃCZENIE.

Wszystkie przyjęte ograniczenia i uproszczenia w realizacji języka TTCN mają na celu zmniejszenie rozmiarów translatora i umożliwienie jego implementacji do września 1989r. Nie ograniczają one możliwości zapisywania dowolnych testów, a jedynie wydłużają ten zapis. Omawiana wersja języka została zastosowana do zapisu kompletu testów bazowych protokołu transportowego [3].

LITERATURA

- [1] Dubielewicz I., i inni : Koncepcja testowania zgodności implementacji protokołów ze standardami., Rap. CO PWr., SPR 8/87.
- [2] Dubielewicz I., i inni : Język definiowania i uruchamiania testów w sieci KASK., Rap. CO PWr., SPR 21/88.
- [3] Dubielewicz I. : Komplet testów zgodności dla klasy 0 Protokołu Transportowego., Rap. CO PWr., SPR 24/88.
- [4] OSI Conformance Testing Methodology and Framework - Part 1 : General Concepts, ISO/IEC DP 9646-1, 1988-03.
- [5] OSI Conformance Testing Methodology and Framework - Part 2: Abstract Test Suite Specification, ISO/IEC JTC 1/ SC21 No 2041 Rev., 2nd DP 9646-2, 1988-03.
- [6] Summary of Voting on 2nd DP, 9646-2, Information Processing Systems- OSI Conformance Testing Methodology and Framework - Part 2: Abstract Test Suite Specification. ISO/IEC JTC 1/SC 21 No 2767, 1988-04.

TTCN IMPLEMENTATION IN THE CONFORMANCE TESTING SYSTEM FOR THE KASK NETWORK

In the paper the characterization and some aspects of the implementation of the TTCN language are given. TTCN language is accepted by the WG16 ISO group as the notation for the describing the standard conformance test suites. The implementation of this language in the KASK network guarantees that it will be possible to use the standard tests for the conformance testing of all the elements working in the KASK network. Due to this solution our conformance testing system can be developed in the conformance testing laboratory in the future.

Odra 1300,
teletransmission,
microprocessor

J. W. DUDZIAK *

M. WNUK **

Microprocessor based data transmission controllers for the Odra computer.

1. Introduction

Teleprocessing is both effective and convenient method of using mainframe resources. Microprocessor-based devices are particularly useful for data transmission control and management because of their byte-oriented architecture, flexibility, and low prices.

The most popular computer in the country, Odra 1300 (compatible with ICL1900) works under the operating system George 3 which provides wide variety of telecommunication procedures. This software allows batch and interactive access to CPU resources using different remote terminals. The lack of suitable devices directed us towards design, development and construction of some types of Z80-based controllers.

There are three main tasks to be performed:

- a) physical splitting of CPU I/O channel to enable multiaccess.
- b) fitting the CPU standard interface and serial interface signals
- c) partial (e.g. physical or physical and communication level) or full realization of communication protocol for various terminals being used.

The tasks mentioned above are divided among some devices. In the Odra system 40 channel multiplexer (MPX) with Line Termination Units (LTU) (one for each channel), 32-channels scanner and front-end processor with scanners are available. The most frequently used configuration contains MPX with appropriate set of LTU's. In this case MPX realizes task a) and LTU - task b) and c).

2. Microprocessor based LTU

Microprocessor based LTU called UT1300 contains 4 channels for data transmission (Fig.1). It is a specialized Z80-based microcomputer containing 1kB RAM and up to 32kB ROM. Each channel of new multichannel line adapter consists of 2 main parts: line controller (modem side) and

*Computex Co. Ltd., Konopnickiej 15b, 51141 Wrocław, Poland

**Institute of Technical Cybernetics, Technical University of Wrocław,
Wybrzeże Wyspiańskiego 27, 50370 Wrocław, Poland

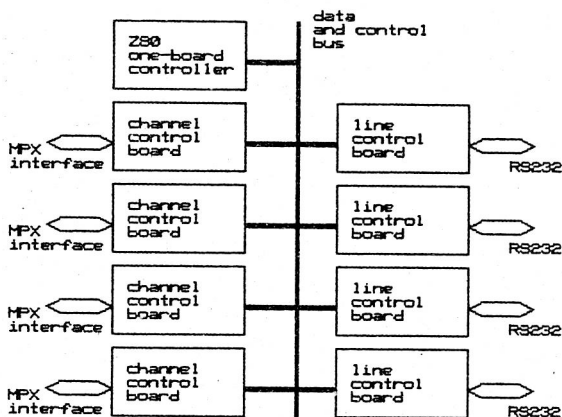


Fig. 1. The structure of LTU.

interface controller (host side). The whole (four channels) is controlled by Z80 based single board microcomputer. The host computer interface is the same for each channel and is independent of the line transmission method (synchronous or asynchronous) and protocol. Line controller is fully programmable. The control program is responsible for programming the line controller accordingly to desired transmission protocol, protocol realization, and servicing and controlling the interface lines. Channel adapter contains 8255 port working in unidirectional hand-shake mode, and support logic to fit MPX interface requirements. Line adapter consists of 8251 USART and 8253 programmable timer providing both: transmission clock and time-out interrupts.

All of the channels are independent. Transmission parameters (i.e. baud rate, transmission mode and type of terminal that are connected are user-programmable using jumpers on the technical board. The control program of LTU consists of two main parts: initialization routine, which sets all the parameters for each channel (depending on the jumpers), and the transmission routine written as pure procedure.

Each channel has its own data area in RAM space. All the in/out devices are memory mapped in order to use the indexed addressing facility. The device address contains also information about the channel number. The service of devices by the program is interrupt based. Each channel or line module can cause an interrupt. The interrupts are daisy chained and serviced in mode 2 (vectored). An 8 bit vector identifying the interrupt is generated using the module address jumpers. The exact identification of interrupt reason can be made in interrupt service routine by reading the device control word or the states of appropriate signals. The interrupt service method used in LTU is shown in Fig.2.

LTU can serve the transmission between MPX and different terminals. There are two basic transmission service procedures: one for all the terminals using asynchronous mode, second for synchronous mode (remote batch stations, VDUs). All the variations caused by individual differences between terminals are included in both procedures. Because of byte-buffered transmission method the transmission protocol is only partially done by LTU. Addressing and selecting the peripheral device are made by mainframe. LTU watches only the addressing sequence in order to check what transfer direction is selected and then its tasks are transmission correctness checking, error service and transmission termination. All the software and hardware solutions aim at speeding the

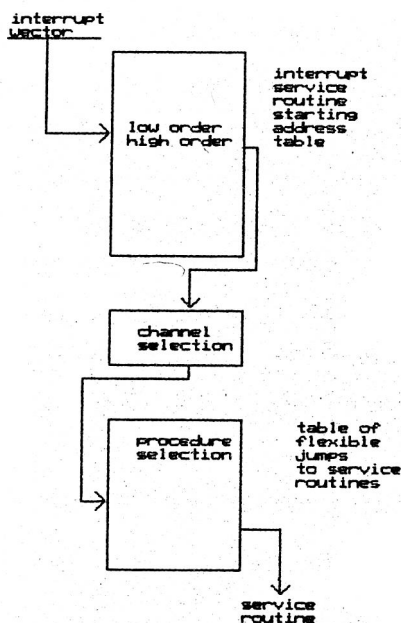


Fig. 2. Interrupt service.

service. In effect the interrupt service time does not exceed 100 microseconds, thus the global data transmission rate (in four channels) can reach 10000 bytes/s.

3. Scanner

Scanner consists of a selector and a set of (up to 4) line scanners, each having 8 transmission lines. Selector mates directly with Odra CPU Standard Interface (SI). Both selector and line scanner are specialized Z80 based microcomputers. Selector serves the SI signals. Because of high speed of data transfer through SI the direct access to selector memory has been used. Dedicated hardware generates the direct response vector. Selector performs also character conversion between internal (6-bit) and standard ASCII code. Scanner can work in many transmission modes. There exists a group of instructions, which prepares each channel of scanner for desired mode of transmission. These instructions are sent to scanner during its initialization. Scanner is fully transparent for the transmission excluding parity generation and checking. The selector program performs the analysis and execution of some instructions, and puts the instructions and data to appropriate line scanner. Each line scanner serves 8 transmission lines (fig.3). Z80SIO are used as line adapters. The address space of line scanner is divided into 8 kB blocks. The communication between selector and line scanners is provided by the dual port RAM. 8 kB RAM of each line scanner is also placed into the address space of selector, so in maximal configuration the memory of selector consists of four block (8kB each) of dual-port RAM, the own selector RAM, and ROM containing the control program. The access to the dual-port RAM is controlled by the arbiter. The partition of line scanner processor address space allows the processor to work without any holding if it asks no access to the common

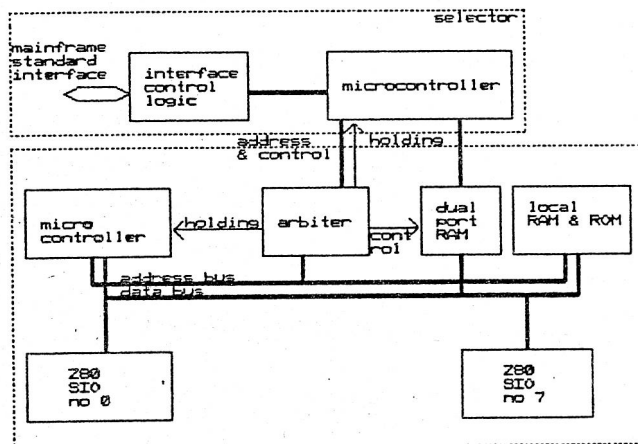


Fig. 3. Scanner structure

RAM area. Line scanner is equipped with watch-dog circuitry providing its automatic restart in case of going out of control.

4. Conclusions

The above presented structures of LTU and scanner were the basis for design and manufacturing of both the units. The produced devices have a good functional properties. The throughput of telecommunication service procedures of Odra's operating system becomes the bottleneck while using the character buffering transmission method. The results mentioned above, together with earlier results of experiments with using computers and minis as front-ends and computer network nodes, encouraged us to start preparing microprocessor-based FEP for Odra mainframe.

Mikroprocesorowe starowniki transmisji danych dla m.c. Odra 1300.

W artykule przedstawione zostały główne założenia i architektura mikroprocesorowych kontrolerów transmisji danych zaprojektowanych dla m.c. Odra 1300. Urządzenia te zastępują stare zrealizowane w pełni hardware'owo: UPD multipleksera MPX325 i skaner ICL7930.

Programowanie współbieżne,
specyfikacja,
logika modalna.

Adam FIERLA*

JEZYK LOGIKI MODALNEJ
DLA SYSTEMÓW KOMUNIKUJĄCYCH SIĘ**

W pracy zaprezentowano zarys modalnego języka specyfikacji logicznych własności protokołów sieciowych, zapisanych w postaci procesów CCS*. Opisano składnię i semantykę operacyjną języka, przedstawiono koncepcję systemu dowodowego oraz przeanalizowano w skrócie ekspresywność języka, tzn. zakres własności w nim wyrażalnych.

1. WSTĘP

Intensywny rozwój różnego typu sieci komputerowych stawia nas przed koniecznością opracowania formalnych metod konstruowania oprogramowania sieciowego.

Dotychczas powstało wiele różnych modeli sieci komputerowych. W [9] zaprezentowano ich schematyczną klasyfikację oraz uzasadniono decyzję skoncentrowania się na abstrakcyjnych programach współbieżnych jako modelu protokołów sieciowych.

Pośród języków programowania współbieżnego unikalną pozycję zajmuje CCS - Calculus of Communicating Systems, zaproponowany przez Milnera [8] jako formalny model współbieżności, dobrze nadający się do studiów teoretycznych. Jego rozwinięciem jest LOTOS [5], niedawno przyjęty przez ISO jako międzynarodowy standard (IS). Może on służyć zarówno jako narzędzie formalnego opisu oprogramowania sieciowego, jak i jako język programowania współbieżnego. LOTOS powstał z połączenia CCS* - pewnej modyfikacji tradycyjnego CCS Milnera, z techniką specyfikacji abstrakcyjnych typów danych ACT ONE.

Samo narzędzie opisu to jeszcze nie wszystko. Chcielibyśmy dysponować nie tylko językiem funkcjonalnego opisu sieci komputerowych,

*Centrum Obliczeniowe, Politechnika Wrocławska

**Pracę zrealizowano w ramach CPBR 8.13 "Budowa Krajowej Akademickiej Sieci Komputerowej", cel nr 52 "Formalne metody specyfikacji i konstrukcji oprogramowania sieciowego".

ale również językiem wyrażania ich własności, a także efektywnym i możliwie sformalizowanym sposobem weryfikacji, że wyspecyfikowany system rzeczywiście ma opisane własności (czyli systemem dowodowym).

Klasyczne systemy (np. logiki Hoare'a) okazują się mało przydatne do tego celu, gdyż ich adaptacja do współbieżności jest trudna. Dlatego w ostatnim dziesięcioleciu nastąpił ogromny wzrost zainteresowania różnymi systemami logiki modalnej i temporalnej. Logiki te są wystarczająco bogate dla wyrażania własności programów współbieżnych, a przy tym umożliwiają dobre sformalizowanie procesu weryfikacji. Do dziś powstało wiele temporalnych systemów dowodowych ([1], [6], [7]).

Wśród wielu nurtów istnieje jeden, zmierzający do zbudowania języka logiki modalnej dla CCS wraz z formalnym systemem dowodowym [11], [10]. Język ten jest, jak dotąd, bardzo ubogi (pominięto wszystkie aspekty nie związane ze współbieżnością) i należy traktować go jedynie jako propozycję wyjściową.

W niniejszej pracy przedstawiono zarys analogicznego języka logiki modalnej dla CCS*, co może stanowić podstawę do dalszych prac ukierunkowanych już na LOTOS. W kolejnych punktach przedstawiono składnię i semantykę CCS* oraz składnię i semantykę języka logiki modalnej wraz z krótką analizą jego ekspresywności. Ze względu na szczupłość miejsca nie zaprezentowano samego systemu dowodowego (aksjomatów i reguł wnioskowania). Znajdzie się on, wraz z przykładami, dowodem jego niesprzeczności i pełności oraz kierunkami dalszych badań, w przygotowywanym raporcie [3].

2. CCS* - WPROWADZENIE

Ponieważ interesują nas przede wszystkim aspekty współbieżności, w niniejszej pracy pominiemy całkowicie mechanizm przekazywania i przypisywania wartości.

CCS* jest niewielką modyfikacją oryginalnego CCS Milnera. Najistotniejsze różnice polegają na nieco odmiennym sposobie komunikacji (synchronizacji) procesów oraz na wprowadzeniu nowego operatora złożenia odwołującego ▶.

CCS* (podobnie jak CCS) najłatwiej opisać w terminach systemu przejść. System przejść jest zdefiniowany jako

$$T = \langle P, Act, \rightarrow \rangle$$

gdzie

P jest zbiorem termów

Act jest zbiorem akcji

$\rightarrow$ jest odwzorowaniem, które każdej akcji $a \in Act$ przyporządkowuje relację binarną $\xrightarrow{a} \subseteq P \times P$.

Zakładamy istnienie następujących, co najwyżej przeliczalnych zbiorów:

$Act = \{a, b, c, \dots\}$ - zbiór akcji

$Var = \{X, Y, Z, \dots\}$ - zbiór zmiennych procesowych

Zakładamy również istnienie wyróżnionego elementu $\tau \in Act$, który oznacza akcję wewnętrzną. Niech $Act' = Act \setminus \{\tau\}$.

Zbiór termów CCS* $P = \{p, q, r, \dots\}$ definiujemy indukcyjnie:

$$p ::= X \mid 0 \mid \alpha.p \mid \mu X.p \mid p \setminus U \mid p+p \mid p \parallel_U q \mid p \triangleright p$$

gdzie $X \in Var$, $U \in Act'$

Operator złożenia równoległego wymaga (inaczej, niż w klasycznym CCS) podania wprost zbioru U akcji synchronizujących. Akcja $\alpha \in U$ działa na oba składniki jednocześnie (synchronizuje je), akcja $\alpha \notin U$ działa na jeden ze składników. Nowością jest operator złożenia odwołującego $\triangleright$, będący kombinacją alternatywy $+$ i złożenia równoległego $\parallel_0$.

Mówimy, że zmienna procesowa X jest zmienna wolna w termie p , jeżeli w p występuje zmienna X nie związana operatorem rekursji $\mu X.q$. Term p jest procesem, jeżeli nie zawiera zmiennych wolnych. Oznaczmy zbiór procesów przez P_0 .

Poniżej zdefiniujemy semantykę operacyjną CCS^* przy pomocy relacji przejścia $\xrightarrow{\alpha} \subseteq P_0 \times P_0$. $p[q/r]$ oznacza term p , w którym wszystkie wystąpienia termu r zostały zastąpione przez q . Kreskę poziomą należy odczytywać jako symbol wnioskowania - jeżeli ... to ...

Relacja przejścia $\xrightarrow{\alpha}$ jest najmniejszą relacją spełniającą następujące warunki:

$$\alpha.p \xrightarrow{\alpha} p$$

$$p[\mu X.p/X] \xrightarrow{\alpha} q$$

$$\mu X.p \xrightarrow{\alpha} q$$

$$\frac{p \xrightarrow{\alpha} q}{p \setminus U \xrightarrow{\alpha} q \setminus U} \alpha \notin U$$

$$\frac{p \xrightarrow{\alpha} p'}{p+q \xrightarrow{\alpha} p'}$$

$$\frac{p \xrightarrow{\alpha} p'}{p \parallel_U q \xrightarrow{\alpha} p' \parallel_U q} \alpha \notin U$$

$$\frac{p \xrightarrow{\alpha} p' \quad q \xrightarrow{\alpha} q'}{p \parallel_U q \xrightarrow{\alpha} p' \parallel_U q'} \alpha \in U$$

$$\frac{p \xrightarrow{\alpha} q}{p \setminus U \xrightarrow{\tau} q \setminus U} \alpha \in U$$

$$\frac{q \xrightarrow{\alpha} q'}{p+q \xrightarrow{\alpha} q'}$$

$$\frac{q \xrightarrow{\alpha} q'}{p \parallel_U q \xrightarrow{\alpha} p \parallel_U q'} \alpha \notin U$$

$$\frac{p \xrightarrow{a} p'}{p \triangleright q \xrightarrow{a} p' \triangleright q}$$

$$\frac{q \xrightarrow{a} q'}{p \triangleright q \xrightarrow{a} q'}$$

3. JĘZYK LOGIKI MODALNEJ

Poniżej zdefiniujemy pewien język logiki modalnej, który posłuży nam do opisania niektórych własności termów CCS, oraz określimy semantykę tego języka. Zrobimy to w dwóch etapach. Najpierw zdefiniujemy nieco prostszy język L_1 , a następnie rozszerzymy go o dodatkowe formuły do języka L_2 .

3.1. Język L_1

Zbiór formuł $F_1 = \langle A, B, C, \dots \rangle$ definiujemy indukcyjnie następująco:

$$A ::= \text{Tr} \mid \text{Fal} \mid \text{AvA} \mid \text{AA} \mid \langle a \rangle A \mid [a]A$$

gdzie $a \in \text{Act}$.

Tr i Fal są jedynymi formułami atomowymi naszego języka. Bardziej złożone formuły można konstruować przy pomocy operatorów alternatywy $\vee$ i koniunkcji $\wedge$ oraz modalnych operatorów możliwości $\langle a \rangle$ i konieczności $[a]$.

Zauważmy, że nasz język nie zawiera negacji. Pozwala to na zbudowanie zwartego i przejrzystego systemu dowodowego.

Fakt, że formuła A opisuje pewną własność termu p , zaznaczamy pisząc $p \models A$ (p spełnia A). Relacja spełniania $\models \subseteq P_0 \times F_1$ jest zdefiniowana jako najmniejsza relacja spełniająca następujące warunki ($\models$ oznacza dopełnienie $\models$):

$$p \models \text{Tr} \quad p \not\models \text{Fal}$$

$$p \models \text{AvB} \quad \text{iff} \quad p \models A \quad \text{lub} \quad p \models B$$

$$p \models \text{AA} \quad \text{iff} \quad p \models A \quad \text{i} \quad p \models B$$

$$p \models \langle a \rangle A \quad \text{iff} \quad \exists q. (p \xrightarrow{a} q \quad \text{i} \quad q \models A)$$

$$p \models [a]A \quad \text{iff} \quad \forall q. (\text{jeżeli } p \xrightarrow{a} q \quad \text{to } q \models A)$$

Pierwsze cztery „klasyczne” warunki nie wymagają wyjaśnienia. Dwa operatory modalne $\langle a \rangle$ i $[a]$ są pochodnymi standardowych operatorów modalnych możliwości $\Diamond$ i konieczności $\Box$ (patrz [4], [2]). $\langle a \rangle A$ czytamy „możliwa akcja a , po której formuła A będzie spełniona” lub krócej „możliwa a , że A ”, natomiast $[a]A$ czytamy „po akcji a koniecznie będzie spełniona formuła A ” lub „po a koniecznie A ”.

Zauważmy, że jeżeli proces p nie może zareagować na akcję a , wówczas na mocy definicji $p \models [a]A$ dla dowolnej formuły A .

Dla przykładu

$p \models \langle a \rangle \text{Tr}$ znaczy „p może zareagować na akcję a”,

$p \models [a] \text{Fal}$ znaczy „p nie może zareagować na akcję a (gdyż żaden term nie spełnia Fal)”,

$p \models [a] \langle b \rangle \text{Tr}$ znaczy „jeżeli p zareaguje na a, to potem jest w stanie zareagować na b”.

3.2. Język L_2

Zauważmy, że w powyższym języku wystąpienie akcji wewnętrznej τ musimy jawnie specyfikować w formule modalnej. Np.

$$a.b.0 \setminus \langle a \rangle \models \langle \tau \rangle \langle b \rangle \text{Tr}$$

lecz

$$a.b.0 \setminus \langle a \rangle \not\models \langle b \rangle \text{Tr}$$

Intuicyjnie jednak akcja wewnętrzna jest akcją zachodzącą „spontanicznie”, poza kontrolą środowiska, i chcielibyśmy, aby formuły temporalne jakoś to odzwierciedlały. W tym celu rozszerzymy język L_1 o dwa nowe operatory modalne $\llbracket a \rrbracket$ i $\langle a \rangle$. A zatem zbiór formuł F_2 języka L_2 jest definiowany indukcyjnie

$$A ::= \text{Tr} \mid \text{Fal} \mid \text{AvA} \mid \text{A} \wedge \text{A} \mid \langle a \rangle A \mid [a]A \mid \langle a \rangle A \mid \llbracket a \rrbracket A$$

Zanim przedstawimy semantykę L_2 , musimy wprowadzić relację tzw. uogólnionego przejścia $\xrightarrow{a}$. Mamy $p \xrightarrow{a} q$ wtedy i tylko wtedy, gdy istnieje taki ciąg $p = p_0, p_1, \dots, p_m, q_0, q_1, \dots, q_n = q$, że

$$p_0 \xrightarrow{\tau} p_1 \xrightarrow{\tau} \dots \xrightarrow{\tau} p_m \xrightarrow{a} q_0 \xrightarrow{\tau} q_1 \xrightarrow{\tau} \dots \xrightarrow{\tau} q_n$$

Kolejna definicja dotyczy problemu nieskończonych pętli. Np. proces $\mu X. a. X \setminus \langle a \rangle$ może reagować na nieskończony ciąg akcji wewnętrznych τ , co dla obserwatora z zewnątrz będzie wyglądało jak „zapętlenie”. Mówimy, że p jest rozbieżny ze względu na a (piszemy $p \nVdash a$), jeżeli istnieje nieskończony ciąg $p = p_0, p_1, \dots, p_m, q_0, q_1, q_2, \dots$ taki, że

$$p_0 \xrightarrow{\tau} p_1 \xrightarrow{\tau} \dots \xrightarrow{\tau} p_m \xrightarrow{a} q_0 \xrightarrow{\tau} q_1 \xrightarrow{\tau} q_2 \xrightarrow{\tau} \dots$$

Mówimy, że p jest zbieżny ze względu na a ($p \vdash a$), jeżeli nieprawda, że $p \nVdash a$; p jest zbieżny ($p \vdash$), jeżeli $p \vdash a$ dla wszystkich $a \in \text{Act}$.

Teraz możemy już zdefiniować semantykę operatorów $\llbracket a \rrbracket$ i $\langle a \rangle$.

$$p \models \langle a \rangle A \text{ iff } \exists q. (p \xrightarrow{a} q \text{ i } q \models A)$$

$$p \models \llbracket a \rrbracket A \text{ iff } p \vdash a \text{ oraz } \forall q. (\text{jeżeli } p \xrightarrow{a} q \text{ to } q \models A)$$

Dla obu języków (L_1 i L_2) istnieją niesprzeczne i pełne systemy dowodowe, tzn. zbiory aksjomatów i reguł dowodzenia, pozwalające formalnie wykazać, że dany term spełnia daną formułę. Dla CCS systemy te

są przedstawione w [10], dla CCS* znajdują się w [3].

4. EKSPRESYWNOSĆ JĘZYKA

Interesującym i ważnym z punktu widzenia ewentualnych zastosowań zagadnieniem jest zakres własności, które dają się zapisać przy pomocy danego języka.

Najogólniej mówiąc własności systemów współbieżnych podzielić możemy na własności bezpieczeństwa („nie stanie się nic złego”) i własności żywotności („stanie się coś dobrego”). Przyjrzyjmy się, jak nasz język radzi sobie z obu typami własności.

Niech $p \in P_0$ oznacza proces, którego własności chcemy badać. Wprowadźmy oznaczenie $p \# U =_{df} p \setminus (Act' \setminus U)$.

Niech $d \in Act$ oznacza wystąpienie niepożądanego zdarzenia. Należy zapisać własność mówiącą o tym, że d nigdy nie wystąpi dla p , tzn. p nie może zareagować na żaden ciąg akcji zawierający d .

Odpowiednia formuła będzie brzmiała

$$p \# \langle d \rangle = \llbracket d \rrbracket Fal$$

Ponieważ żaden term nie spełnia Fal , więc formuła mówi nam, że p nie może zareagować na żaden skończony ciąg akcji, którego elementem jest d .

Bardziej złożone własności bezpieczeństwa mogą polegać np. na tym, że wystąpi co najwyżej jedna spośród dwóch akcji $d, e \in Act$. Odpowiednie formuły wyglądają następująco

$$p \# \langle d, e \rangle = \llbracket d \rrbracket \llbracket e \rrbracket Fal \wedge \llbracket e \rrbracket \llbracket d \rrbracket Fal$$

W ten sposób możemy rozwijać formuły dla dowolnych zbiorów akcji.

Podobnie ma się sprawa z formułowaniem własności żywotności. Niech $h \in Act$ oznacza wystąpienie zdarzenia pożądanego. Własność, że h wystąpi, zapisujemy

$$p \# \langle h \rangle = \llbracket \tau \rrbracket \langle h \rangle Tr$$

co oznacza, że p nie może zareagować na nieskończony ciąg akcji nie zawierający h oraz po każdym skończonym ciągu wystąpienie h jest możliwe.

Najprostsza własnością żywotności jest własność stopu, tzn. że program kiedyś się zatrzyma, co zapisujemy

$$p \setminus Act' = \llbracket \tau \rrbracket Tr$$

Możemy także zapisać własność relatywnej żywotności: jeżeli wystąpi g (np. komunikat), to potem wystąpi h (potwierdzenie), gdzie $g, h \in Act$

$$p \# \langle g, h \rangle = \llbracket g \rrbracket \langle h \rangle Tr$$

5. UWAGI KOŃCOWE

W pracy przedstawiono zarys opartego na logice modalnej języka

опису і систему доведення власності процесів CCS*.

Можна wyróżнити два напрю, на яких концентрувати ся повинны далше prace:

1° Розszerzenie języка о далше операторы (np. implikację, операторы temporalne) w celu zwiększenia jego ekspresywności

2° Uogólnienie systemu na wszystkie struktury występujące w LOTOSie.

LITERATURA

- [1] Barringer H., Kuiper R., Pnueli A., "Now You May Compose Temporal Logic Specifications", w *Proc. of the 16th ACM Symposium on Theory of Computing*, 1984, pp 51-63
- [2] Fierla A., *Logika Modalna i Temporalna - Przykłady Zastosowania do Specyfikacji Programów Współbieżnych*, Raport SPRA 3/88, C.O.PWr, Wrocław 1988
- [3] Fierla A., *Specyfikacja Programów Współbieżnych - Modalny System Dowodowy dla CCS**, Raport SPRA [w przygotowaniu]
- [4] Hughes G.E., Cresswell M.J., *An Introduction to Modal Logic*, Methuen & Co., London 1968
- [5] LOTOS - *A formal description technique based on the temporal ordering of observational behaviour*, pod red. Ed. Brinksmy, ISO 8807, 1988
- [6] Manna Z., Pnueli A., "Verification of Concurrent Programs, Part I: The Temporal Framework", w *The Correctness Problem in Computer Science* (pod red. S. Boyera i J.S. Moore'a), International Lecture Series in Computer Science, Academic Press, London 1981, pp 215-273
- [7] Manna Z., Pnueli A., "Verification of Concurrent Programs, Part II: Temporal Proof Principles", w *Proc. of the Workshop on Logics of Programs*, LNCS, Springer, Berlin 1981
- [8] Milner R., *A Calculus of Communicating Systems*, LNCS 92, Springer, Berlin 1980
- [9] Sławski Z., *Formalne Metody Specyfikacji i Konstrukcji Oprogramowania Sieciowego - Problemy i Założenia*, Raport SPRA 15/87, C.O.PWr, Wrocław 1987
- [10] Stirling C., "Modal Logics for Communicating Systems", w *Theoretical Computer Science*, vol.49, No 2/3, 1987
- [11] Winskel G., "A Complete Proof System for SCCS with Modal Assertions", w *Fundamenta Informaticae*, vol.9, No 4, 1986, pp 401-420

MODAL LOGIC LANGUAGE FOR COMMUNICATING SYSTEMS

An introduction to a modal language for specification of logical properties of network protocols is presented. A protocol should be given as a CCS* process. The syntax and the operational semantics of the language is described, a concept of proof system is introduced and a language expressiveness, i.e. the range of properties which can be expressed in it is shortly analysed.

ЯЗЫК МОДАЛЬНОЙ ЛОГИКИ ДЛЯ ВЗАИМОДЕЙСТВУЮЩИХ СИСТЕМ

Представляется введение в язык модальной логики для спецификации логических свойств сетевых протоколов. Протокол задается в виде процесса на расширенном языке исчисления взаимодействующих систем (CCS*). Приводится описание синтаксиса и операционной семантики языка, вводится концепция доказательств и анализируется выразительная сила языка.

Mariusz FLASIŃSKI*

allocation control, distributed systems

Łaszek KOTULSKI *

EGD-graph, edNLC-graph grammar

On the Linguistic Approach to Controlling a Distributed Software Allocation*

In the paper a problem of controlling the allocation of the software in the local area networks is considered. A family of attributed EDG-graphs appears to be sufficiently strong a tool for description of the distributed system. An attributed programmable edge-labelled directed node-label controlled graph grammar is used for controlling allocation process.

1. Introduction

During the past decade there has been a considerable growth of interest in problems of software systems for Local Area Networks. The interest is connected with the fact that the software which is necessary for supporting resource sharing in distributed environment is difficult to construct. Therefore, constructing such systems should be automatized as much as possible.

The research project COSID (Concurrent Software Implementation for Distributed Systems), allowing us to create, test and formally prove the software system for distributed environment, is being implemented in the Department of Computer Science, Jagiellonian University. The solution of the problem controlling an allocation phase of COSID is presented in this paper. Main ideas of COSID system are presented in chapter 2. We discuss the problem of controlling an allocation phase with the help of a graph grammar in chapter 3. We describe the allocation system for networks - ASNET in chapter 4, whereas chapter 5 includes concluding remarks.

2. Main ideas of COSID system

The creation of a distributed software in COSID system is made in two steps. In the first step (called a compilation phase) patterns of all

* Dept. of Computer Science, Jagiellonian University.

* This work was supported by Ministry of Education under RP. I.09 project.

the processes and managers are described in COLNET [6] language. Manager concept (which allows us to synchronize the work of a processes) is an extension of the monitor concept [1,4] for distributed environment and is discussed more precisely elsewhere [2].

In the second step all the needed modules (processes and managers) should be allocated in the computing nodes of a network and all the connections among these modules should be generated. This phase, called an allocation, is supported by ASNET system.

Thus, the ASNET helps us to create a required software-hardware structure of the network in a certain moment. The expression "a required software-hardware structure has two meanings in this context. First of all, it means a structure demanded by a user of the COSID system, who demands a distributed system. Secondly, it means a structure which can be created on the basis of physical resources of a network (for example size of a memory offered by computing nodes of a network) currently available.

In COSID system, a special family of graphs is used for representing a software-hardware structure of the network and a special kind of graph grammars (belonging to a family of edNLC grammars [5]) is used for controlling an allocation phase.

3. The control of an allocation phase with the help of edNLC-graph grammar.

A current structure of a network will be described by a family of attributed EDG-graphs, which has been introduced in [2] in the following way.

Definition 1

An attributed directed node- and edge-labelled graph over Σ and Γ (an attributed EDG-graph) is a seven-tuple:

$H = \langle V, E, \Sigma, \Gamma, \delta, \alpha, \beta \rangle$ where:

V is a finite, nonempty set of nodes,

Σ is a finite, nonempty set of node labels,

Γ is a finite, nonempty set of edge labels,

E is a set of edges of the form $\langle v, \mu, w \rangle$, in which $v, w \in V$ and $\mu \in \Gamma$,

$\delta: V \rightarrow \Sigma$ is a node labelling function,

$\alpha: V \rightarrow 2^A$ is a node attributing function in which A

is a finite set of node attributes,

$\beta = \{\beta_\gamma\}_{\gamma \in \Gamma}$, $\beta_\gamma: E_\gamma \rightarrow 2^B$ is a γ -edge attributing function in which B is a finite set of edge attributes.

Each node of a graph represents one component of a distributed system. Node labels, ascribed to nodes by the node labelling function, describe types of these components (computing nodes of a network, processes,

managers, entry procedures to managers e.t.c.). If a more precise characteristics of a component is needed, a set of node attributes may be ascribed to it with the help of the node attributing function. For example, we may ascribe attributes: "number of a computing node", "a size of a memory, which is currently offered by a computing node", to components labelled with "N" ("a computing node of the network").

Relations among these components are defined with directed edges connecting proper nodes of a graph. The set of edge labels is used for a description of a kind of each relation. For example, a node of a graph v_1 , representing a process (labelled with "P") may be connected with a node of a graph v_2 representing a computing node (labelled with "N") with a direct edge describing a relation "is allocated in" (labelled with "a"). It means that the edge (v_1, a, v_2) belongs to the set of edges E of a considered graph. We assume that the set of edge attributes β is an empty set because a characteristics of graph edges with the help only of edge labels is sufficient for our purposes.

Now we define a graph grammar, which is used for controlling an allocation phase as in [2].

Definition 2

An attributed programmable edge-labelled directed node-label controlled graph grammar (an attributed programmable edNLC - graph grammar) is a quintuple:

$$G = (\Sigma, \Gamma, P, Z, D), \text{ where:}$$

Σ is a finite nonempty set of node labels,

Γ is a finite nonempty set of edge labels,

Z is an initial attributed EDG-graph,

P is a finite set of productions of the form

$(L, R, \mathbb{W}, C, F)$ in which:

L is an EDG-graph of a left-hand side of production

R is an EDG-graph of a right-hand side of production

$\mathbb{W}$ is a predicate of the production applicability,

$C: \Gamma \times \{in, out\} \rightarrow 2^{\Sigma \times \Sigma \times \Gamma \times \{in, out\}}$

is an embedding transformation,

F is a set of functions which ascribe attributes to nodes of R ,

D is a control diagram.

Components L, R and C of an edNLC-production are standard components of graph grammar production and are discussed more precisely elsewhere [5]. Let us only mention, that an embedding transformation allows us to embed a graph of right hand side of production in a transformed graph during a derivation step. The predicate of a production applicability $\mathbb{W}$ is a conjunction of logical conditions defined for each production. For example

it may be defined as follows: the size of memory demanded by a process, which should be allocated in a computing node is less than the size of a memory offered by this computing node.

Let us discuss a mechanism of controlling an allocation, used in an allocation control module ALLCONT (being a part of ASNET system), with the help of a grammar defined above. The network-state description file is an attributed EDG-graph. Each allocation command corresponds to some production of the attributed edNLC graph grammar. The possibility of an application of a demanded production is checked by the control diagram.

The control diagram is a non-attributed graph, which nodes are labelled with number of productions (and there are two distinguished nodes: the initial node and the final node). Its edges are labelled with "Y" (Yes) or "N" (No). If we are situated at the point labelled with number of a production, we try to employ this production. If the graph of the left-hand side $-L$ is identified as a subgraph of the transformed graph and the predicate of applicability of this production $\mathbb{T}=\text{TRUE}$, then:

- 1) we apply this production,
- 2) we go to the node pointed by the edge labelled with "Y".

Otherwise we go to the node pointed by the edge labelled with "N".

Let us return to our discussion. Thus, if the application of the production is not possible then ALLCONT informs the user about the causes of this fact (the system gives an optional suggestion in some cases). In the other case the production is applied what causes bringing up to date of the net-state description file and giving a proper allocation extracode.

4. Concluding remarks

As we have mentioned, the allocation system for networks ASNET has been implemented on the basis of a formalism presented above.

The main module of ASNET, which controls the allocation phase - ALLCONT has occurred to be very effective. It results mainly from the fact that we have applied derivation procedures of the graph grammar parser GRAPHPARS [3] as a basic skeleton of ALLCONT system.

The second module of ASNET is a compiler of the allocation language LANGALL2. The third one - GRAPHNET - allows the user to see a concurrent situation in a distributed system which is of the graphic output data form. There is a possibility of giving some allocation commands in a graphic form in GRAPHNET, as well. Kernel of the COSID system performs extracodes of an allocation given by ALLCONT module. ENVDESC is module which links allocation subsystem with the compilation subsystem. This idea is described in more detail in [2].

As we have shown graphs seem to be the most natural formalism for a

description of a software-hardware structure of a network. On the other hand the graph grammar discussed in this paper is a strong formalism [5] and algorithms based on it are very effective [2,3]

5. References

1. Brinch Hansen P.: Architecture of concurrent programs. Prentice Hall, Inc., Englewood Cliffs, New Jersey 1977.
2. Flasiński, Kotulski L.: On the use of graph grammars for the control of a distributed software allocation. Accepted for publication in Computer Journal.
3. Flasiński M.: Parsing of edNLC-graph grammars for scene analysis. Pattern Recognition, vol. 21, no 6, 1988.
4. Hoare C. A. R.: Monitors: an operating systems structuring concept. Comm of A.C.M. 1974, 17, 10, pp. 549-557.
5. Janssens D., Rozenberg G., Verraedt R.: On sequential and parallel node-rewriting graph grammars. Computer Graphics and Image Processing 18, pp. 279-304, 1982.
6. Kotulski L.: A two level approach to the implementation LANs software. - in preparation -

Streszczenie.

W artykule rozważano problem kontroli alokacji oprogramowania w lokalnych sieciach komputerowych. Rodzina atrybutowanych EDG-grafów wydaje się być wystarczająco mocnym narzędziem do opisu systemu rozproszonego. Atrybutowana programowalna gramatyka grafowa generująca grafy o zaetykietowanych i skierowanych krawędziach, której transformacja osadzenia jest sterowana etykietami wierzchołków, jest używana do kontroli przebiegu alokacji.

Содержание.

В работе рассуждается проблема управления розприделения прогпамм в местных вычислительных сетях. Представляется, что атрибутивные ЕДГ-графы являются достаточно сильным формализмом чтобы описывать рассеянные системы. Целью управления розприделения программ использовано атрибутивную программированную графовую гпамматику генерирующую направленные и реберно этикетированные графы, которой трансформация осаднения управляетца этикетами углоб.

computer network,
DNA, DECnet, VAX

Jacek Gruber*

VAX NETWORKING AND COMMUNICATION SOFTWARE - - GENERAL OVERVIEW

DECnet is a family of software and hardware communications products that enable Digital Equipment Corporation operating systems to participate in a network environment. Overview of Digital's Network Architecture, VAX Networking Product Set and types of networks are briefly presented. Four main kinds of Digitals' network connections and a basic information of implementation of DECnet-VAX software and hardware are described in the paper. Short characteristic of VAX Packetnet System Interface networking system, several types of network access control, network services and performing of network operations in DECnet-VAX are done in the article, too.

1. Introduction

Digital Equipment Corporation (DEC) offers a range of products to link processes or tasks running under VMS operating system, other Digital operating systems or other manufacturer's systems as a communication software system [4,5,8]. It can be configured on all VMS operating systems (including Micro-VMS operating system). The components of networking software are: DECnet-VAX software products (VAX to other Digital Systems), DECnet-VAX Internet software products (VAX to other manufacturer's systems) and VAX PSI (Packetnet System Interface) networking software system (VAX to public Packet Switched Data Network-PSDN). Phase IV DECnet [6,8] supports large and small networks with single and multiple area networks. In a single area network, a maximum of 1023 nodes is possible. In a multiple area network, as many as 63 subnetworks (single area networks).

2. Overview of Digital's Network Architecture

The Digital Network Architecture, DNA [1,4,8], is the framework for all DEC Communications products. DECnet, DECnet/SNA Gateway (SNA-IBM System Network Architecture), Communications servers are among many of these products. DNA (Table 1) is based on the architectural model for open systems interconnection OSI/ISO [3].

There are basically two types of networking environments (Table 2): those involving communications Digital-Digital systems and between Digital systems and other vendor's systems (Digital-to-non-Digital). Both local and remote communications are possible.

The incorporation of the Ethernet [2] local area network technology into DNA Phase IV - provides a complete set of products to address local area networking needs. Ethernet offers remote or long-haul connections and local connections as one integral network.

3. Networks Components

A node is an operating system that uses DECnet software to communicate with other operating systems across a network. A VMS node uses DECnet-VAX software to communicate with other DECnet-VAX nodes and with other Digital operating systems that support DECnet.

* Computer Centre, Technical University of Wrocław, Wrocław, Poland

Table 1 Architectural Layers of DNA and Their Associated Functions (related to OSI/ISO model)

OSI/ISO LAYERS	DECnet Function's	DNA Layers	DNA Protocols
	File Access	User	User Protocols
Application	Command Terminals	N E T	Data Access Protocol (DAP) and others
Presentation	Host Services		
Session	Network Control		
	Program-to-program	M A N A G E M E N T	Session Control Protocol
Transport			END Communication
Network	Adaptive Routing		Routing Protocol
Data Link	Host Services		Data Link
Physical	Packet Transmission/Reception		Physical Link
			DDCMP* sync async E T H E R N E T
			CI X.25

*Digital Data Communication Message Protocol

Table 2 Digital's VAX Networking Product Set

Type of Communication	VAX Software Communications Products	LAN Communications Subsystem Servers
Digital-to-Digital Host Communication Local Traditional	All DECnet Software	N/A
LAN	DECnet-VAX	DECnet Router/ DECnet Router/ X.25 Gateway
REMOTE	All DECnet Software	DECnet Router/ DECnet Router/ X.25 Gateway
Digital-to-non-Digital Host Communications	All Internets VAX PSI*	DECnet/SNA Gateway, DECnet Router/ X.25 Gateway
Terminal Connection	CX/DX/AX poly-COM**	Terminal Server

*Packetnet System Interface

**see [1.4]

A VMS operating system uses DECnet-VAX Internet products to communicate with other manufactures operating systems and VAX PSI software products to communicate with other DEC operating systems via a Packet Switching Data Network.

A node, called host node provides services for another node. A local node is the VMS operating system your user account resides on. Other

nodes in the same network the local node is attached are called remote nodes. Nodes that accept and forward messages intended for other nodes in a network are called routing (full- routing) nodes. Nodes that only accept messages are called end nodes (non-routing). If VAX PSI software in multi-host configuration is installed on the local node and a user is on an Ethernet, the user can configure its' node to serve as a connector node. This is a gateway to a PSDN for host nodes on an Ethernet. It is possible use on node on such multi-host Ethernet connector the node to access a PSDN without being configured as a DTE, provided have VAX PSI Access software installed.

A VMS operating system connected to a PSDN, with VAX PSI software installed, can function as a DTE. To configure a local DTE it is necessary installed a X.25 protocol module.

The X.25 protocol module is a software component that controls the transmission of data packets over PSDN.

Types of DECnet Nodes. DECnet supports a variety of types of nodes developed during different phases of DNA implementation [1]. The following types of nodes can be configured as being adjacent to each other on the network: Phase II/Phase II, Phase II/Phase III, Phase III/Phase III, Phase III/Phase IV, Phase IV/Phase V. Phase II nodes can communicate with each other as long as there is a physical data link between the nodes. They support only point-to-point connections without support for Ethernet. With Phase III, DECnet introduced adaptive routing, which allows a reasonably large number of nodes to communicate conveniently without support for Ethernet. Phase IV DECnet permits the configuration of very large networks and expands the types of data links available for use. Phase IV software supports Ethernet circuits, as well as DDCMP (Digital Data Communication Message Protocol), CI (Computer Interconnect bus) and X.25 circuits. The nodes of Phase IV can be one of two kinds: routing nodes (routers) or end nodes (nonrouters).

Other routing and nonrouting nodes are: Phase III routers deliver packets to and receive packets from other nodes, and route packets from other destination nodes whose addresses are less than 256. They use DDCMP, X.25, and CI circuits, but do not support Ethernet circuits. Phase II nodes can send packets to adjacent PHASE III routers or to other adjacent Phase II nodes. However, Phase II nodes can send packets only in point-to-point configurations. Phase III nodes can not communicate with a Phase II node through another Phase III node.

Circuits. Circuits are high-level communications data paths over physical lines between nodes or DTEs (Data Terminal Equipments). DECnet-VAX employs four classes of circuits [1,6]: DDCMP circuits, CI circuits, Ethernet and X.25 circuits. DDCMP circuits provide the logical, point-to-point, or multi-point connection between two or more nodes. A point-to-point circuit operates over a corresponding synchronous or asynchronous DDCMP point-to-point line. Multi-point control circuits operate over synchronous DDCMP control lines. CI circuits are available only on those node which are attached to a CI-780 or CI-750 Interconnect. CI circuits are similar in many ways to DDCMP multi-point circuits, but use their own protocol. Ethernet circuits provide for a multi-access connection. X.25 circuits use the level 3 protocol (the packet level), and provide for communication over PSDN. The circuits are made available to PSI software [1], or DECnet data link mapping (DLM) [1,6]. DLM permits the use of X.25 as a data link through the mapping of data link information between the DECnet routing layer and the X.25 native circuits and X.25 DLM circuits.

4. VAX/VMS communication software: DECnet-VAX and VAX PSI

DECnet-VAX is the implementation of the DECnet which causes a VMS

operating system to function as a network node. As the VMS network interface, DECnet-VAX supports both the protocols necessary for communicating over the network and the functions necessary for configuring, controlling, and monitoring the network.

Under DECnet-VAX, Digital's Internet family of products supports the interconnection of Digital's computers and networks to systems built by other manufactures. The Internet are tools for distributed data processing in a multi-vendor environment. They provide transparent communications with the equipment of other vendors, and at the same time, offer the flexibility of local file systems [i.e.5].

VAX PSI (Packetnet System Interface) is the software product that allows the VAX/VMS user to participate in a packet switching environment. VAX PSI implements the X.25 and X.29 recommendations providing a user interface to a Packet Switching Data Network.

By using it is possible to communicate DECnet nodes across a PSDN through DLM. Another way in which we can communicate with a remote node over PSDN is through and X.25 multi-host connector mode with VAX PSI software in the multi-host VAX/VMS node. The host node services of the connector node is a VAX/VMS system configured with VAX Access software (Ethernet gateway, without being a DTE).

Alternatively, a VAX/VMS node can use a server-based X.25 gateway node to communicate over a PSDN, (as a local DTE) provided VAX XEP [5] software is installed on the node. Both calls, X.25 and X.29 can be received by host nodes.

5. DECnet-VAX configurations

DECnet software products manufactured by DEC offers possibilities to build wide range of Digital's computer network configurations. Digital Network Architecture (DNA) is a very flexible architecture and allows computers from different manufactures to work together in a common computer networks with the efficient DECnet network operations utilization.

DECnet supports the following network connections:

- a connection to an Ethernet circuit in a local area network configuration,
- point-to-point or multi-point connections to a node running under DECnet using the DDCMP [6],
- a connection over the Computer Interconnect [6] to another node running under DECnet software,
- an X.25 connection to a node running DECnet (either a direct connection over PSDN or a connection made by means of an X.25 multi-host connector node that access a PSDN).

Figure 1 is a simple DECnet-VAX Phase IV [1,8] configuration illustrating various DECnet-VAX nodes, that is, Micro VMS end nodes, VMS routers, and VMS end nodes in a VAX cluster (see below) connected to an Ethernet, and two Ethernets connected by routers. This figure also shows the use of a DDCMP synchronous line to connect a router to additional DECnet.

DECnet-VAX Ethernet Local Area Network (LANS) configuration. Ethernet circuit devices supported by DECnet are DEUNA (VMS) and DEUNA (micro VMS) intelligent high-performance synchronous interfaces, that mount on the VAX (and PDP-11) processor UNIBUS or on the Micro-VAX processor [7].

DDCMP point-to-point and multi-point network configurations. Digital Data Communication Message Protocol (DDCMP) provides a low-level communications path between systems. DDCMP performs the basic communications function of moving information blocks over an unreliable communications functions channel. DDCMP is also used to manage the orderly transmission and reception of blocks on channels with one or more transmitters and receivers.

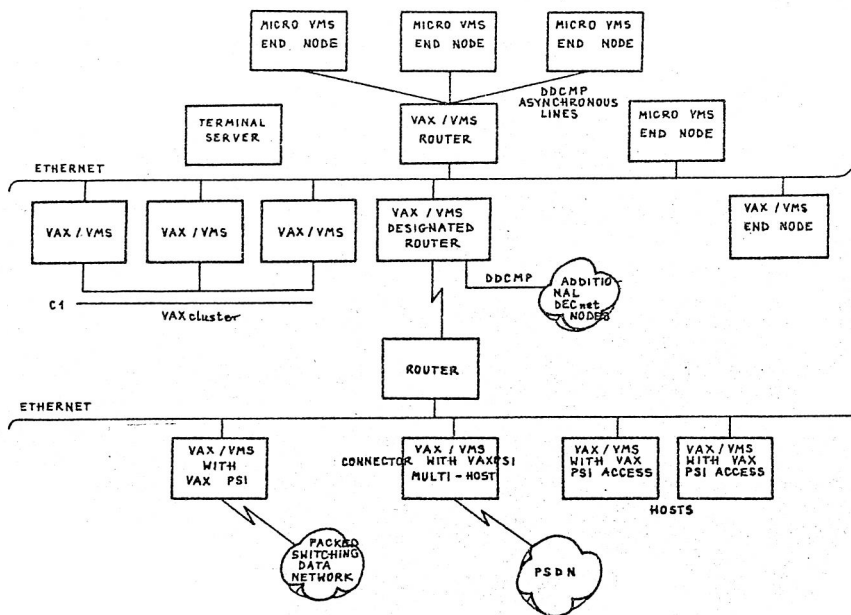


Fig. 1 Simpla DECnet Phase IV configuration.

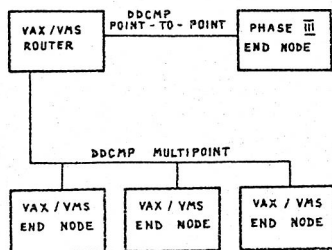


Fig. 2. Typical DDCMP point-to-point connections.

Figure 2 illustrates DDCMP point-to-point and multi-point configuration. A point-to-point configuration consists of two systems connected by a single communication channel. A multi-point configuration consists of two or more systems connected by the communications channel, called the control station. All other the systems are known as "tributaries".

DECnet-VAX configurations for VAXclusters. VAXcluster is an organization of VAX systems which communicate over a high-speed communications path, the CI (Computer Interconnect bus), and share processor resources as well as disk storage. The CI physical links from the individual nodes in the VAX cluster are connected to a Star Coupler [7]. DECnet-VAX connections are required for all VAX/VMS systems in the VAX cluster.

X.25 network configurations. Figure 3 show a typical DECnet network configuration of X.25 connections. Packet switching data network provides fast, dependable communications between geographically

distributed nodes. Two of CCITT series of recommendations (for standard communication protocols) could be used by PSDN and other common carriers to provide data communications products. X.25 (three levels protocol) and X.29 (PAD) interface definitions between the computer and the network and the definition of control of asynchronous terminals connected directly to a network define, together, the ITI (Interactive Terminal Interface).

6. DECnet-VAX network services

Objects. Objects provide known general-purpose network services [1,8]. An object is identified by object type, which is a discrete numeric identifier for either a user task or a DECnet program such as NML or FAL (see below). The DECnet network software uses object type numbers to enable logical link communications with NSP (Network Service Protocol). There are two general types of DECnet-VAX objects. Objects with a @ object value are usually user defined images for special-purpose application. Non zero objects are known objects that provide a specific network services such a file access (FAL) or network management (NML). They may also be user-defined tasks; these objects should be for user-supplied, known services.

The following DEC-supplied objects are defined inside the network [1,4,5]: File Access Listener (FAL), Data Access Protocol (DAP), Network Management Listener (NML), Event Logger (EVL), Loopback Mirror (MIRROR), DECnet Test Receiver (DTR), DECnet Test Sender (DTS), Mail, Phone.

Host Loader (HLD) provides downline task-loading support for RSX-11S tasks. VAX PSI object contain a command procedure that is initiated when the incoming call arrives, and do account for the incoming call.

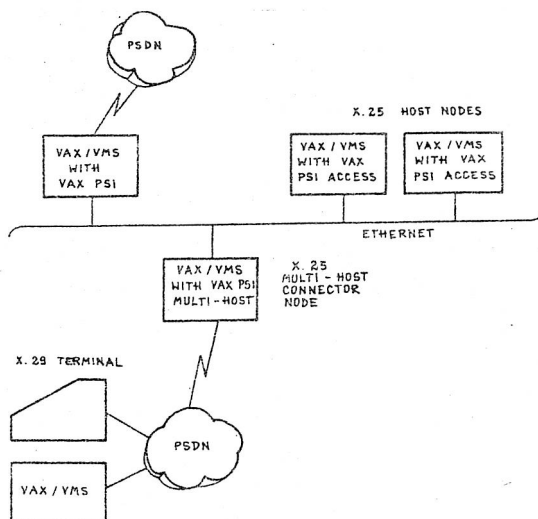


Fig.3. DECnet network configuration showing examples of X.25 connections.

Network access control. DECnet-VAX regulates access to the network at various levels. Routing-Initialization Passwords - Whenever a communications circuit is established, the local node in the circuit attempts to initialize with the DECnet software at the remote node connection for the circuit. System Level Access Control - DECnet-VAX provides system-level access control over the logical link connections.

The network user on the initiating node may explicitly supply an access control string to control which account is used on the remote node. Optionally DECnet provides default access control, when sending the request to the remote node and for incoming logical links.

Performing network operations in DECnet-VAX software. DECnet-VAX software offers user a variety of operations that can be accomplished over the network [1,4,5]. Three of which are:

- a. manipulating files on remote nodes,
- b. access remote files at the record level,
- c. perform task-to-task (interprocess) communications.

VMS operating system and DECnet-VAX communications software are integrated to provide a high degree of transparency for user operations [4,6]. For some applications however, it is desirable to have more direct access to network-specific information and operations. For this purpose, DECnet-VAX provide nontransparent communication [5,6].

Designing User Application for Network Operations. DECnet-VAX supports several programming languages for design of network applications. With any of languages, it is possible to access remote files and create tasks that exchange information across the network. The normal use of the programming languages for specific network operations that can be performed with DECnet-VAX, is summarized in table 3.

Table 3 Access Levels.

Access Levels			
User Languages	Network Operation	Language Calls	Access Level
DCL	Network command terminals Remote file manipulation Task-to-task communication	DCL commands	Transparent network access via DCL
Higher-Level	Remote file access (file and records)	Higher-level language I/O statements	Transparent network access via RMS
Macro or higher-level	Remote file access (file and records) Task-to-task communication	RMS** service calls	
MACRO or higher-level	Task-to-task communication	System service calls	Transparent and nontransparent network access via QIO*

*Queue Input/Output driver [6]

**Record Management Services [6]

7. Remarks

DNA architecture is very flexible, but is not closely based on the OSI/ISO [2] model. DECnet-VAX is the implementation of DECnet which causes a VMS operating system to function as a network node. As the VMS network interface, DECnet-VAX supports both the protocols necessary for communicating over the network and the functions necessary for configuring, controlling, and monitoring the network. DECnet-VAX software allows user very flexible and effective possibilities for design of network applications.

References

- [1] Guide to Networking on VAX/VMS, Order Number: AA-Y512B-TE, DEC, July 1985
- [2] IEEE P802.1-1982, Local Area Network Standards (ISO DIS 8802/1, ANSI P 802.1)
- [3] Information Processing Systems-Open Systems Interconnection, Reference Model, ISO/IS 7498, 1983
- [4] Networks and Communications Buyer's Guide, DEC, 1987
- [5] Networks Guidebook, DEC, 1986
- [6] Networks Handbook, DEC, 1985
- [7] VAX/VMS Hardware, VMS Hardware Handbook, DEC, 1986
- [8] VAX/VMS Software, VMS System Software Handbook, DEC, 1986

SIEC DECnet KOMPUTEROW VAX FIRMY DIGITAL EQUIPMENT CORPORATION

Przedstawiono architekturę DNA (Digital Network Architecture) sieci DECnet i porównano ją z architekturą OSI/ISO sieci komputerowych. Dokonano przeglądu sprzętu i oprogramowania stosowanego do budowy sieci DECnet-VAX oraz struktur topologicznych tej sieci. Scharakteryzowano system PSI umożliwiający wykorzystywanie publicznej sieci X.25 jako podsieci komunikacyjnej komputerów VAX z oprogramowaniem DECnet. Scharakteryzowano operacje, usługi sieciowe i zestawiono możliwości projektowania aplikacji sieciowych.

КОМПЬЮТЕРНАЯ СЕТЬ DECnet ФИРМЫ DEC

В статье представлена архитектура DNA сети DECnet и сравнено её с архитектурой OSI/ISO компьютерных сетей. Совершен просмотр оборудования и программного обеспечения применяемого при построении сети DECnet-VAX, а также топологических структур этой сети. Дана характеристика системы PSI дающей возможность использования общедоступной сети X.25 в качестве коммуникационной подсети компьютеров VAX с программным обеспечением DECnet. Дана характеристика операций и сетевых услуг, а также представлены возможности проектирования сетевых приложений.

Jacek GRUBER*

Leszek Z. MISIURA*

Lokalna sieć komputerowa,
rozproszony system przetwarzania
danych, badania czasowe

**BADANIA PRZYDATNOŚCI LOKALNEJ SIECI MIKROKOMPUTEROWEJ
10NET W SYSTEMIE ROZPROSZONEGO PRZETWARZANIA DANYCH 'KOMPAN'**

W pracy przedstawiono ilościową ocenę własności lokalnej sieci komputerowej 10NET mikrokomputerów profesjonalnych zgodnych z IBM PC XT/AT pod kątem jej zastosowania w systemie KOMPAN. Rozważono zarówno rozwiązania komunikacji realizowane sprzętowo (metoda dostępu do sieci) jak i programowo (sieciowe mechanizmy transmisji).

1. WSTĘP

Lokalne sieci komputerowe (LAN) umożliwiają bardziej efektywne wykorzystanie zasobów mikrokomputerów, stanowiąc system rozproszonego przetwarzania danych. Przykładem ciekawego profesjonalnego wykorzystania sieci LAN jest rozproszony system Komputerowego Projektowania Automatyki Energetycznej (KOMPAN) budowany w Centrum Naukowo-Produkcyjnym Automatyki Energetycznej we Wrocławiu [6].

KOMPAN został zaprojektowany jako otwarty system wspomagania projektowania systemów sterowania procesami technologicznymi we wtórnym obiegu elektrowni jądrowej oraz automatyki elektrowni konwencjonalnych. Celem systemu KOMPAN jest :

- wspomaganie opracowania projektu technicznego,
- aktualizacja projektu technicznego w fazie opracowywania dokumentacji powykonawczej,
- wspomaganie kompletacji wyposażenia.

Biorąc pod uwagę stawiane systemowi wymagania, tj.:

- odpowiednio duża liczba stanowisk pracy użytkowników,
- mały czas oczekiwania na fizyczny dostęp do zasobów systemu,

*Centrum Obliczeniowe, Politechnika Wrocławska

- duża niezawodność funkcjonowania i odpowiedni system zabezpieczeń zasobów przy jednoczesnej dużej elastyczności ich udostępniania użytkownikom.

jako środowisko pracy systemu KOMPAN wybrano lokalną sieć mikrokomputerową Cx-NET firmy Computex [7]. Jest to kopia sieci 10-NET firmy Fox Research (USA). Stanowiskami systemu (stacjami sieci) są mikrokomputery personalne zgodne z IBM PC XT/AT.

W pracy dokonano oceny trafności wyboru sieci lokalnej mikrokomputerów jako technicznego środka obliczeniowego wykorzystywanego w systemie KOMPAN. Jej przedmiotem jest ilościowa ocena funkcjonowania sieci Cx-NET. W tym celu przeprowadzono eksperymenty na analitycznym modelu formalnym sieci. Uzyskane wyniki stanowią zbiór zaleceń dotyczących optymalnej alokacji komponent i zasobów systemu KOMPAN w środowisku sieciowym.

2. CHARAKTERYSTYKA SYSTEMU KOMPAN

System KOMPAN składa się z 18 podsystemów specjalizowanych, programu nagłówkowego oraz podsystemów wspomaganie planowania dostaw [8]. Otoczenie programowe systemu w środowisku sieciowym, stanowią :

- Centralna Baza Danych (CBD) wraz z procedurami dostępu,
- programy narzędziowe,
- systemy operacyjne stacji,
- prywatne zbiory danych.

KOMPAN jest przeznaczony do realizacji dwóch podstawowych klas zadań: projektowych i administracyjnych. Zadania te mogą realizować użytkownicy następujących kategorii: projektant, asystent, kreślarz, sekretarka, zaopatrzeniowiec, ekonomista, kierownik (pracowni projektowej, Biura Generalnych Dostaw), programista/informatyk. Dla użytkowników poszczególnych kategorii określono sposób korzystania z usług sieci, w tym z zasobów CBD.

W systemie KOMPAN można wyróżnić dwa podstawowe tryby przetwarzania informacji:

- tryb lokalny; przetwarzane są dane zgromadzone na dyskach twardych własnej stacji,
- tryb zdalny; zbiory danych są sprowadzane poprzez sieć z CBD, przetwarzane, a następnie odsyłane do CBD oraz wyprowadzane (lub nie) na urządzenia zewnętrzne (np. drukarki, plottery).

W sieci jako środowisku systemu KOMPAN pracują dwa typy stacji:

- a) stacje centralne (ang. superstations) - przechowują CBD oraz posiadają zasoby sprzętowe i programowe udostępniane w trybie zdalnym innym stacjom. W trybie lokalnym mogą być jedynie

stanowiskami pracy projektanta.

- b) stacje robocze (ang. workstations) - nie posiadają zasobów udostępnianych innym stacjom, a korzystają w trybie zdalnym z zasobów stacji centralnych. Są typowymi stanowiskami pracy projektantów i administracji. Dla tych stacji istnieje możliwość pracy w trybie lokalnym.

Wyposażenie stacji poszczególnych typów przedstawiono szczegółowo w [4].

3. BADANIA MODELOWE SIECI CX-NET

W celu oceny przydatności sieci Cx-NET jako środowiska systemu KOMPAN wykonano eksperymenty:

- 1) Weryfikujące parametry techniczne sieci Cx-NET.
- 2) Wyznaczające parametry eksploatacyjne systemu.
- 3) Pozwalające ustalić optymalne konfiguracje systemu w kolejnych etapach jego rozbudowy.

Eksperymenty pierwszej grupy polegały na wykonaniu szeregu obliczeń bazujących na zasadach przesyłania danych w sieciach lokalnych z dostępem CSMA/CD [2,3].

Drugą i trzecią grupę eksperymentów wykonano za pomocą analitycznej metody przybliżonej opisanej w [1,5]. Dane do tych eksperymentów przyjęto z założeń systemu KOMPAN dotyczących: liczby i typu stanowisk pracy, rodzaju przesyłanych siecią danych (transmisje interakcyjne i transmisje zbiorów) i częstości ich przekazywania pomiędzy stacjami, rodzaju wykorzystywanych usług sieciowych itp. [6].

Wszystkie eksperymenty wyznaczające parametry eksploatacyjne systemu zostały wykonane, dla przewidywanej jako końcowa wersji systemu w środowisku sieciowym o pokazanej na rys. 3.1 konfiguracji, w warunkach największego obciążenia sieci, tzn. gdy siecią jest przesyłanych dużo zbiorów w krótkich odstępach czasu.

3.1. Omówienie badań

W punkcie tym podano jedynie krótką charakterystykę przeprowadzonych badań, natomiast dokładny opis poszczególnych eksperymentów, uzyskane wyniki oraz ich ocena znajduje się w [4].

Parametry techniczne sieci

Zweryfikowano dwa podstawowe parametry techniczne sieci Cx-NET: szybkość transmisji równą 1 Mb/s oraz długość ramki równą 512 bajtów. W tym celu wyznaczono obciążenie sieci przy transmisji zbiorów różnej długości (od 1 kB do 512 kB). Jako kryterium poprawnego doboru wartości parametru przyjęto, że obciążenie sieci nie powinno przekraczać 20% jej

teoretycznej przepustowości.

Na podstawie uzyskanych wyników stwierdzono, że obowiązujące w sieci Cx-NET: szybkość transmisji i długość ramki gwarantują jej poprawną pracę w systemie KOMPAN.

Parametry eksploatacyjne KOMPAN'a

Tą grupę eksperymentów przeprowadzono w celu oszacowania maksymalnej intensywności zgłoszeń użytkowników po usługi sieciowe oraz ustalenia dopuszczalnej długości zbiorów przesyłanych przez sieć.

Maksymalna intensywność zgłoszeń użytkowników została wyznaczona w zależności od przepustowości i opóźnienia tranzytowego sieci, otrzymanych przy transmisji zbiorów o różnych długościach. Na podstawie uzyskanych wyników (rys. 3.2, 3.3) ustalono, jako obligatoryjną dla dalszych eksperymentów, intensywność zgłoszeń, przy której nie są przekroczone obie miary (tabela 3.1).

Dopuszczalną długość zbiorów przesyłanych przez różnych użytkowników wyznaczono w oparciu o przepustowość (rys. 3.4). Stwierdzono, że przy zakładanej konfiguracji sprzętowo - funkcjonalnej KOMPAN'a (4 stacje centralne, 8 stanowisk projektowych, 18 stanowisk administracyjnych) dopuszczalne długości zbiorów wynoszą odpowiednio:

- 512 kB dla stanowiska projektowego,
- 16 kB dla stanowiska administracyjnego,

co w pełni odpowiada potrzebom tego systemu [6].

Konfiguracje systemu KOMPAN

Ponieważ konfiguracja sprzętowa sieci ma być rozwijana równolegle z rozbudową zasobów programowych i CBD systemu KOMPAN, przeprowadzono dodatkowe badania ustalające optymalne konfiguracje systemu na poszczególnych etapach jego rozbudowy. Dla wykonania tych badań przyjęto z [6], że:

- sesja trwa średnio 30 minut,
- na stanowiskach projektowych są przesyłane do/z bazy danych zbiory o długości 512 kB (dwa razy w ciągu sesji), a na administracyjnych o długości 16 kB (5 razy w ciągu sesji),
- pozostałe transmisje (poczta elektroniczna, biuletyny itp.) wymagają ok. 10 % czasu wykorzystania sieci przez poszczególne stanowiska.

Po wyznaczeniu czasów transmisji zbiorów na różnych stanowiskach, przy założonym rozmieszczeniu CBD, uzyskano ilość stanowisk projektowych i administracyjnych w zależności od liczby funkcjonujących stacji centralnych (tabela 3.2).

Wymagania na pamięć dyskową

Przewidywane zapotrzebowanie na pamięć dyskową systemu KOMPAN wynosi ok. 170 MB, z czego 130 MB dla przechowywania 'projektowej' bazy danych, a ok. 40 MB dla bazy 'administracyjnej' [6]. Przeprowadzone wcześniej eksperymenty pozwoliły ustalić rozmieszczenie baz danych w stacjach centralnych, dzięki czemu uzyskano wymagania na pojemność dysków twardych dla tych stacji. Okazało się, że aby zrealizować zapotrzebowanie systemu na pamięć masową z zalecanym rozmieszczeniem CBD należy posiadać dwa dyski po 80 MB każdy oraz jeden 40 Mbajtowy.

4. PODSUMOWANIE

Badania analityczne sieci lokalnej Cx-NET pozwoliły w pełni ocenić jakość jej funkcjonowania w warunkach rozproszonego systemu przetwarzania danych KOMPAN.

Uzyskane z eksperymentów wyniki potwierdziły trafność wyboru tej sieci dla potrzeb KOMPAN'a. Spełnia ona wszystkie stawiane przez projektantów systemu wymagania w sposób zadowalający.

Literatura

- [1] Błach L., Lewoc J.B., Misiura L.Z.: Throughput Approximation for Local Area Networks, 8th International Conference on Systems Engineering, Coventry, pp 113-119, 1988.
- [2] Hammond J.L., O'Reilly P.J.P.: Performance Analysis of Local Computer Networks, Addison-Wesley Publ. Co., 1986.
- [3] Hopper A., Temple S., Williamson R.: Local Area Network Design, Addison-Wesley Publ. Co., 1986.
- [4] Jeleniewski T., Gruber J., Misiura L.Z.: Badania funkcjonowania sieci lokalnej mikrokomputerów Cx-NET w systemie KOMPAN, CPBR 5.1, kierunek nr 3: Energetyka jądrowa, CNPAE "IASIE" nr 88005, Wrocław, 1988.
- [5] Lewoc J.B., Misiura L.Z.: Analiza przepustowości w sieciach lokalnych z transmisją zbiorów, Sieci Komputerowe: Teoria, technika, zastosowanie, Prace naukowe Centrum Obliczeniowego Politechniki Wrocławskiej, Seria 3, nr 5, Wrocław, 1987.
- [6] Mikołajczyk Z.: Mikrokomputerowy system wspomagania projektowania systemów automatyki elektrowni jądrowej, CNPAE Wrocław, nr ew. 86001, Wrocław, 1986.
- [7] Opis sieci Cx-NET, Computex co. Ltd., Warszawa, 1987.

STUDIES OF FUNCTIONAL PROPERTIES OF THE 10NET MICROCOMPUTER LOCAL AREA NETWORK IN KOMPAN SYSTEM

In the paper the quantitative evaluation of the properties of the local area network 10NET of IBM PC XT/AT compatible microcomputers, from the point of view of its application in the KOMPAN system of the control blocks of nuclear power stations, has been presented. The communication methods accomplished by hardware (a method of the access to the network) as well as accomplished by software (network transmission mechanisms) have been discussed.

ИССЛЕДОВАНИЯ ОТНОСИТЕЛЬНО ПРИГОДНОСТИ МИКРОКОМПЬЮТЕРНОЙ ЛОКАЛЬНОЙ СЕТИ 10NET В СИСТЕМЕ РАСПРЕДЕЛЕННОЙ ОБРАБОТКИ ДАННЫХ "КОМΠΑИ"

В работе представлена количественная оценка свойств локальной компьютерной сети 10NET персональных микрокомпьютеров совместимых с IBM PC XT/AT с точки зрения ее применения в системе КОМПАИ. Рассмотрены решения коммуникации реализованных как аппаратным (метод доступа к сети), так и программным (сетевые механизмы передачи) путем.

Tabela 3.1. Intensywność zgłoszeń dla różnej kategorii użytkowników

- a) wartość maksymalna zależna od przepustowości
- b) wartość maksymalna zależna od opóźnienia tranzytowego
- c) wartość przyjęta dla następnych eksperymentów

Table 3.1. Entries rates for the various classes of users

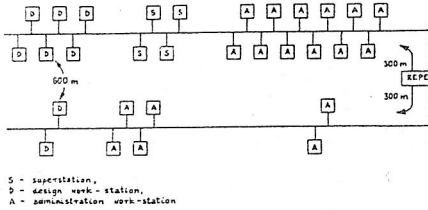
- a) maximal values determined in dependence on the throughput
- b) maximal values determined in dependence on the transit delay
- c) the user entries rates as values for next experiments

Długość zbioru [KB]	Intensywność zgłoszeń [1/s]			Typ stanowiska
	a)	b)	c)	
0	1/1	1/2	1/5	administra- cyjne
16	1/2	1/3	1/5	
32	1/3	1/4	1/5	
64	1/4	1/7	1/10	projektowe
128	1/8	1/10	1/20	
256	1/15	1/14	1/30	
512	1/30	1/20	1/60	

Tabela 3.2. Dopuszczalna liczba stanowisk dla różnych konfiguracji systemu KOMPAN

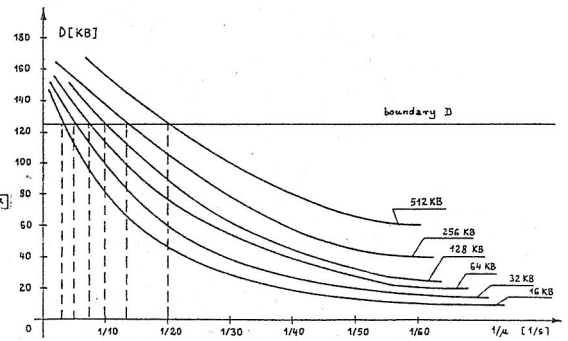
Table 3.2. Admissible number of stations of the KOMPAN system

Typ stanowiska podległego	Liczba stacji centralnych					
	2			3		
projektowe	4	6	8	4	6	8
administracyjne	14	7	1	16	10	4



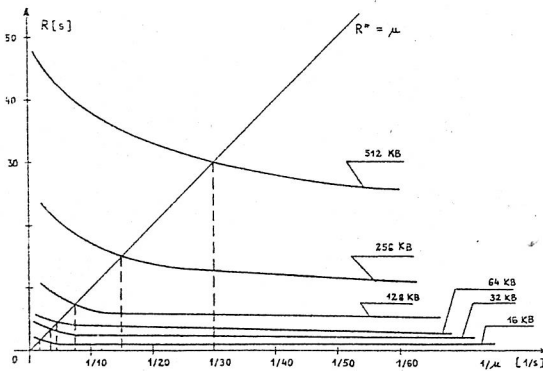
Rys. 3.1 Konfiguracja docelowa sieci Cx-NET

Fig. 3.1 Final stage configurations of the Cx-NET network



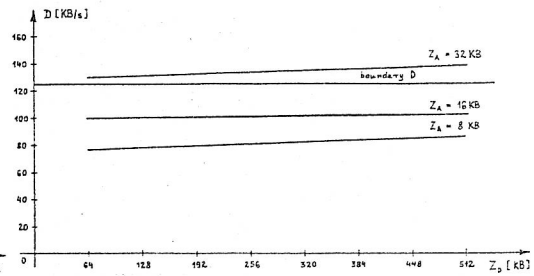
Rys. 3.2 Przepustowość sieci D w funkcji intensywności zgłoszeń użytkownika $1/\mu$ dla różnej długości zbiorów

Fig. 3.2 Throughput D of the network as a function of the user entries rate $1/\mu$ for various file lengths



Rys. 3.3 Opóźnienie tranzytowe R w funkcji intensywności zgłoszeń użytkownika $1/\mu$ dla zbiorów różnej długości

Fig. 3.3 Transit delay R as a function of the user entries rate $1/\mu$ for various file lengths



Rys. 3.4 Przepustowość sieci D w funkcji długości zbioru Z

Fig. 3.4 Throughput D of the network as a function of the length of the file Z

sieć komputerowa,
warstwa sieciowa,
zarządzanie

Elżbieta Hudyma*
Elżbieta Kosmulska-Bochenek*

ZARZĄDZANIE WARSTWĄ KOMUNIKACYJNĄ SIECI KOMPUTEROWEJ

W artykule przedstawiono ogólny model zarządzania warstwą komunikacyjną w DTE sieci komputerowej. Podstawowym celem zarządzania jest niedopuszczenie do negatywnych skutków przeciążenia sieci. Jako przykład przedstawiono mechanizmy ochronne warstwy sieciowej w mikrohoście realizowanym przez Centrum Obliczeniowe.

1. WSTĘP

Zarządzanie warstwą nie podlega standaryzacji w opisie modelu ISO/OSI 31, stąd literatura np. [1,2], dotycząca sieci komputerowych nie precyzuje bliżej tych funkcji. W konkretnej implementacji należy określić następujące zagadnienia:

- cele zarządzania warstwą,
- funkcje realizowane przez zarządzanie warstwy,
- współpraca z zarządzaniem warstw sąsiednich i zarządzaniem globalnym.

Celem artykułu jest próba zdefiniowania ogólnego modelu zarządzania warstwą, który byłby przydatny dla warstw komunikacyjnych, tj. od liniowej do transportowej włącznie. W artykule przedstawiono również przykład implementacji zarządzania dla warstwy sieciowej, będącej elementem mikrohosta realizowanego obecnie przez Centrum Obliczeniowe.

2. CELE ZARZĄDZANIA

Jednym z podstawowych celów zarządzania jest zapewnienie poprawnego zachowania się warstwy w warunkach konkretnej implementacji począwszy od włączenia urządzenia do sieci aż do końca pracy. Standardy protokołów każdej warstwy wydają się całkowicie gwarantować poprawną transmisję danych, jednakże abstrahują od ograniczeń implementacyjnych, jak np. ograniczenie pamięci lub błędy sprzętowe i programowe. Brak kontroli może doprowadzić do sytuacji w której warstwa nie może wysłać ani przyjąć żadnej jednostki protokołowej. Ponieważ może się to zdarzyć w dowol-

*Centrum Obliczeniowe Politechniki Wrocławskiej

nej chwili czasowej, warstwa może zachowywać się nieoczekiwanie, tzn. wysłać pewne komunikaty do warstw wyższych, a do sieci nie, lub odwrotnie (np. z powodu braku pamięci). Doprowadzić to może w najgorszym przypadku do utraty informacji lub blokady, lub co najmniej do znacznego pogorszenia pracy sieci jeśli wymuszone będą retransmisje. Jednym z podstawowych celów zarządzania warstwą powinno być zabezpieczenie przed tego rodzaju zdarzeniami.

Zarządzanie może również zmieniać efektywność działania sieci drogą sterowania przepływem. Korzystając z dostępnych danej warstwie środków może ona przyspieszać lub zwalniać strumień przesyłanych przez nią informacji. Uzależniać to powinna od aktualnych parametrów jakości sieci oraz od innych danych, jak np. żądań użytkowników.

Istotnym zadaniem szeroko pojętego zarządzania jest zbieranie danych o zdarzeniach występujących w warstwie i ich parametrach. Dane te wykorzystywane są w trybie on-line i off-line. Tryb on-line oznacza, że dane zdarzenie wywołuje natychmiastową reakcję ze strony warstwy, natomiast tryb drugi wykorzystywany jest dla celów statystycznych i badawczych.

3. NARZĘDZIA I METODY ZARZĄDZANIA

W zarządzaniu warstwą można wyróżniać następujące problemy:

- zbieranie i pamiętanie informacji dla celów zarządzania,
- środki oddziaływania na otoczenie,
- inicjacja akcji zarządzających,
- algorytmy decyzyjne zarządzania.

Zarządzanie korzysta z danych znajdujących się w strukturach lokalnych węzła niezbędnych dla poprawnego przepływu informacji przez warstwę zgodnie z zaleceniami i protokołami ISO/OSI. Pierwszy z tych problemów polega na zdefiniowaniu w ogólnej strukturze danych lokalnych warstwy dodatkowych danych niezbędnych do przechowywania wybranych informacji dla celów zarządzania, a także określenia momentu i sposobu ich aktualizacji. Oczywiście lokalizacja tych dodatkowych danych powinna być rozproszona i związana na ogół z najniższą rozróżnialną jednostką ciągu informacyjnego na danym poziomie. Na przykład dla warstwy liniowej taką jednostką jest linia, a dla warstwy sieciowej - połączenie. Część z tych danych nie związanych z taką jednostką może być zlokalizowanych w danych globalnych warstwy.

Dla warstwy sieciowej struktura danych dla zarządzania została przedstawiona w tabeli 1.

Tabela 1

Rodzaj informacji	Moment aktualizacji	Lokalizacja
Liczba zajętych buforów	Pobieranie i zwalnianie buf	Globalna
Liczba typów pakietów	Odebranie pakietu danego typu	Na połączeniu
Liczba typów prymitywów	Odebranie prymitywu danego typu	Na połączeniu
Kody diagnost. Przyczyny błędu	Odebranie pakietu CLEAR, RESET, RESTART	Na połączeniu
Czas przyścia ostatniego pak.	Odebranie pakietu	Na połączeniu

Kolejnym problemem, wymagającym określenia są środki, którymi zarządzanie może się posługiwać aby oddziaływać na otoczenie. Możliwe są następujące rozwiązania:

- wykorzystanie środków danych przez protokół warstwy w celu oddziaływania na bliźniaczą warstwę w zdalnym DTE,
- zdefiniowanie dodatkowych prymitywów lokalnych, służących do komunikacji z sąsiednimi warstwami,
- określenie współpracy z zarządzaniem globalnym DTE/DCE.

Przykładowo zarządzanie warstwą sieciową minihosta może komunikować się z otoczeniem wykorzystując:

- pakiety RR, RNR, RESET, CLEAR, RESTART, którymi może wstrzymać transmisję pakietów z danymi na dowolny czas, zerować połączenie, a w skrajnych przypadkach rozłączyć jedno lub więcej połączeń,
- prymitywy lokalne do warstwy liniowej oraz do warstwy transportowej za pomocą których blokuje się nadsyłanie prymitywów z danymi (mechanizm regulujący przepływ danych na styku warstw),
- komunikaty do zarządzania globalnego realizowane na wzór prymitywów międzywarstwowych.

Zarządzanie warstwą w oparciu o posiadane informacje oraz przy użyciu zdefiniowanych wcześniej środków oddziaływania realizuje określone procedury, z których najważniejsze to:

- Ustawienie stanu gotowości do pracy łącznie z zapewnieniem dostępności usług warstw niższych. Inicjacja globalna.
- Zmniejszenie napływu danych z warstw wyższych z powodu trudności wysłania pakietów i związanego z tym gromadzenia się pakietów. Inicjacja samoistna.

- Zmniejszenie napływu danych z warstwy niższej z powodu trudności wysłania prymitywów. Inicjacja samoistna.
- Zatrzymanie napływu danych ze zdalnego DTE na określonym połączeniu z powodu trudnej sytuacji w gospodarce pamięcią. Inicjacja samoistna.
- Zerowanie jednego lub kilku połączeń w celu zwolnienia pewnego obszaru pamięci niezbędnego dla poprawnego zrealizowania jakiejś operacji w warstwie. Inicjacja samoistna.
- Rozłączenie jednego lub kilku połączeń nieaktywnych aby umożliwić nawiązanie nowego połączenia. Inicjacja samoistna.
- Restart warstwy w przypadku uszkodzenia struktury danych. Inicjacja samoistna lub globalna.

Oprócz tego rodzaju samodzielnych działań niektóre akcje zarządzania warstwą mogą być inicjowane przez zarządzanie globalne poprzez odpowiednie komunikaty. Wyżej przedstawione akcje mogą być rozszerzone o inne procedury w zależności od stopnia złożoności zaimplementowania protokołu, tzn. czy zrealizowano jedynie obowiązkowy protokół, czy też zaimplementowano wszelkie możliwe opcje tego protokołu. W tym drugim przypadku zarządzanie ma większe możliwości wpływu na otoczenie, ale także dodatkowe obowiązki sterowania przepływem, negocjacji parametrów jakości usług itp.

4. ZAKOŃCZENIE

Funkcje zarządzania warstwą nie są zdefiniowane w protokołach poszczególnych warstw. Mogą one wydawać się zbędną przy pracy w nieprzeciążonej sieci, braku błędów zarówno w hardware jak i software i przy nieograniczonej pamięci. Dlatego przy projektowaniu urządzeń sieciowych często problemy związane z zarządzaniem zarówno globalnym jak i poszczególnymi warstwami są często pomijane. W warunkach realnej pracy może okazać się, że tylko dobrze zaprojektowane zarządzanie może uchronić sieć przed niekontrolowanym upadkiem lub co najmniej nieefektywnym działaniem, co bardzo obniża ocenę działania sieci w oczach użytkownika.

5. LITERATURA

- [1] Deasington R.J. X.25 Protocols for Packet Switching Networks Ellis Horwood Ltd 1986.
- [2] Schwartz Telecommunication Networks, Protocols, Modeling and Analysis, Addison-Wesley Publishing Comp. 1987.
- [3] CCITT Recommendation, Interface Between Data Terminal Equipment (DCE) for Terminals Operating in the Packet Mode on Public Data Networks, Red Book, Fascicle VIII.3 Rec. X.25 Malaga-Torremolinos 1984.
- [4] Tanenbaum A.S. Sieci komputerowe WNT Warszawa 1988.

- [5] Information Processing Systems, Open System Interconnection-Internal Organization of the Network Layer, ISO 8648 1988.02.15
- [6] Information Processing Systems, Open System Interconnection_Management Information Protocol Specification Common Management Protocol ISO SC21N2060 1988.12.87
- [7] Information Processing Systems OSI, Management Information Service Definition, Service Definition ISO SC21N2059 1987.12.16.

Layer Management of a Computer Network

In the paper a layer management service of a communication network is considered. Two major problems of it are pointed out: congestion and deadlock. The management service of network level is shown as an example.

sieci komputerowe, teleinformatyka

Zbigniew HULICKI*

KRAKOWSKA AKADEMICKA SIEĆ TELEINFORMATYCZNA - KONCEPCJA I ZAŁOŻENIA PROJEKTOWE

W pracy przedstawiono koncepcję oraz założenia projektowe sieci teleinformatycznej dla potrzeb uczelni Krakowa. Uwzględnia ona aktualne warunki ekonomiczne oraz możliwości szkół wyższych, a także istniejące, przede wszystkim technologiczne, ograniczenia. Zbadano również możliwości ewolucji sieci oraz jej perspektywiczne zastosowania.

1. WSTĘP

Rozproszenie zasobów komputerowych już od dłuższego czasu wywołuje i potęguje potrzebę ich integracji. Jest ona szczególnie istotna tam, gdzie praca zależy od wielu służb oraz informacji zbieranych z różnych źródeł. Rozwiązaniem jest sieć teleinformatyczna umożliwiająca intensyfikację działalności instytucji oraz zmniejszenie kosztów komunikacji (wewnętrznej i zewnętrznej) [1], [4], [5], [6], [7].

2. ZAŁOŻENIA PROJEKTOWE

Zasoby komputerowe krakowskich uczelni są zdecentralizowane i ponadto rozproszone w obrębie całego miasta (zob. [5]). Brak wzajemnych połączeń prowadzi do słabego (w porównaniu z potencjalnym) wykorzystania sprzętu oraz do ograniczenia liczby i rodzajów usług, które mogłyby być potencjalnie dostępne. Brak możliwości zaspokojenia wszystkich potrzeb użytkowników oraz różnorodność sprzętu i oprogramowania wymuszają szukanie nowych rozwiązań prowadzących do wykorzystania rezerw mocy obliczeniowej, wymiany oprogramowania, baz danych itd. W tym celu analizuje się możliwości budowy akademickiej sieci komputerowej integrującej zasoby informatyczne poszczególnych uczelni Krakowa, a w perspektywie, również instytutów naukowo-badawczych. W projekcie przewidziano stworzenie jednej sieci rozwijanej i modernizowanej krok po kroku, która połączy między sobą wszystkie zasoby komputerowe, m.in. projektowane sieci lokalne

* Katedra Telekomunikacji, Akademia Górniczo-Hutnicza, Kraków

- AGH, UJ i innych uczelni, oraz umożliwi dostęp do powszechnej sieci z komutacją pakietów [7] oraz Krajowej Akademickiej Sieci Komputerowej [2], [3], [5]. Instalacja projektowanej sieci komputerowej powinna umożliwić:
- dostęp z dowolnej końcówki (np. LSBT, TTY, itp.) do każdego komputera dołączonego do sieci oraz do wszystkich rodzajów usług (biblioteki oprogramowania, bazy danych itd.), tzn. pełną wzajemność połączeń,
 - wspólne użytkowanie miejscowych drogich urządzeń peryferyjnych (drukarki, plotery, itp.),
 - zcentralizowaną realizację dostępu do powszechnej sieci z komutacją pakietów oraz sieci KASK i innych sieci,
 - zwolnienie central telefonicznych z zadań transmisji danych tak szybko jak to będzie możliwe,
 - wysoką niezawodność oraz szybkość transmisji danych z końcówki użytkownika nie mniejszą niż 9.6 kBaud,
 - minimalizację opóźnień transmisyjnych oraz przestojów sieci,
 - możliwość zaspokojenia wszystkich potrzeb dotyczących komunikacji w obrębie projektowanej sieci przynajmniej w ciągu 15-tu lat po jej uruchomieniu.

Projektowana sieć powinna również odpowiadać zaleceniom ISO/OSI dotyczącym tzw. otwartej struktury sieci (zob. [6]) oraz wykorzystywać dostępny w kraju sprzęt (urządzenia, wyposażenie).

3. WYBÓR PODSIECI KOMUNIKACYJNEJ

W projektowanej sieci można przyjąć jedno z kilku podstawowych, konkurencyjnych w stosunku do siebie rozwiązań podsieci komunikacyjnej:

- wydzielona automatyczna centrala telefoniczna [4],
- wydzielona sieć z komutacją pakietów, wykorzystująca protokół X.25,
- szerokopasmowy kabel współosiowy, tzn. rozwiązanie typu sieci ETHERNET lub sieci pętlowej (zob. [6]),
- system szerokopasmowy typu CATV (Community Antenna Television)(zob. [4]),
- system radiowy z komutacją pakietów (zob. [6]).

Dokładną analizę przydatności powyższych rozwiązań podsieci komunikacyjnej dla lokalnych uczelnianych sieci komputerowych przedstawiono w pracy [5]. Wykazano tam, że w przypadku poszczególnej uczelni, jak również środowiska międzyuczelnianego, najbardziej korzystnym rozwiązaniem w warunkach Krakowa jest system typu CATV. Jest on znany od ponad 20 lat. Wyposażenie konieczne do instalacji systemu jest produkowane masowo. Z elektrycznego punktu widzenia, podsieć komunikacyjna oparta na tym systemie jest w stanie objąć obszar o średnicy do 50 km (zob. [4]). Łatwość rozbudowy oraz implementacji takiej sieci zwiększa jej atrakcyjność w przypadku długofalowego planowania rozwoju i działalności poszczególnych uczelni. Ten system nie jest niestety pozbawiony mankamentów (zob. [4])

dotyczących np. dostępu do kabla w ramach protokołu CSMA/CD, a w szczególności wpływu opóźnień propagacji sygnału na mechanizm wykrywania kolizji. Zmniejszanie się kosztów mikroprocesorów powinno zwiększać konkurencyjność tego systemu w stosunku do pozostałych rozwiązań podsieci komunikacyjnej [4], [6].

4. INSTALACJA SIECI KOMPUTEROWEJ

4.1. Pierwszy etap implementacji sieci

W pierwszym etapie instalacji sieci przewidziano połączenie całego sprzętu (o istotnym znaczeniu) w celu uzyskania doświadczeń przy wstępnej eksploatacji systemu. W tym celu należy wykorzystać istniejące i eksploatowane łącza ŚCO "Cyfronet" oraz czasowo wydzierżawić jedynie trzy dodatkowe łącza z powszechnej sieci telefonicznej. Ta prosta konfiguracja powinna być stosunkowo niezawodna. Długość głównych łączy będzie się wahać od 0.3 do 6 km, natomiast pozostałych od 1.2 do 12 km. W celu wykorzystania pełnej mocy obliczeniowej głównych komputerów przewidziano zastosowanie procesorów czołowych.

4.2. Instalacja sieci szerokopasmowej

Druga faza implementacji sieci międzyuczelnianej powinna rozpocząć się jednocześnie z pierwszą (wyżej opisaną). Obejmuje ona położenie kabla współosiowego między AGH i ŚCO "Cyfronet" oraz jego przedłużenie do UJ i AR. Ten etap winien zakończyć się okablowaniem budynków ww uczelni. Jednocześnie należałoby w ŚCO lub AGH zainstalować radiowe urządzenia teletransmisyjne umożliwiające przy pomocy kanałów radiowych dostęp do sieci szerokopasmowej jej pozostałym użytkownikom. Przewidziano również czasowe zastosowanie komutowanych połączeń modemowych sterowanych mikroprocesorowo do tych maszyn, które nie zostałyby od razu włączone do sieci szerokopasmowej. Łącza dzierżawione będą wykorzystywane do zakończenia instalacji całej sieci, co pozwoli porównać je z mikrofalowymi łączami radiowymi, ocenić zasadność ich stosowania w projektowanej sieci oraz uzyskać doświadczenia w praktycznej eksploatacji systemu szerokopasmowego i różnych technik komunikacji komputerowej. Rozbudowa sieci obejmuje przedłużenie kabla współosiowego z AGH do WSP i PK, a w perspektywie do AE i AM. Na tym etapie rozwoju sieci przewidziano instalację sprzęgu sieci szerokopasmowej z siecią KASK, z powszechną siecią z komutacją pakietów i z innymi sieciami. Ażeby uniknąć zakłóceń występujących w łączach miejscowych, implementację tego sprzęgu przewidziano w międzymiastowej centrali telefonicznej. W przyszłości szerokopasmowa sieć międzyuczelniana powinna objąć jedną lub dwie uczelniane lokalne sieci komputerowe, czyli całość zasobów, zapewniając tym samym realizację wszystkich potrzeb środowiska akademickiego dotyczących komunikacji komputerowej, przetwa-

rzania informacji oraz baz danych.

LITERATURA

1. Batycki T. Sieci Komputerowe. Wrocław Wyd. Pwr 1984
2. Bieleninik E. (i in.) Podsieć komunikacyjna Międzyuczelnianej Sieci Komputerowej. Informatyka 1985 T. 20 nr 11/12 s. 15 - 18
3. Góral S. (i in.) Użytkowanie Międzyuczelnianej Sieci Komputerowej. Informatyka 1987 T. 22 nr 3 s. 15 - 18
4. Huber W. Das Breitbandkommunikationssystem Kometh. Bulletin des SEV/VSE 1984 Bd. 75 Nr 17 s. 1036 - 1039
5. Hulicki Z. Local Area Network at the Academy of Mining and Metallurgy. praca złożona w IEEE Trans. on Commun.
6. Franta W.R. Imrich Ch. Local Networks. Lexington Lexington Books 1981
7. Kolendowski J. (i in.) Założenia tech.-ekon. sieci regionalnej MASK. Oprac. Nr 256/19/86 Kraków 1986

DATA COMMUNICATION NETWORK FOR THE UNIVERSITIES IN CRACOW

The reasons of development as well as the design aspects of a data communication network for the universities situated in the city of Cracow are considered. A solution is based on the actual economic conditions and possibilities of these institutions and takes into account technological limitations. Trends of evolution and further application of this network are examined too.

СЕТЬ ЭВМ ДЛЯ ВУЗОВ КРАКОВА

Работа знакомит с основными предпосылками построения сети эвм для вузов Кракова. Принятое решение учитывает экономические возможности вузов и ряд технологических ограничений. Рассмотрены возможности развития сети и её применений.

synchronizacja procesów,
sieć komputerowa KASK

Andrzej HUZAR*

Andrzej KALIŚ*

PROBLEMY SYNCHRONIZACJI W STACJI TRANSFERU ZBIORÓW SIECI KOMPUTEROWEJ KASK

W pracy zostały omówione sposoby synchronizacji procesów w stacji transferu zbiorów sieci komputerowej KASK na tle ogólnych metod synchronizacji procesów oraz uwarunkowań środowiska, w którym stacja transferu zbiorów powstała. Przedstawiono model synchronizacji procesów i jego realizację na potrzeby stacji. Krótko omówiono zalety proponowanego podejścia.

1. PROBLEMY SYNCHRONIZACJI W STACJI TRANSFERU ZBIORÓW

Wraz z powstaniem możliwości i potrzeb tworzenia oprogramowania złożonego z kilku procesów traktowanych jako programy sekwencyjne, pojawiła się konieczność ich synchronizacji. Korzystają one zazwyczaj z pewnych wspólnych zasobów, do których dostęp w danej chwili mogą mieć tylko niektóre z nich (często tylko jeden). W przypadku tworzenia programów działających współbieżnie, ich twórcy znajdują się w następującej sytuacji: z jednej strony dostępny jest im zbiór metod synchronizacji, z drugiej strony, środowisko programowe dostarcza pewne mechanizmy, przy pomocy których można synchronizować procesy. Usługi dostarczane przez system operacyjny (środowisko) nie muszą być gotowymi do wykorzystania mechanizmami synchronizacji. W większości przypadków charakter oprogramowania narzuca wybór pewnych metod synchronizacji.

Wydaje się, że współpracujące procesy można podzielić na dwie grupy różniące się sposobem synchronizacji:

1. procesy, które zasadniczo pracują niezależnie, tzn. praca innych procesów nie jest potrzebna do ich właściwego działania, korzystając jedynie od czasu do czasu ze wspólnych zasobów środowiska (drukarek, dysków itp.). Przydziałami tych zasobów zajmuje się nadzorca systemu, rozwiązując problemy synchronizacji,

*

Centrum Obliczeniowe Politechniki Wrocławskiej

2. procesy, które współpracując ze sobą wymieniają przetwarzaną informację (np. procesy będące implementacją tej samej warstwy sieci pracujące w różnych komputerach).

Stacja transferu zbiorów, została zaprojektowana jako zbiór współpracujących ze sobą akcji i podakcji (procesów). System może powodować przechodzenie procesów w stan "niegotowe", jeżeli muszą one na coś czekać. Z jednej strony, oczekiwanym zdarzeniem może być na przykład: zakończenie operacji wejścia-wyjścia, odebranie odpowiedzi od operatora, upływanie pewnego odcinka czasu, uzyskanie możliwości modyfikacji zbioru przetwarzanego równocześnie przez wiele zadań. Z drugiej zaś strony, proces może sama przechodzić w stan zawieszenia, oczekując na realizację pewnego zdarzenia, uzależnionego zazwyczaj od wykonania ściśle ustalonych czynności przez współpracujący z nim inny proces.

W stacji transferu zbiorów procesy współpracują ze sobą różnie, tzn. przez dostęp do wspólnych zasobów i przez wymianę komunikatów. Również ich synchronizacja jest zróżnicowana i odbywa się częściowo bezpośrednio przez system operacyjny i częściowo przez stosowanie mechanizmów utworzonych tylko na potrzeby stacji. W stacji można wskazać następujące miejsca, w których następuje synchronizacja procesów:

1. Stacja wysyła i odbiera treść przesyłanych zbiorów. Musi być więc rozwiązany problem dostępu do zbiorów przez stację i inne programy.. W tym przypadku skorzystano z systemowego mechanizmu synchronizacji (makroinstrukcje supervisora ENQ i DEQ z odpowiednią nazwą regionu krytycznego [4]),
2. Ze stacją współpracują programy uruchamiane przez użytkowników systemu transferu zbiorów, które wpisują zlecenia do kolejki. W tym przypadku również skorzystano z systemowego mechanizmu regionów krytycznych ustalając własną jego nazwę,
3. Stacja komunikuje się z inną stacją. Tutaj synchronizację zapewnia zastosowany protokół transferu zbiorów [5],
4. Procesy wchodzące w skład stacji komunikują się według własnych zasad.

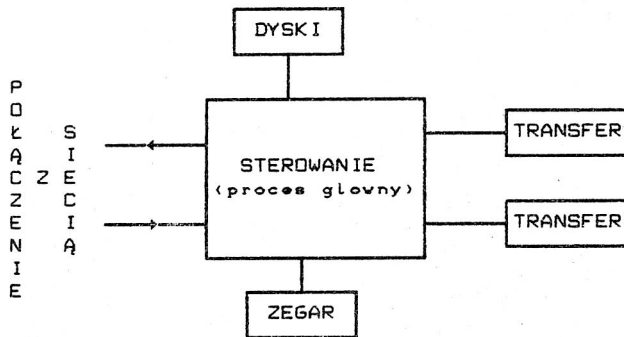
Proces główny o nazwie STEROWANIE nadzoruje pracę stacji, pobiera zlecenia z kolejki zleceń, pośredniczy w przesyłaniu komunikatów do sieci oraz komunikuje się z innymi procesami stacji, którymi są:

- a) proces ZEGAR informujący o upływie kolejnego kwantu czasu,
- b) procesy o nazwie TRANSFER (praktycznie w dowolnej liczbie), które realizują protokół transferu zbiorów.

Utworzenie takiej struktury wynika głównie z możliwości podziału funkcji pomiędzy poszczególnymi modułami (procesami) oraz z tego, że system OS/MVT dostarcza środków do stosunkowo łatwego tworzenia programów wieloprocessorowych (współbieżnych). Problemy, które twórcy musieli roz-

wiązać sami to:

- a) definicja wspólnych struktur danych,
- b) definicja własnego modelu synchronizacji zgodnego z potrzebami i możliwościami realizacyjnymi.



Rys 1. Struktura stacji transferu zbiorów

2. MODEL SYNCHRONIZACJI PROCESÓW

Przyjęto, że procesy będą przysyłać pomiędzy sobą żądania wykonania pewnych działań przez dwa kanały logiczne (po jednym dla każdego kierunku). Każdy kanał, do którego mają dostęp dwa procesy (nadawca i odbiorca), może się znajdować tylko w jednym z dwóch stanów:

1. kanał wolny - wtedy proces nadający może zainicjować żądanie wykonania pewnego działania (zbiór możliwych żądań jest ustalony). Żądanie spowoduje zmianę stanu kanału na zajęty i uaktywnienie odbiorcy (jeśli był zawieszony),
2. kanał zajęty - wtedy proces odbierający może wykonać żądane działanie zmieniając stan kanału na wolny (po jego zakończeniu). W tym stanie proces nadający nie może inicjować nadania kolejnych żądań.

W dowolnej chwili każdy z procesów może odczytywać stan kanału. Istotną cechą zastosowanego modelu synchronizacji jest to, że żaden z procesów nie musi czekać ani na dotarcie żądania do odbiorcy, ani na dostarczenie żądania od nadawcy. Ma to istotne znaczenie w przypadku, gdy proces STEROWANIE przysyła do procesu TRANSFER dane mające być zapisane na dysku, natomiast proces TRANSFER nie może zdążyć tych danych zapisać na dysku.

3. REALIZACJA MODELU SYNCHRONIZACJI PROCESÓW

Do realizacji przedstawionego modelu synchronizacji wykorzystano następujące mechanizmy i makroinstrukcje supervisoru systemu OS/MVT (ich

pełny opis można znaleźć w [4]):

1. ECB - zmienna, z którą związany jest mechanizm zawieszania i uaktywniania procesów (podakcji). Zawartość ECB wskazuje na to czy jakiś proces jest zawieszony, może zostać zawieszony lub nie zostanie zawieszony,
2. POST (ECB) - makroinstrukcja supervisor'a, która uaktywnia proces zawieszony lub zapobiega zawieszeniu takiego procesu, gdy ten wykona makroinstrukcję WAIT (ECB). Proces identyfikowany jest poprzez nazwę ECB.
3. WAIT (ECB) - makroinstrukcja powodująca zawieszenie danego procesu, aż do momentu wykonania przez inny proces makroinstrukcji POST (ECB). Oczywiście proces nie będzie zawieszony, gdy wcześniej została wykonana makroinstrukcja POST (ECB).

Jak można się domyślić, wartość zmiennej ECB ustawiana jest przez system operacyjny.

Po to by wykorzystać opisane wyżej makroinstrukcje w celu realizacji przedstawionego modelu synchronizacji, zdefiniowano wspólną strukturę danych. Poniżej została ona pokazana dla połączenia TRANSFER-STEROWANIE :

```

type
  DZIAŁANIE = (puste,działanie1,...,działanien); {odpowiada
                                                    stanowi kanału}
  ECB       = integer;
  POLE_SYNCHRONIZACJI = record
    ecb_sterowania,
    ecb_transferu           : ECB;
    kanał_transfer_sterowanie : DZIAŁANIE;
  end;
```

Na zmiennej x typu POLE_SYNCHRONIZACJI proces STEROWANIE działa według następującego schematu:

```

with x do
begin
  ecb_sterowania := 0;
  wykonaj_początek;
  while true do
  begin
    wait (ecb_sterowania);
    ecb_sterowania := 0;
    {jeśli ecb_sterowania=0, to wykonanie wait(ecb_sterowania)
     powoduje zawieszenie procesu }
    case kanał_transfer_sterowanie of
      wolny : ;
    ;
  ;
  działaniei : if możliwe_wykonanie_działaniai then
  begin
    wykonaj_działaniei;
    kanał_transfer_sterowanie:=wolny;
    post (ecb_transfer)
```

```

end; {case}
end; {while}
end; {with}

```

Natomiast proces TRANSFER może żądać wykonania pewnej działania od procesu STEROWANIE tylko wtedy, gdy wartość zmiennej kanał_transfer_sterowanie = wolny, czyli kanał jest wolny. Analogicznie został zdefiniowany kanał dla kierunku przeciwnego.

Warto zauważyć, że:

- proces nadający uaktywnia odbiorcę, jeśli ten jest zawieszony oraz uniemożliwia jego zawieszenie dopóki odbiorca nie sprawdzi stanu tego kanału,
- żaden z procesów nie musi aktywnie czekać w pętli na żądanie innych procesów,
- proces nadający zawsze jest odwieszany (jeśli był zawieszony) przez proces odbierający po zrealizowaniu żądania (jest w ten sposób informowany o zrealizowaniu działania).

4. PODSUMOWANIE

Przedstawione sposoby synchronizacji w stacji transferu zbiorów sprawdziły się w praktyce, ponieważ:

- a) umożliwiły niezależne uruchamianie poszczególnych procesów (Rys. 1),
- b) nie doprowadziły nigdy do jakichkolwiek problemów z synchronizacją w stacji,
- c) stacja, pracując w warunkach pełnego obciążenia, wykorzystuje około 10% czasu procesora, co wydaje się być wielkością dopuszczalną,
- d) metoda synchronizacji pełni w oprogramowaniu stacji rolę metody strukturalnej. Wprawdzie nie następuje syntaktyczne sprawdzenie poprawności metody, ale łatwe jest (co było robione) sprawdzenie "ręczne" tekstu programu,
- e) jest jednolitą dla całej stacji oraz innych komponentów oprogramowania komputera obliczeniowego i nie zależy od liczby typów działań.

Okazało się też, że mimo braku nowoczesnego języka do programowania współbieżnego (np. Concurrent Pascal), można uzyskać podobny komfort programowania stosując się jedynie do pewnych reguł programowania [1] (moduł STEROWANIE zakodowany został w Pascalu 360).

LITERATURA

- [1] Huzar A., Programowanie współbieżne w systemie OS/JS na poziomie języka Pascal, INFORMATYKA, nr 6, 1983,
- [2] Huzar Z., Wstęp do programowania współbieżnego, Wydawnictwa

- Politechniki Wrocławskiej, Wrocław 1985,
- [3] Iszkowski W., Maniecki M., Programowanie współbieżne, WNT, Warszawa 1982,
 - [4] OS/JS Makroinstrukcje supervisor, WZE Mera-Elwro, Wrocław 1976
 - [5] A Network Independent File Transfer Protocol, File Transfer Protocol Implementore Group, NIFTP-B(80), 5th Feb. 1981 ("Blue Book").

SOME SYNCHRONIZATION PROBLEMS IN FILE TRANSFER STATION OF COMPUTER NETWORK KASK

In the paper synchronization methods of processes in file transfer station are described. These methods are compared with general synchronization methods. Advantages of proposed methods are shortly depicted.

ПРОБЛЕМЫ СИНХРОНИЗАЦИИ В СТАНЦИИ ТРАНСФЕРА ФАЙЛОВ ВЫЧИСЛИТЕЛЬНОЙ СЕТИ KASK

В работе представляются методы синхронизации процессов в станции трансфера файлов. Методы сравниваются с общими методами синхронизации процессов. Коротко обсужданы достоинства представляемых способов.

Zbigniew HUZAR*
Zdzisław SPŁAWSKI*

Metodologia programowania,
specyfikacja,
implementacja

Tworzenie oprogramowania sieciowego - Problemy metodologiczne i teoretyczne**

Przedstawiono fazy konstruowania programu, interpretacje pojęć specyfikacji i implementacji, ich związki, własności i sposoby dowodzenia. Omówiono problemy związane z wyborem metodologii tworzenia oprogramowania oraz wyborem języków specyfikacji i programowania. Omówiono niektóre spośród tych zagadnień w kontekście systemów reaktywnych, a następnie zilustrowano je przykładem specyfikacji i weryfikacji w CCS prostego protokołu komunikacyjnego.

1. WSTĘP

1.1. Fazy konstruowania programu

Oprogramowanie ma stanowić rozwiązanie problemu pochodzącego z konkretnej dziedziny zastosowań. Punktem wyjścia do tworzenia takiego rozwiązania jest sformułowanie wymagań przez użytkownika oprogramowania. Podczas gdy program stanowi dobrze zdefiniowany obiekt, dziedzina zastosowań zwykle taką nie jest. Dlatego wymagania użytkownika, wyrażone w języku takiej dziedziny, są najczęściej - w znacznym przynajmniej stopniu - nieformalne. Budowa programu, o którym można twierdzić, że jest programem poprawnym, tzn. stanowiącym rozwiązanie wyjściowego problemu, wymaga uprzedniego przedstawienia jego formalnej specyfikacji, tzn. jednoznacznego i wyczerpującego sformułowania wymagań użytkownika w pewnym języku formalnym.

Z punktu widzenia inżynierii oprogramowania w konstruowaniu programu można wyróżnić trzy fazy [21,25,28]:

Specyfikacja → Projekt → Implementacja

* Centrum Obliczeniowe, Politechnika Wrocławska

** Pracę zrealizowano w ramach CPBR 8.13 "Budowa Krajowej Akademickiej Sieci Komputerowej", cel nr 52. "Formalne metody specyfikacji i konstrukcji oprogramowania sieciowego".

Specyfikacja, opisując zachowanie programu z punktu widzenia użytkownika, stanowi ekstensjonalną charakterystykę programu. Opisuje ona co program ma robić, bez żadnych wskazówek jak to osiągać. Projekt polega na określeniu struktury programu. Podejmowane są decyzje jak osiągać cel i są one wyrażane przy użyciu komponentów, o których wiadomo co robią, bez wnikania jak to robią. Wreszcie implementacja polega na określeniu tego jak działają poprzednio użyte komponenty.

1.2. Relatywność pojęć specyfikacji i implementacji

W praktyce każda z wymienionych faz może składać się z wielu etapów, przy czym niekiedy trudno jednoznacznie zaliczyć konkretny etap do jednej z faz [21,25,28]. Przykładem umowności tego, czy dany opis traktować jako specyfikację czy jako implementację jest fakt, że w latach 50-tych program w Pascalu, czy też w latach 60-tych program w Prologu mogłyby być traktowane jako specyfikacja implementacji tworzonej w Fortranie. Przykład ten wskazuje, że gdy na dany opis patrzymy z punktu widzenia sposobu jego działania traktujemy go jako program, jeżeli zaś aspekt ten pomijamy, to traktujemy go jako specyfikację.

Przyjmuje się, że proces tworzenia oprogramowania składa się z ciągu kroków [18,21]:

$$S_1 \rightarrow \dots \rightarrow S_i \rightarrow S_{i+1} \rightarrow \dots \rightarrow S_n$$

z których pierwszy S_1 jest specyfikacją wymagań użytkownika, następne są kolejnymi jego rozwinięciami, a ostatni S_n stanowi końcową implementację. Schemat ten jest wygodny do rozpatrywania problemów tworzenia oprogramowania. Dokładniej, będą rozpatrywane związki zachodzące pomiędzy dwoma sąsiednimi poziomami tego schematu. Z tego punktu widzenia obiekt S_i będzie traktowany jako specyfikacja, natomiast S_{i+1} jako implementacja.

1.3. Problemy

Ogólnie można wyróżnić problemy dotyczące:

- wyboru metodologii przechodzenia pomiędzy poziomami;
- przyjęcia na obu poziomach S_i oraz S_{i+1} jednego bądź dwóch języków,
- rozumienia pojęcia implementacji.

Celem artykułu jest krótkie omówienie wymienionych wyżej problemów. W dalszej części, w rozdziale 2, będą przedstawione problemy związane z tworzeniem specyfikacji wymagań użytkownika, rozdział 3 jest ogólnym przeglądem wymienionych wyżej problemów, a pozostałe rozdziały 4 i 5 odnoszą się do niektórych spośród tych problemów w kontekście tworzenia oprogramowania sieciowego.

2. TWORZENIE SPECYFIKACJI WYMAGAŃ UŻYTKOWNIKA

2.1. Co opisuje specyfikacja?

Specyfikacja opisuje zewnętrznie dostrzegalne efekty działania programu. Możliwe są dwa sposoby rozumienia tych efektów [22].

Pierwszy zakłada, że obserwowalne są zmiany stanów (wartościowania globalnych zmiennych) programu. Takie podejście jest stosowane w specyfikacjach programów transformacyjnych i programów reaktywnych, wykorzystujących do komunikacji zmienne dzielone. Programy transformacyjne to takie programy, które dla pewnego zestawu danych produkują pewien wynik, po czym kończą działanie. Specyfikacja w tym przypadku wyraża związek pomiędzy stanem początkowym a stanem końcowym programu. Programy reaktywne nie produkują określonego wyniku, lecz reagują na informacje kierowane do nich z otoczenia, w którym funkcjonują (przykładem programów tego typu są systemy operacyjne, systemy rezerwacji biletów, poczty elektronicznej itp.). Specyfikacje programów reaktywnych ze zmiennymi dzielonymi opisują zmiany wartości tych zmiennych w czasie.

Drugi sposób zakłada, że obserwowalne są interakcje pomiędzy programem a jego otoczeniem, zachodzące w trakcie wykonywania programu. W tym przypadku specyfikacja przedstawia zbiór ciągów interakcji, które może dostrzec zewnętrzny obserwator programu [9,14,19,20]. Takie rozumienie jest stosowne przy opisie programów reaktywnych, współpracujących ze swym otoczeniem poprzez wymianę komunikatów.

2.2. Czym jest specyfikacja?

Obecnie w większości prac poświęcanych temu zagadnieniu [13,18,22,28,29] jest wyrażany pogląd, że specyfikację należy traktować jako teorię sformalizowaną. Bazą specyfikacji jest odpowiednio wybrany system logiczny np. logika klasyczna pierwszego rzędu, logika modalna i temporalna, logika klauzul Horna, rachunek równości algebraicznych i wiele innych. Specyfikację tworzy się poprzez dołączenie do wybranego systemu logicznego symboli i aksjomatów pozalogicznych.

2.3. Sprawdzanie poprawności specyfikacji

Specyfikacja powinna w sposób zadowalający opisywać rozwiązywany problem, a o jej adekwatności decyduje oczywiście użytkownik, przy czym środki informatyczne mogą go częściowo wspomóc. Pierwsza kwestia dotyczy niesprzeczności specyfikacji, natomiast druga - jej zupełności. Odpowiedzi na pierwsze pytanie - teoretycznie przy najmniej w pewnym zakresie - można udzielić automatycznie, natomiast na drugie pytanie można odpowiedzieć tylko empirycznie. Decyzje podejmuje tu użytkownik,

zaś komputer może mu jedynie udzielić pewnej pomocy. Pomoc ta może polegać na wykorzystaniu techniki szybkiego makietowania lub specyfikacji wykonywalnych [6,29].

Szybkie makietowanie polega na zbudowaniu możliwie prostego programu, który działa zgośnie ze specyfikacją, chociaż może to czynić bardzo nieefektywnie.

Specyfikacje wykonywalne wykorzystuje się bądź do generowania twierdzeń wyprowadzalnych w teorii stanowiącej specyfikację, bądź do dowodzenia zadanych twierdzeń w tej samej teorii. W pierwszym przypadku użytkownik stwierdza czy wygenerowane twierdzenia są prawdziwe, tzn. mają właściwą interpretację, w konkretnej dziedzinie zastosowań, natomiast w przypadku drugim użytkownik zadając twierdzenie prawdziwe w dziedzinie zastosowań sprawdza czy ma ono dowód w specyfikacji.

2.4. Mechanizmy strukturalizacji specyfikacji

Drugi problem przy tworzeniu specyfikacji polega na tym, że rzadko udaje się utworzyć ją w całości bez odwoływania się do innych obiektów wcześniej już zdefiniowanych bądź takich, które dopiero będą definiowane. Wyłaniają się tutaj dwa postulaty: po pierwsze - chodzi o to, aby język specyfikacji dopuszczał takie odwoływania, oraz - po drugie - aby udostępniał odpowiednie mechanizmy strukturalizacji specyfikacji. Tutaj także powszechnie jest wyrażany pogląd, że paradygmatycznymi mechanizmami strukturalizacji są mechanizm parametryzacji, realizowany przez moduły generyczne (polimorficzne), oraz mechanizm wzbogacania (prefiksowania). Oba te mechanizmy znalazły zastosowanie w języku algebraicznych specyfikacji abstrakcyjnych typów danych ACT ONE [12], który z kolei stał się fragmentem języka formalnych specyfikacji standardów sieci komputerowych LOTOS [5,15].

3. PROBLEMY TWORZENIA OPROGRAMOWANIA

3.1. Podejścia do formalnej konstrukcji programów

Według Jonesa [16] w tworzeniu oprogramowania rozróżnia się trzy, nie zawsze rozłączne podejścia:

Podejście oparte na schemacie specyfikacja - projekt - weryfikacja polega na pisaniu specyfikacji i programu w odmiennych językach. Poprawność implementacji zapewnia się przez udowodnienie zachodzenia odpowiednich związków pomiędzy specyfikacją a implementacją. Podejście to jest bardzo popularne, a jego przykładami są systemy dowodzenia własności programów oparte na logice Hoare'a i licznych jej uogólnieniach, logice dynamicznej, logice algorytmicznej, logice temporalnej [6,22].

Podejście następne opiera się na transformacji specyfikacji. Jeżeli specyfikacja początkowa jest wykonywalna, ale - być może - mało efektywna, wtedy jej przekształcenie, w drodze zabiegów syntaktycznych - z którymi niekiedy wiąże się obowiązek dowodzenia poprawności ich stosowania - daje nową, bardziej efektywną specyfikację. Przykładem tego podejścia jest projekt CIP. Innym celem transformacji specyfikacji może być zmiana reprezentacji z postaci abstrakcyjnej na postać bliższą końcowej implementacji (rozszerzenie konserwatywne [18]). Jeszcze innym celem może być uściślenie specyfikacji. Należy również wspomnieć o pracach, których celem jest jednolite spojrzenie na różne możliwości transformacji specyfikacji - chodzi tu głównie o pojęcie instytucji [13,28].

Ostatnie z podejść opiera się na matematyce konstruktywnej. Specyfikację uważa się tutaj za twierdzenie do udowodnienia a program uzyskuje się z konstruktywnego dowodu tego twierdzenia. Podejście to jest nowym i chyba perspektywicznym kierunkiem badań. Przykładem zastosowania tego podejścia do programów funkcyjnych i typów danych jest praca [26]. Jak dotąd podejście to nie znalazło jeszcze zastosowania w programowaniu współbieżnym.

3.2. Język specyfikacji a język implementacji

Druga grupa problemów tworzenia oprogramowania wynika z tego czy używa się jako języka specyfikacji i implementacji tego samego czy też różnych języków.

Użycie dwóch języków praktycznie oznacza, że językiem specyfikacji jest język logiczny a językiem implementacji - język programowania. Pomiedzy specyfikacją S_1 a programem S_{1+1} powinna zachodzić relacja spełniania, co będzie oznaczane przez S_{1+1} sat S_1 [10,22,25]. Interpretacja tej relacji może być różna i zależy od klasy własności, które wyraża specyfikacja. Program który spełnia swą specyfikację nazywa się programem poprawnym.

Dwoma zasadniczymi problemami formalnymi tworzenia programów są tu problemy konstrukcji i weryfikacji. Problem konstrukcji polega na tym, aby znając specyfikację S_1 zbudować spełniający ją program S_{1+1} . Natomiast weryfikacja polega na zbadaniu czy program S_{1+1} spełnia specyfikację S_1 . Wspólnym elementem dla obu problemów jest system dowodowy.

W podejściu jednojęzykowym zakłada się, że język zawiera pewien fragment, który może być wykonywany. Początkowa specyfikacja w tym języku jest ukierunkowana na określenie tego co program robi, bez zwracania uwagi na efektywność działania. Następnie poprzez serię transformacji specyfikacja jest przekształcana tak, aby ostatecznie

otrzymać akceptowalnie efektywną implementację [10,22]. Pomiedzy kolejnymi specyfikacjami S_i , S_{i+1} powinien zachodzić związek inkluzji lub równoważności, co oznacza się $S_i \text{ eqv } S_{i+1}$. Interpretacja tej relacji może być rozmaita i zależy od klasy porównywanych własności. Ogólnie podejście jednojęzykowe charakteryzuje wykorzystanie środków algebraicznych [10,19,20].

3.3. Pojęcia implementacji

Pojęcia implementacji, jako przejścia od S_i do S_{i+1} i, w ślad za tym, rozumienie relacji sat lub eqv mogą być następujące [4].

Specyfikacja S_{i+1} może być redukcją specyfikacji S_i . Oznacza to taką sytuację, gdy specyfikacja S_{i+1} ustala zachowania w tych przypadkach, w których specyfikacja S_i dopuszcza wybór niedeterministyczny. Specyfikacja S_{i+1} spełnia dwa następujące wymogi:

- każde zachowanie działania S_{i+1} jest także dopuszczalnym działaniem S_i ;
 - każde działanie, które S_{i+1} odrzuca, jest również odrzucane przez S_i .
- Użyte powyżej pojęcie odrzucenia działania odnosi się tylko do systemów reaktywnych. Odrzucenie przez system pewnego działania, czyli pewnego ciągu interakcji oznacza, że w pewnym kroku takiego działania powstaje blokada systemu.

Specyfikacja S_{i+1} może być rozszerzeniem S_i . Związek ten zachodzi wówczas, gdy specyfikacja S_i jest specyfikacją częściową. W tym przypadku S_{i+1} dokonuje ustalenia tych zachowań, które są poza obszarem definicji S_i . Specyfikacja S_{i+1} spełnia dwa warunki:

- każde działanie, które jest dopuszczalne przez S_i jest także działaniem wykonywanym przez S_{i+1} (co nie wyklucza jednak wykonywania jeszcze dodatkowych działań);
- każde działanie, które S_{i+1} odrzuca jest także odrzucane przez S_i (S_{i+1} nie może odrzucać więcej zachowań niż czyni to S_i).

Specyfikacja S_{i+1} może być uściśleniem S_i . W tym przypadku obie specyfikacje są równoważne ekstensjonalnie, tzn. reprezentują identyczne zbiory obserwowalnych zachowań. Różnica pomiędzy nimi polega na tym, że S_{i+1} wnosi dodatkowe szczegóły o wewnętrznej strukturze programu, przedstawiając go jako kolekcję komponentów o mniejszym poziomie abstrakcji, czyli S_{i+1} jest pewną dekompozycją S_i .

3.4. Własności implementacji i ich dowodzenie

Ogólnie o programie można formułować własności należące do dwóch klas: bezpieczeństwa i żywotności [24]. Własności bezpieczeństwa mówią o tym, że program nigdy nie znajdzie się w stanie niedopuszczalnym, tzn. "nie zdarzy się nic złego", na przykład program nie zerwie obliczeń,

badź też jeżeli zakończy się, to wyprodukuje spodziewany wynik (poprawność częściowa). Własność żywotności gwarantuje, że program osiągnie pożądany stan, tzn. "zdarzy się coś dobrego", na przykład program zakończy swe działanie. Zatem zależnie od klasy własności, których oczekuje się od programu, mówi się o różnych rodzajach poprawności programu S_{i+1} względem S_i .

Dowodzenie ustalonych własności programu, a więc związku pomiędzy jego specyfikacją i implementacją prowadzi się w oparciu o system dowodowy. Stanowi on teorię formalną, której język jest "sumą" języków specyfikacji i programowania.

System dowodowy powinien być niesprzeczny i - w miarę możliwości - zupełny. Niesprzeczność jest konieczną własnością systemu i oznacza, że każde dwa programy S_{i+1} , S'_{i+1} semantycznie równoważne, co piszemy $S_{i+1} \text{ sem } S'_{i+1}$, równocześnie spełniają albo równocześnie nie spełniają danej specyfikacji S_i , tzn. jeżeli $S_{i+1} \text{ sem } S'_{i+1}$ to $S_{i+1} \text{ sat } S_i$ oraz $S'_{i+1} \text{ sat } S_i$. Zupełność jest pożądaną ale niekonieczną własnością i oznacza, że zachodzi implikacja odwrotna, tzn. jeżeli $S_{i+1} \text{ sat } S_i$ oraz $S'_{i+1} \text{ sat } S_i$, to $S_{i+1} \text{ sem } S'_{i+1}$.

Bardzo ważnym postulatem metodologicznym odnoszącym się do systemów dowodzenia własności programów jest kompozycyjność (modularność). Wymóg ten polega na tym, aby każdemu konstruktorowi CON operującemu na programach (złożenie sekwencyjne, równoległe, wybór, iteracja itp.) odpowiadał funktor FUN działający na specyfikacjach taki, że jeżeli

$$S_{i+1}^1 \text{ sat } S_1^1, \dots, S_{i+1}^n \text{ sat } S_1^n,$$

to

$$\text{CONCS}_{i+1}^1, \dots, S_{i+1}^n \text{ sat FUNCS}_1^1, \dots, S_1^n$$

4. KONSERUOWANIE SYSTEMÓW REAKTYWNYCH

4.1. Co specyfikować?

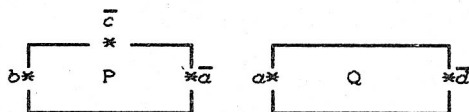
Systemy reaktywne, do których można również zaliczyć protokoły komunikacyjne, zostały krótko opisane w p.2.1. System reaktywny z wymianą komunikatów można przedstawić jako zbiór procesów, komunikujących się ze sobą i ze środowiskiem (które również można potraktować jako proces) przez punkty interakcji zwane portami (Rys.1).

Typy danych, przesyłanych między procesami można specyfikować niezależnie od specyfikacji procesów, czyli

specyfikacja systemu reaktywnego =

specyfikacje abstrakcyjnych typów danych + specyfikacje procesów.

Takie podejście zastosowano w języku formalnych specyfikacji standardów sieci komputerowych LOTOS [5,15], w którym specyfikacje procesów są wyrażane w języku, stanowiącym rozszerzenie języka CCS



Rys.1. Dwa współpracujące procesy P,Q
(a, b - porty wejściowe, $\bar{a}, \bar{c}, \bar{d}$ - porty wyjściowe)

Fig.1. Two interacting processes P,Q
(a, b - input ports, $\bar{a}, \bar{c}, \bar{d}$ - output ports)

Milnera [20], a do specyfikacji typów danych jest używany algebraiczny język ACT ONE [12]. Niżej zostanie przedstawiony podzbiór języka CCS, który będzie wykorzystany w rozdziale 5 do specyfikacji prostego protokołu.

4.2. CCS (Calculus of Communicating Systems)

CCS pozwala opisywać procesy, współpracujące ze sobą poprzez wymianę komunikatów, wysyłanych i odbieranych przez porty. W elementarnej interakcji uczestniczą dwa procesy, jeśli są do niej gotowe (mechanizm rendez-vous). Np. proces Q na rys.1 jest gotowy do przyjęcia komunikatu przez port wejściowy a , a proces P jest gotowy do wysłania komunikatu przez port wyjściowy $\bar{a}$. Może więc nastąpić interakcja, polegająca na przesłaniu komunikatu z $\bar{a}$ do a , która przekształci oba procesy. Nazwy portów wejściowych można więc także interpretować jako nazwy akcji, a odpowiednie nazwy portów wyjściowych jako nazwy akcji sprzężonych, których jednoczesne wykonanie stanowi niepodzielną interakcję.

Niżej zostanie opisana składnia podzbioru CCS (bez przekazywania danych i zmian nazw portów). Wykorzystywane są następujące oznaczenia:

$P = \{P, Q, R, \dots\}$ - zbiór procesów;

$V = \{X, Y, Z, \dots\}$ - zbiór zmiennych oznaczających procesy;

$A = \{a, b, c, \dots\}$ - zbiór portów wejściowych (akcji obserwowalnych);

$\bar{A} = \{\bar{a}, \bar{b}, \bar{c}, \dots\}$ - zbiór portów wyjściowych (akcji obserwowalnych sprzężonych) takich, że dla każdego $a \in A$ istnieje $\bar{a} \in \bar{A}$ i odwrotnie, oraz $\bar{\bar{a}} = a$;

τ - wewnętrzna akcja procesu, wykonywana przez proces autonomicznie i niewidoczna dla obserwatora zewnętrznego (nie posiadająca akcji sprzężonej, czyli $\bar{\tau} = \tau$).

Niech m oznacza dowolną akcję ze zbioru $A \cup \bar{A} \cup \{\tau\}$. Zbiór termów $E = \{E, F, \dots\}$ jest zdefiniowany następująco:

$$E ::= X \mid \text{NIL} \mid m; E \mid E + E \mid \text{rec } X. E \mid E \mid E \mid E$$

gdzie USA .

Termy CCS bez zmiennych wolnych przedstawiają procesy. Nieformalne

wyjaśnienie konstrukcji języka ułatwi zrozumienie formalnych reguł, zadających semantykę.

- (1). NIL - proces pusty, który nie jest w stanie wykonać żadnej akcji.
- (2). $m;E$ - proces, który może wykonać akcję m , po czym zachowuje się jak proces E .
- (3). $E+F$ - proces, zachowujący się jak E lub F (wybór jest niedeterministyczny).
- (4). $\text{rec } X.E$ - rekurencyjna definicja procesu. Operator rec wiąże wszystkie wystąpienia zmiennej wolnej X w E .
- (5). $E \setminus U$ - proces, zachowujący się jak proces E z zakazem wykonywania akcji ze zbioru U . Jeśli $a \in U$ to nie może on wykonać akcji a ani akcji sprzężonej $\bar{a}$. Inaczej mówiąc, wszystkie porty ze zbioru U stają się portami lokalnymi procesu, niedostępnymi dla środowiska. Taki mechanizm abstrakcji umożliwia modularyzację systemów.
- (6). $E|F$ - równoległe złożenie procesów E i F z możliwością komunikacji.

Semantyka operacyjna jest zdefiniowana za pomocą przedstawionych niżej reguł wnioskowania. Zapis

$$P \xrightarrow{m} Q$$

oznacza, że proces P może wykonać akcję m i przekształcić się w proces Q .

$$(1). \quad m;E \xrightarrow{m} E$$

$$(2). \quad \frac{E \xrightarrow{m} E' \quad F \xrightarrow{m} F'}{E+F \xrightarrow{m} E'}$$

$$(3). \quad \frac{E[\text{rec } X.E/X] \xrightarrow{m} F}{\text{rec } X.E \xrightarrow{m} F} \quad \text{gdzie } E[\text{rec } X.E/X] \text{ oznacza wyrażenie } E, \text{ w którym wszystkie wystąpienia zmiennej wolnej } X \text{ zostały zastąpione wyrażeniem } \text{rec } X.E$$

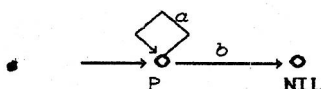
$$(4). \quad \frac{E \xrightarrow{m} F}{E \setminus U \xrightarrow{m} F \setminus U} \quad m, \bar{m} \notin U \quad \frac{E \xrightarrow{m} F}{E \setminus U \xrightarrow{\tau} F \setminus U} \quad m, \bar{m} \in U$$

$$(5). \quad \frac{E \xrightarrow{m} E' \quad F \xrightarrow{m} F'}{E|F \xrightarrow{m} E'|F} \quad \frac{E \xrightarrow{\alpha} E' \quad F \xrightarrow{\bar{\alpha}} F'}{E|F \xrightarrow{\tau} E'|F'}$$

Konstruktory procesów są uporządkowane według malejącej siły wiązania: $;$, $\setminus$, $|$, $+$, rec . Umożliwi to opuszczanie większości nawiasów w termach.

Przykłady.

(1). Proces $P = \text{rec } X. \bar{a}; X + b; \text{NIL}$ można również przedstawić za pomocą następującego automatu



co podsuwa jeszcze jedną interpretację termów CCS jako stanów systemu reaktywnego. Język CCS jest jednak znacznie bogatszy od języka teorii automatów, np. w procesie $\text{rec } X. (a; X | \bar{b}; X)$ ilość podprocesów wzrasta po każdej akcji, nie można go więc opisać za pomocą automatu skończonego.

(2). Proces $a; P | \bar{a}; Q$ może wykonać akcję a lub $\bar{a}$, komunikując się ze środowiskiem (jeśli jest do tego gotowe), lub akcję τ . Natomiast proces $(a; P | \bar{a}; Q) \setminus \{a\}$ może, ze względu na ograniczenie $\{a\}$, wykonać tylko akcję τ , czyli $(a; P | \bar{a}; Q) \setminus \{a\} \xrightarrow{\tau} (P | Q) \setminus \{a\}$.

4.3. Równoważność obserwacyjna i kongruencja obserwacyjna

CCS jest językiem, umożliwiającym opis systemu na rozmaitych poziomach abstrakcji (p.1.2). Jest to typowe podejście jednojęzykowe (p.3.2) i wymaga zdefiniowania relacji równoważności dla procesów. Takich relacji może być wiele [7]. Jedną z nich jest zdefiniowana przez Milnera relacja równoważności obserwacyjnej $\approx$. Mówiąc nieformalnie, dwa procesy są równoważne obserwacyjnie, jeśli nie można ich odróżnić na podstawie obserwacji zewnętrznego zachowania (nie bierze się pod uwagę τ -akcji, które są niewidoczne dla obserwatora).

Definicje.

(1). $P \xrightarrow{\tau}^* Q \equiv_{\text{def}} \exists n \geq 0, P_1, \dots, P_n. P \xrightarrow{\tau} P_1 \xrightarrow{\tau} \dots \xrightarrow{\tau} P_n = Q$.

(2). Niech $s = a_1 \dots a_n$ dla $n \geq 0$ i $a_i \in A \cup \bar{A}$, ($1 \leq i \leq n$).

$P \xrightarrow{s} Q \equiv_{\text{def}} \exists P_0, \dots, P_n. P \xrightarrow{\tau^*} P_0 \xrightarrow{a_1} P_1 \xrightarrow{\tau^*} \dots \xrightarrow{a_n} P_n \xrightarrow{\tau^*} Q$.

(3). $P \approx_0 Q$ dla wszystkich procesów P, Q .

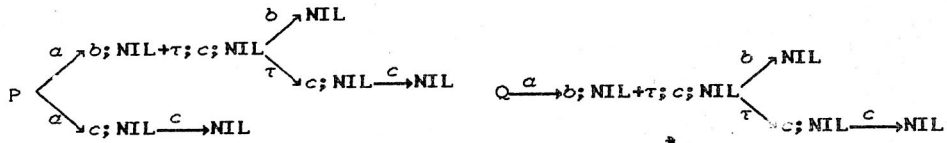
$P \approx_{i+1} Q \equiv_{\text{def}} \forall s \in (A \cup \bar{A})^*. ((\text{jeśli } P \xrightarrow{s} P' \text{ to } \exists Q'. Q \xrightarrow{s} Q' \text{ \& } P' \approx_i Q') \text{ oraz } (\text{jeśli } Q \xrightarrow{s} Q' \text{ to } \exists P'. P \xrightarrow{s} P' \text{ \& } P' \approx_i Q))$,
gdzie $(A \cup \bar{A})^*$ oznacza zbiór skończonych ciągów akcji.

(4). $P \approx Q \equiv_{\text{def}} \forall i. P \approx_i Q$.

Przykłady.

(1). $P = a; (b; \text{NIL} + \tau; c; \text{NIL}) + a; c; \text{NIL}$ $Q = a; (b; \text{NIL} + \tau; c; \text{NIL})$

Drzewa wyvodu dla tych procesów, otrzymane przez zastosowanie reguł wnioskowania z p.4.1 i ilustrujące ich możliwe zachowania, wyglądają następująco:



Można udowodnić, że $P \approx Q$.

$$(2). P = a; NIL + b; NIL$$

$$Q = \tau; a; NIL + b; NIL$$



Te procesy nie są równoważne obserwacyjnie, nie są nawet $\approx_1$ -równoważne, ponieważ proces P jest gotowy do akcji τ i b, natomiast proces Q może autonomicznie wykonać akcję wewnętrzną τ , po której oferuje już tylko akcję a.

Na podstawie definicji równoważności obserwacyjnej można udowodnić wiele własności procesów, np. $P+Q \approx Q+P$, $P+NIL \approx P$, $\tau; P \approx P$, które pozwolą na formalne przekształcenia termów reprezentujących równoważne procesy. W praktyce znacznie ważniejsza jest jednak możliwość przeprowadzania lokalnych przekształceń, zachowujących równoważność. Zauważmy, że $\tau; NIL \approx NIL$, ale nieprawda, że $P+\tau; NIL \approx P+NIL$, ponieważ proces $P+\tau; NIL$ może autonomicznie wykonać τ -akcję, po której przekształci się w proces pusty NIL, nie równoważny $P+NIL$. Zdefiniujemy więc relację kongruencji obserwacyjnej $\approx_c$ następująco:

$$P \approx_c Q \stackrel{\text{def}}{=} \text{dla każdego kontekstu } \mathcal{C}[\] \text{ zachodzi } \mathcal{C}[P] \approx \mathcal{C}[Q].$$

Nie jest możliwa rekursywna aksjomatyzacja kongruencji obserwacyjnej w CCS, posiadająca własność zupełności, niżej zostaną jednak podane niektóre własności kongruencji.

- | | |
|---|--|
| (C1) $P+(Q+R) \approx_c (P+Q)+R$ | (C2) $P+Q \approx_c Q+P$ |
| (C3) $P+P \approx_c P$ | (C4) $P+NIL \approx_c P$ |
| (C5) $P+\tau; P \approx_c \tau; P$ | (C6) $m; (P+\tau; Q) \approx_c m; (P+\tau; Q)+m; Q$ |
| (C7) $m; \tau; P \approx_c m; P$ | (C8) $P NIL \approx_c P$ |
| (C9) $P (Q R) \approx_c (P Q) R$ | (C10) $NIL \setminus U \approx_c NIL$ |
| (C11) $(P+Q) \setminus U \approx_c P \setminus U + Q \setminus U$ | (C12) $m; P \setminus U \approx_c m; (P \setminus U)$ jeśli $m, \bar{m} \in U$ |
| | $m; P \setminus U \approx_c NIL$ jeśli $m, \bar{m} \notin U$ |

$$(C13) P|Q \approx_c \sum_{i=1}^n m_i; (P_i|Q) + \sum_{j=1}^n n_j; (P|Q_j) + \sum_{m_i=\bar{n}_j} \tau; (P_i|Q_j)$$

$$\text{gdzie } P = \sum_{i=1}^n m_i; P_i, \quad Q = \sum_{j=1}^n n_j; Q_j,$$

$$\sum_{i=1}^n m_i; P_i = m_1; P_1 + \dots + m_n; P_n, \quad \sum_{i=1}^0 m_i; P_i = NIL.$$

Ta własność wyraża fakt, że w CCS wykorzystywany jest 'przeplotowy' model współbieżności.

Oczywiście $\approx_c \subseteq \approx$. Mamy także

(R14) $\tau; P \approx P$ (R15) $\text{rec } X; \tau; X \approx \text{NIL}$

5. SPECYFIKACJA I WERYFIKACJA PROSTEGO PROTOKOŁU

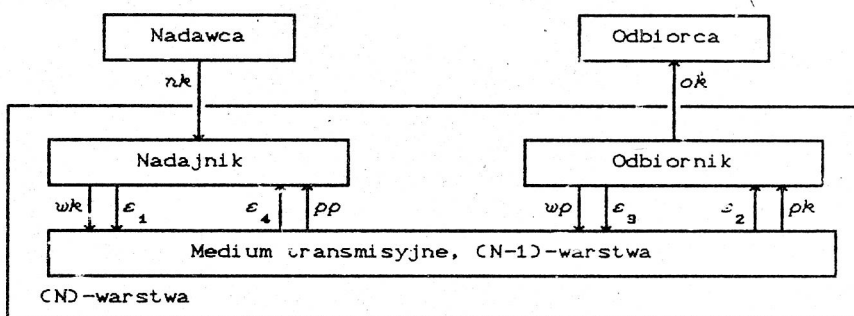
5.1. Specyfikacja

Protokół kodyfikuje zasady postępowania komunikujących się jednostek. W systemach rozproszonych jego głównym celem jest zapewnienie niezawodnej transmisji przy użyciu zawodnego medium (kanału przesyłowego, podsieci komunikacyjnej).

Standardowy model systemów otwartych ISO/OSI jest warstwowy, każda warstwa jest dostarczycielem usług dla warstwy wyższej. Wewnątrz (ND)-warstwy działają niezależne procesy, nazywane jednostkami protokołowymi, które komunikują się zgodnie z protokołem (ND)-warstwy, wykorzystując usługi (N-1)-warstwy. (ND)-warstwa jest dostarczycielem usług dla (N+1)-warstwy.

Formalny opis (ND)-warstwy zawiera więc trzy specyfikacje:

- (1) specyfikację usług świadczonych przez (ND)-warstwę,
- (2) specyfikację wykorzystywanych usług, tj. usług świadczonych przez (N-1)-warstwę,
- (3) specyfikację protokołu (ND)-warstwy.



Rys.2. Architektura (ND)-warstwy

Fig.2. The architecture of (ND)-layer

W poniższym przykładzie zostanie naszkicowana metodologia specyfikacji i weryfikacji protokołów komunikacyjnych oraz będą zarysowane niektóre problemy. Przytoczone automaty skończone ułatwią zrozumienie specyfikacji, przedstawianych w postaci termów CCS. W celu zwiększenia czytelności porty wejściowe będą oznaczane "a?", porty wyjściowe zaś "a!".

"Architektura" specyfikowanej warstwy jest przedstawiona na rys.2. Jest to uproszczona, asynchroniczna wersja protokołu, opisanego w [1].

Specyfikacja świadczonych usług

W rozpatrywanym przykładzie (ND)-warstwa ma za zadanie przesyłanie komunikatów, wysyłanych przez nadawcę przez port nk (nadanie komunikatu) do odbiorcy, który odbiera je przez port ok (odbiór komunikatu). W danym czasie może być przesyłany tylko jeden komunikat. Świadczone usługi można więc wyspecyfikować za pomocą procesu

$$S = \text{rec } X.nk?; ok!; X$$

któremu odpowiada następujący automat

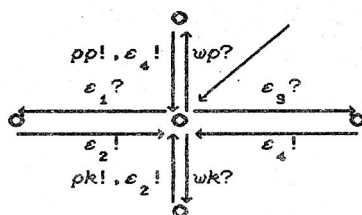


Specyfikacja wykorzystywanych usług

Komunikacja odbywa się za pośrednictwem zawodnego medium, które może gubić komunikaty i potwierdzenia (nie może ich zniekształcać ani powielać), wysyłając zamiast nich sygnał "pusty" ϵ , przesyłany zawsze jako ϵ (nie jest on gubiony). Medium przyjmuje komunikat przez port wk (wysłanie komunikatu) i przesyła go do odbiornika przez port pk (przyjęcie komunikatu) lub gubi komunikat i zamiast niego przesyła sygnał pusty przez port ϵ_2 . Potwierdzenie przyjęcia komunikatu przyjmowane jest przez port wp (wysłanie potwierdzenia) i przekazywane do nadajnika przez port pp (przyjęcie potwierdzenia) lub gubione i zastępowane sygnałem pustym, wysyłanym przez port ϵ_4 . Sygnały puste, wysyłane przez nadajnik przez port ϵ_1 są przesyłane bez zmian przez port ϵ_2 do odbiornika i odwrotnie, sygnały wysyłane przez odbiornik przez port ϵ_3 są odbierane bez zmian przez nadajnik przez port ϵ_4 . Wykorzystywane usługi specyfikuje proces

$$M = \text{rec } X.(wk?; (pk! + \epsilon_2!) + wp?; (pp! + \epsilon_4!) + \epsilon_1?; \epsilon_2! + \epsilon_3?; \epsilon_4!); X$$

któremu odpowiada automat



Specyfikacja protokołu

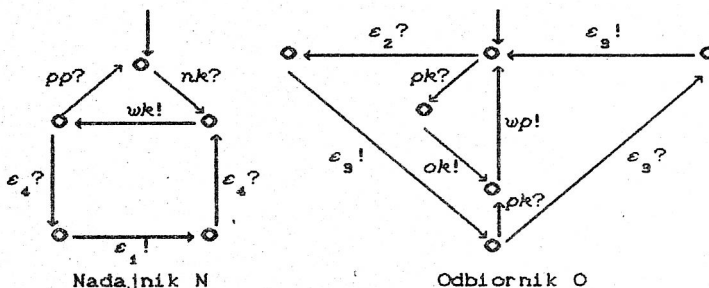
Zadaniem protokołu jest zapewnienie niezawodnej transmisji komunikatu od nadawcy do odbiorcy przy użyciu wyspecyfikowanego wyżej zawodnego medium. Odbiornik oczekuje parzystej liczby symboli pustych między kolejnymi komunikatami; nieparzysta liczba symboli pustych

poprzedza retransmisję komunikatu. Zachowanie nadajnika N i odbiornika O można wyspecyfikować za pomocą następujących procesów

$$N = \text{rec } X. nk?; wk!; (\text{rec } Y. pp?; X + \varepsilon_4?; \varepsilon_1!; \varepsilon_4?; wk!; Y)$$

$$O = \text{rec } X. pk?; ok!; wp! X + \varepsilon_2?; \varepsilon_3!; (\text{pk}?; wp!; X + \varepsilon_2?; \varepsilon_3!; X)$$

lub odpowiadających im automatów:

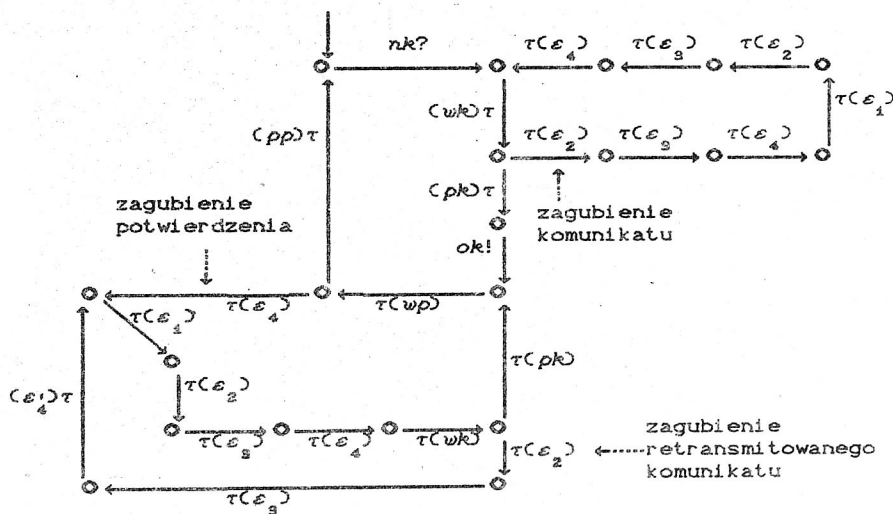


Następujący proces specyfikuje protokół

$$P = (CN \mid M \mid O) \setminus \{wk, pk, wp, pp, \varepsilon_1, \varepsilon_2, \varepsilon_3, \varepsilon_4\}$$

5.2. Weryfikacja

Formalną specyfikację protokołu można wykorzystać do rozmaitych celów [3], m.in. do jego weryfikacji, polegającej na udowodnieniu zgodności protokołu ze specyfikacją świadczonych usług. W naszym



przypadku oznacza to udowodnienie równoważności $S \approx P$. Wykorzystując własność (C13) można udowodnić kongruencję $P \approx_c P'$, gdzie

$P' = \text{rec } V.nk?$;

$(\text{rec } X.\tau; (\tau^7; X+\tau; \text{ok!}; (\text{rec } Y.\tau; (\tau; V+\tau; (\text{rec } Z.\tau^5; (\tau^8; Z+\tau; Y))))))$

Zapis τ^n oznacza n-krotne wykonanie akcji τ .

Zachowanie procesu P' ilustruje powyższy automat. Dla ułatwienia w nawiasach pokazano interakcje wewnętrzne, na które środowisko nie ma wpływu (co symbolizuje akcja τ).

Można teraz udowodnić $S \approx P'$. Zauważmy jednak, że proces P' może wykonywać nieskończone ciągi akcji wewnętrznych τ , co odpowiada ciągłemu gubieniu przez medium komunikatu lub potwierdzenia. Protokół P posiada wszystkie własności bezpieczeństwa (p.3.4), które zechcemy sformułować ale nie daje gwarancji, że wysłany komunikat dotrze do odbiorcy (własność żywotności). Rzeczywiście, w dowodzie równoważności $S \approx P'$ trzeba korzystać z własności (R15), która utożsamia proces zakleszczony (ang. *deadlock*) z rozbieżnym (ang. *divergence*). Problem ten można rozwiązywać rozmaicie, np. zakładając bezstronność (ang. *fairness*) w obsłudze procesów.

6. UWAGI KOŃCOWE

Protokół komunikacyjny ostatecznie jest implementowany, zwykle jako zbiór programów. Zarysowana wyżej metodologia traktuje protokół jako system reaktywny, specyfikowany za pomocą termów CCS, które również można potraktować jako aplikatywne programy abstrakcyjne. Można ją więc stosować na wszystkich etapach tworzenia oprogramowania, od specyfikacji S_1 do ostatecznej implementacji S_n (p.1.2). W powyższym przykładzie można dokonać kolejnego uściślenia, precyzując wewnętrzną strukturę medium, np. przedstawiając je jako dwa kanały półdupleksowe NO (nadajnik-odbiornik) i ON (odbiornik-nadajnik), a potem udowodnić kongruencję $M \approx_{\text{NO|ON}} \text{NO|ON}$. Używając do specyfikacji sieci Petriego czy automatów skończonych, zmuszeni jesteśmy do stosowania na różnych etapach różnych metodologii.

Z zarysowaną w przykładzie metodologią projektowania systemów reaktywnych wiąże się duże nadzieje na przyszłość. Jądem jest tu język CCS Milnera, a dokładniej odpowiednie algebry procesów komunikujących się. W pracy pominięto związek CCS z logikami programów, w szczególności logikami modalnymi i temporalnymi, które wyznaczają komplementarne, dwujęzykowe podejście do projektowania systemów.

LITERATURA

- [1] AHO A.V., ULLMAN J.D., YANNAKIS M., *Modelling communications protocols by automata*, 20th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 1979.
- [2] BARCZYŃSKI J., HUZAR Z., SPŁAWSKI Z., *Wstępna analiza problemów*

specyfikacji i weryfikacji oprogramowania sieciowego, Centrum Obliczeniowe, Politechnika Wrocławska, Raport SPR 7/87, 1987.

- [3] BOCHMANN G.V., i inni, *Use of formal specifications for protocol design, implementation and testing*, w [30], 1985.
- [4] BRINKSMA E., SCOLLO G., STEENBERGEN C., *LOTOS specifications, implementations and their tests*, w [23], 349 - 360, 1987.
- [5] BRINKSMA E. (red.), *Information processing systems. Open systems interconnection. LOTOS - a formal description technique based on temporal ordering of observational behaviour*, ISO 8807.
- [6] BROOKS Jr. F.P., *No silver bullet - essence and accidents of software engineering*, w [17], 1069 - 1076, 1986.
- [7] BROOKS S.D., ROUNDS W.C., *Behavioural equivalence relations induced by programming logics*, w [11], 97 - 108, 1983.
- [8] BROOKS S.D., ROSCOE A.W., WINSKEL G. (eds), *Seminar on concurrency*, Lecture Notes in Computer Science, vol. 197, Springer, 1985.
- [9] DE NICOLA, *Transition systems and testing preorders: an alternative to Petri nets for systems specification*, w [17], 833 - 836, 1986.
- [10] DE ROEVER W.-P., *Questions to Robin Milner - a responder commentary*, w [17], 515 - 518, 1986.
- [11] DIAZ J. (ed), *Automata, languages and programming*, Lecture Notes in Computer Science, vol. 154, Springer, 1983.
- [12] EHRIG H., MAHR B., *Fundamentals of algebraic specification. Part I: Equations and initials semantics*, Springer, 1985.
- [13] GOGUEN J.A., *One, none, a hundred thousand specification languages*, w [17], 995 - 1003, 1986.
- [14] HOARE C.A.R., *Communicating sequential processes*, Prentice Hall International, 1985.
- [15] HUZAR Z., KUŹNIARZ L., *LOTOS - język formalnych specyfikacji systemów reaktywnych*, Centrum Obliczeniowe, Politechnika Wrocławska, Raport SPR 25/88, 1988.
- [16] JONES C.B., *Specyfikacje a programy(1)*, Informatyka, nr 10, 1-4, 1987.
- [17] KUGLER H.-J. (ed), *Information Processing 86*, Participants edition, North Holland, 1986.
- [18] MAIBAUM T.S.E., *Role of abstraction in program development*, w [17], 135 - 142, 1986.
- [19] MILNER R., *Process constructors and interpretations*, w [17], 507 - 514, 1986.
- [20] MILNER R., *Lectures on calculus for communicating systems*, w [8], 206 - 228, 1985.
- [21] OREJAS F., *Response to "The role of abstraction in program development"*, w [17], 143 - 146, 1986.
- [22] PNUELI A., *Specification and development of reactive systems*, w [17], 845 - 858, 1986.
- [23] SARIKAYA B., BOCHMANN G.V. (eds), *Protocol specification, testing and verification*, VI, Elsevier Science Publisher, 1987.
- [24] SAJKOWSKI M., *Protocol verification techniques: status and perspectives*, w [30], 697 - 720, 1985.

- [25] SPŁAWSKI Z., *Formalne metody specyfikacji i konstrukcji oprogramowania sieciowego - problemy i założenia*, Centrum Obliczeniowe, Politechnika Wrocławska, Raport SPR 15/87, 1987.
- [26] SPŁAWSKI Z., *Teoriiodowodowe podejście do programów funkcyjnych i typów danych*, Centrum Obliczeniowe, Politechnika Wrocławska, Raport SPR 16/88, 1988.
- [27] STROIŃSKI M., *Metody formalnej specyfikacji protokołów oraz ich zastosowania*, Prace Naukowe 5, Konferencje 3, Centrum Obliczeniowe, Politechnika Wrocławska, 131 - 149, 1987.
- [28] TARLECKI A., *Software system development - an abstract view*, w [17], 685 - 688, 1986.
- [29] TURSKI W.M., *And no philosophers' stone, either*, w [17], 1077 - 1080, 1986.
- [30] YEMINI Y., STROM R., YEMINI S. (eds), *Protocol specification, testing and verification, IV*, Elsevier Science Publishers B.V., 1985.

- NETWORK SOFTWARE DEVELOPMENT - METHODOLOGICAL AND THEORETICAL PROBLEMS

A number of issues concerning the concepts of software development are overviewed. Possible interpretations of the notions of specification and implementation are surveyed, together with their interconnections, properties and proving techniques. Three distinct methodological approaches to program development and possible decisions concerning specification languages are pointed out. Some of these problems are discussed for reactive systems and are exemplified by the specification and verification in CCS of a simple communication protocol.

РАЗРАБОТКА СЕТЕВОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ - МЕТОДОЛОГИЧЕСКИЕ И ТЕОРЕТИЧЕСКИЕ ПРОБЛЕМЫ

Обозреваются главные проблемы разработки программного обеспечения. Рассматриваются возможные интерпретации понятия спецификации и реализации, их взаимосвязи, свойства и техники доказательства. Указываются три различные методологические подходы к разработке программ, а также возможные решения, касающиеся языков спецификации. Некоторые из перечисленных проблем обсуждаются в контексте реактивных систем и поясняются примером спецификации и верификации в исчислении взаимодействующих систем (CCS) простого сетевого протокола.

sieci lokalne, współbieżność

Wacław IRZĘSKI *

WARSTWA SIECIOWA ROZPROSZONEGO SYSTEMU DYDAKTYCZNEGO

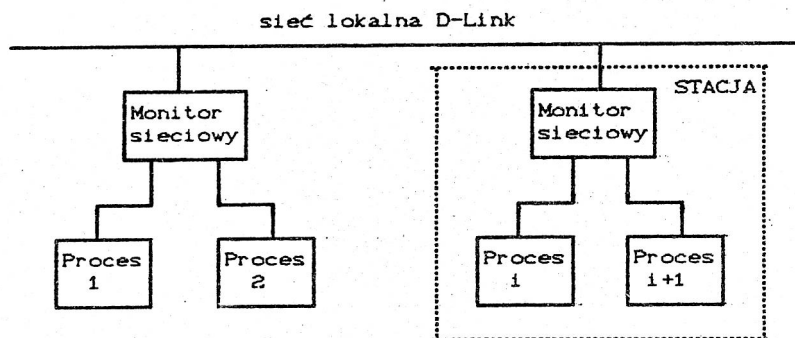
W referacie zaprezentowano warstwę sieciową realizowanego rozproszonego systemu dydaktycznego umożliwiającą komunikację między współbieżnymi procesami. Procesy te mogą działać na różnych mikrokomputerach typu IBM PC XT/AT połączonych siecią lokalną D-Link. Przedstawiono budowę oraz spełniane funkcje tej części systemu.

1. Wprowadzenie

Międzyuczelniany Zespół Informatyczny Politechniki Rzeszowskiej i Wyższej Szkoły Pedagogicznej w Rzeszowie realizuje w ramach RPI-09 rozproszony system dydaktyczny. System zaimplementowano na mikrokomputerach IBM PC XT połączonych siecią lokalną D-Link. Schematycznie przedstawiono go na rysunku 1. Założono, że w systemie może działać kilka procesów współbieżnych. Ich synchronizację zapewniają odpowiednie, wzajemnie wymieniane wiadomości. Dydaktyczne przeznaczenie systemu implikuje funkcje procesów. Przesyłanie zadań do uczniów, weryfikacja przez nauczyciela aktualnego stanu pracy, dostęp do kartoteki zawierającej "historię postępów ucznia" mogą stanowić przykłady takich funkcji. W referacie zaprezentowano budowę procesu nazywanego dalej monitorem sieciowym (Crys.1). Realizuje on przesyłanie wiadomości synchronizujących między aktywnymi procesami. Oprócz pośredniczenia w wymianie informacji synchronizujących między procesami prezentowana część systemu dydaktycznego oferuje także pewne funkcje usługowe, z których mogą korzystać procesy.

* Politechnika Rzeszowska

Zakład Automatyki i Informatyki

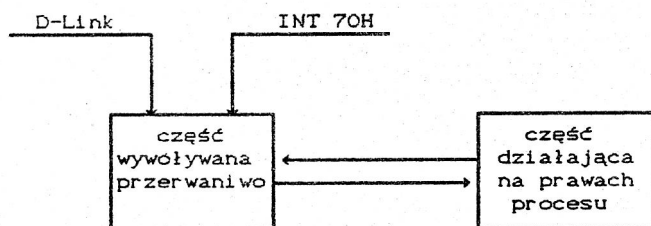


Rys. 1. Organizacja rozproszonego systemu dydaktycznego

Fig. 1. Organization of the distributed education system

2. Ogólna charakterystyka monitora sieciowego

Monitor sieciowy zbudowany jest z dwóch głównych części. Ilustruje to rysunek 2.



Rys. 2. Budowa monitora sieciowego

Fig. 2. Network monitor structure

Część pierwsza, wywoływana przerwaniwo służy do odbioru nadchodzących wiadomości, ich analizy i zapamiętywania. Tutaj również podejmowane są decyzje związane z przerywaniem, zawieszaniem, aktywowaniem lub usuwaniem procesów lokalnych. Asynchroniczne, za pośrednictwem przerwań przekazywanie sterowania do tej części monitora odbywa się w dwojaki sposób. Może być ono spowodowane przerwaniem programowym (INT 70H), w wypadku gdy nadawcą jest proces lokalny. Gdy wiadomość nadejdzie za pośrednictwem sieci - nadawcą jest w tym wypadku proces zdalny, monitor sieciowy wywoływany jest w przerwaniu zegarowym po stwierdzeniu, że wewnętrzny bufor D-Link (wersja 3.24) jest wypełniony [1]. W zależności od tego, czy odbiorcą jest proces lokalny lub zdalny wiadomość jest

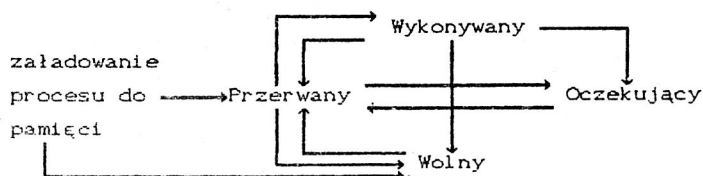
zapamiętywana, ewentualnie wysyłana za pośrednictwem sieci do innego mikrokomputera. Założono, że monitor sieciowy przechowuje informacje o aktualnym stanie systemu. Dowolny proces może zwrócić się do monitora, przysyłając do niego odpowiednią wiadomość o te informacje. Obsługę takich żądań realizuje druga, przedstawiona na rysunku 2 część monitora sieciowego.

3. Współpraca procesów z monitorem sieciowym

Przyjęto, że monitor sieciowy każdej ze stacji przechowuje pełną informację o działających w systemie procesach. Informacje te zawarte są w dwóch odpowiednio aktualizowanych tablicach. Pierwsza z nich o strukturze przedstawionej poniżej dotyczy procesów lokalnych:

```
struct
{
    char nazwa[3];
    char stan;
    char rezerwa;
    unsigned pkt_powr[14];
    unsigned pkt_star[14];
    unsigned pkt_konc[14];
} t_pr_lok[8];
```

Wypełniana jest ona podczas ładowania procesu do pamięci, gdy realizowana jest specjalna sekwencja inicjująca. Z każdym lokalnym procesem związany jest rekord zawierający nazwę procesu (kod ASCII), aktualny stan procesu oraz trzy zestawy zawartości wszystkich rejestrów mikroprocesora 8086. W danym momencie proces może przebywać w jednym z czterech stanów [2]. Sposób zmian stanów procesów ilustruje rysunek 3.



Rys. 3. Stany procesu

Fig. 3. Process states

Drugą wspomnianą tablicą monitora sieciowego, służącą do ewidencji aktywnych procesów jest tablica o następującej strukturze:

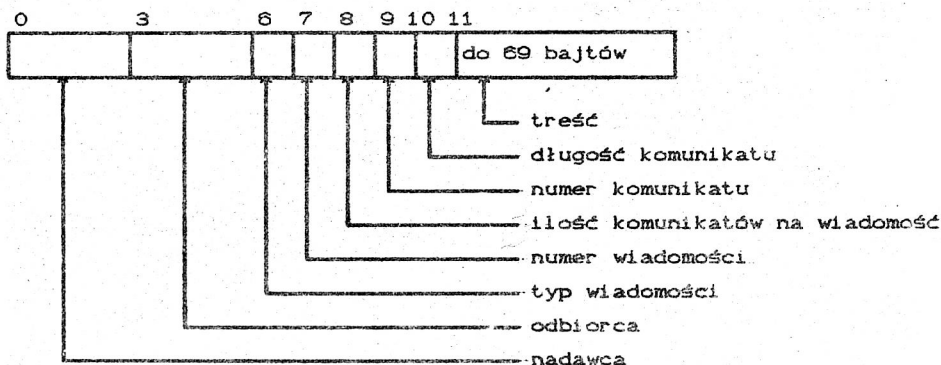
```
struct
(
    char nazwa[3];
    int nr_stacji;
    t_pr_zda[20];
```

Przechowywane są w niej informacje dotyczące procesów zdalnych, działających na innych niż rozważany monitor sieciowy mikrokomputerach.

Wszelkie akcje monitora sieciowego podejmowane są w oparciu o informacje pamiętane w tych tablicach. Wobec tego istotna jest integralność pamiętanych tutaj informacji. Zapewnia to odpowiednia organizacja ładowania procesu do pamięci operacyjnej i jego z niej usuwania.

4. Obsługa komunikatów, usługi monitora sieciowego

W systemie nie narzucono ograniczenia na długość przesyłanych wiadomości. Przyjęto więc, że może się ona składać z kilku komunikatów o formacie jak na rysunku 4.



Rys. 4. Akceptowany przez monitor sieciowy format komunikatu

Fig. 4. Valid message format

Wprowadzono następujące typy komunikatów:

- 'K': uruchom załadowany proces
- 'W': wyślij i wstrzymaj nadawcę aż do otrzymania odpowiedzi
- 'O': odpowiedź na wysłany wcześniej komunikat typu 'W'
- 'S': komunikat specjalny

Monitor sieciowy stacji na której znajduje się proces-odbiorca przechowuje nadchodzące komunikaty. Używa do tego specjalnej, dynamicznie przydzielanej struktury danych pozwalającej również na kompletowanie komunikatów tworzących wiadomość.

W obecnej wersji monitor sieciowy oferuje funkcje usługowe pozwalające na:

- a) załadowanie procesu do pamięci
- b) załadowanie procesu do pamięci i jego uruchomienie
- c) zatrzymanie wykonywanego procesu
- d) wznowienie wykonywania zatrzymanego uprzednio procesu
- e) zakończenie procesu
- f) zakończenie procesu i usunięcie go z pamięci
- g) określenie czy włączono daną stację do sieci
- h) stwierdzenie czy dany proces jest załadowany

5. Uwagi implementacyjne, podsumowanie

Monitor sieciowy zaimplementowano w języku C, używając kompilatora Lattice wersja 2.14 [3]. Pewna fragmenty, związane z wykorzystaniem funkcji systemu operacyjnego i standardowego oprogramowania sieci lokalnej D-Link napisano w języku wewnętrznym mikroprocesora 8088. Wykorzystano przy tym makroassembler MASM-wersja 4.0 [4]. W fazie uruchamiania oprogramowania korzystano także z programu narzędziowego "Advanced Trace 86".

Monitor sieciowy, w zaprezentowanym w referacie zakresie został uruchomiony i przetestowany. W obecnej wersji reakcją monitora na niespodziewane odłączenie stacji od sieci, odebranie wiadomości dla nieznanego odbiorcy itp jest wyświetlenie odpowiedniej informacji na ekranie i zakończenie pracy. Zapewnienie bardziej złożonej reakcji na tego typu zdarzenia jest głównym kierunkiem aktualnie realizowanej rozbudowy monitora sieciowego.

6. Literatura

1. "D-Link Network Program User's Manual"
2. Hansen P. B., "Podstawy systemów operacyjnych", WNT Warszawa, 1979
3. Kernighan B. W., Ritchie D. M., "Język C", WNT Warszawa, 1987
4. "Microsoft Macro-86 Macro Assembler Manual"

NETWORK LAYER OF THE DISTRIBUTED EDUCATION SYSTEM

In the following is presented network layer of the distributed education system. The system consist of some mikrocomputers IBM PC/XT attached to D-Link Local Area Network. Concurrent processes transmit and receive synchronizing messages with the network layer. Particular attention is paid to showing structure and functions of the presented education system part.

computer network, transfer
scheduling, mail service

Roman KASZUBA*

Jan KWIATKOWSKI*

MESSAGE TRANSFER SCHEDULING

A computer network creates the natural environment for distributed processing. In such an environment a management function ordering the utilization of the connection resource is needed. The paper deals with some issues which should be considered when designing the application layer entity. It is described using a mail service as an example.

1. INTRODUCTION

The computers and their interconnection facilities provide for many contemporary services, among which the most popular one is the electronic mail. It is popular not only within the computer network users community but also on the lines, i.e. majority of the traffic there is caused by transferred mail messages. This heavy traffic requires appropriate procedures for the most efficient exploitation of computer and communications resources to be developed. The paper presents some solutions undertaken in the implementation of the file transfer service [2] in the Interuniversity Computer Network (ICN) connected with arranging resource assignment for individual transfers. These are also applicable to the mail message transfer as its execution is similar to that of ordinary files. The mail service differs only in much greater volume of transfers and lower time requirements. But the entities providing the mail service constitute the same competitive environment as file transfer stations. Then the implemented mechanism in the file transfer service would serve for the mail service, especially that the intention is to use the existing file transfer service for secure message transfer at the first step of a mail service implementation.

2. DESIGN OVERVIEW

The competitive environment is composed of application entities

* Computer Centre, Tech. Univ. of Wrocław, Wrocław

maintaining queues of message transfers. Each message transfer is executed in cooperation with the remote entity implied by a routing algorithm. The entity sending the message controls the transfer and is responsible for its reliable execution. There exists the asymmetry in the communication between the two performing transfer entities: one side plays the role of initiator, the other one the role of responder. Any entity can maintain several activities, each independent in message transfer executions of the others. All message transfers are independent and may be executed in any order, but any possible order should satisfy the user demands on expected transfer time. Since each system has limited resources, then all message transfers can not be executed concurrently and there has to be a management function determining the order in which the message transfers are to be performed. The local strategy adopted by an entity must lead to the completion of all message transfers from its queue and at the same time must ensure resources for the entity cooperation in transfers controlled by the others. The ways of such an arrangement are discussed.

The application entity activity would operate as the initiator (if it is controlling a transfer from its local queue), the responder (if it is cooperating in a transfer controlled by a remote entity), or the listener (if it is waiting for a connection request from a remote entity) or would be an idle entity. In each entity there are sets of: initiator activities (I), responder activities (R) and listener activities (L).

The message transfers in the local queue require, for their execution, cooperation of a subset of remote community entities, called connectivity domain (C).

The management function controls the activities operation in terms of the preference order in which transfers are taken for execution and the assignment of the roles to the activities (also activities creation/deletion). This function would take into account such factors as: present activities sets (I), (R), (L), the present operational state of the the community and entities in connectivity domain (C), the properties of the transfers in the local queue (e.g. volume of data to be transferred), etc.

Each attempt to execute a transfer requiring the cooperation of a given remote entity supplies information about that entity: if a connection has been established, then the entity is accessible; if the connection establishment request has been refused, then the entity is temporary/permanently inaccessible. In the case of refusal the reason code indicating a place and a cause of refusal is been supplied. Both these information are important for scheduling of transfers. The

persistence property of the reason indicates for how long the transfer execution should be delayed or that the transfer should be aborted as unexecutable. The typical reason codes are network/remote system down, network/remote network service access point congested, application busy, transfer not executable, etc.

3. STRATEGIES OF TRANSFER MANAGEMENT

Strategy adopted in the ICN is based on the principle that the resource assigned to an application activity in the state of being a listener is negligible. Then there are two different kinds of activities: real activities, which are able to enter any operational state (initiator, responder, listener), and dummy activities, which are able to enter the listener state only. In the case when all real activities are busy the reply "application busy" to the subsequent connection request is infinitely delayed, as the connection request is directed to a dummy activity. When any of busy real activities is freed the first of the awaiting reply connection requests is reassigned to it. That is, the connection requests are queued and served in the FIFO discipline. It is left to the decision of the initiator if it waits for remote application resource (blocking the dummy listener, the connection and own activity) or cancels the request to execute another transfer. Of course only few of connections can be in such a waiting state in a system (limited queue length), other connection requests are refused with reply "TSAP congested", and the average time of obtaining this response is determined to enable setting of outstanding connection request time-out reasonably. Simple initiator strategy is such that if the initiator connectivity domain (C) is not empty, the initiator will wait for connection confirmation for a period of time, then, if without success, it will give up and will take another transfer or, if its connectivity domain is empty, it will wait "almost" infinitely.

However, at the first look one could envisage a drawback in the solution, as the initiator activity, the listener and the connection are blocked, but in fact this resource consumption is less than active use of the same resources by continuous polling (strategy commonly employed). Moreover, further benefit of the solution is obtained due to other entities having received "TSAP congested" reply are sure that the queue is full, so it is useless to poll this entity for some time. Also one could think that in the case when a waiting initiator cancels the connection request, releasing the remote listener, there is a decrease in the system performance because the listener becomes free and unused (the other entities, having received "TSAP congested" reply, became engaged with different listeners, so they refrain from polling this

one). But this is negligible, as only dummy activities perform the role of listeners and there are still more occupied listeners in that remote end system.

To eliminate the feature of the resource blocking described above the improvements in management strategy were made. The first improvement is connected with the enrichment of information obtained from the failed connection attempts (localization of the refusing entity and diagnostic code). This information enable the management function to classify the reason on one of the four categories: transient, short term, long term, and permanent; and to determine the subset of the (C) domain which is inaccessible. That indicates the subset of transfers excluded from short term scheduling.

The second improvement is connected with the support of the reservation mechanism, which enables to save resources. Namely the connection and the remote listener are not blocked if the same course of action as in the previous strategy is assumed. The operation scheme differs in that the listener records the requesting initiator in its local queue instead of delaying a negative response, and then almost immediately rejects the connection request releasing itself and the connection. Therefore the initiator, when issuing the connection request, is almost sure that it will connect to a dummy entity at least. If the initiator connects to the real activity, the transfer execution will start immediately, if not, then having received the negative response it will wait for a reverse call from the called entity for a period corresponding to the period of waiting for the connection confirmation in the previous strategy. If this period expires it will begin the same procedure with the next transfer. The reverse call will be received in the same situation as in the first solution the connection would be accepted, that is, when at the called remote entity a responder activity will be freed to serve this transfer. The original call will be repeated if the reverse call is not received within a relatively long period of expected arrival. If the reverse connection request is refused the recorded request will be treated as other transfers from the called entity local queue, that is, reverse call will be repeated according to scheduling.

One could notice that the resource wastage is not necessarily lower than that in the previous strategy. This is because the refused connection request record is not limited as the queue was, then it is needed to wait for a reverse call much longer than previously for the connection confirmation. This will decrease the probability for the reverse call to succeed if the initiator entity is well loaded with connection requests, so the scheme is likely to stop working (becomes

similar to ordinary polling). Moreover, there are no underlying mechanism ensuring the acceptance of the connection request from a given entity when any other appears at the same time, so the reverse call can be refused even if the waiting activity is free. At the end note that sometimes it would be the resource wastage to refuse another call due to uncertainty of the reverse call arrival within a short time. Then in fact this reverse call mechanism benefits in a different way - it increases the (C) domain of the called entity, whose scheduling function distributes the entity connection requests on its domain. Briefly speaking the community connectivity scheme is such that the overcalled entities first realize message transfers with less occupied ones, and next the message transfers with other "popular" ones, but in fact the exact scheme depends on scheduling undertaken by these function to "popular" entities. To support the reverse call a special command was added to the application protocol: the command setting the assignment of initiator and responder roles of both connected entities.

4. CONCLUSIONS

The presented above procedures are not completely satisfactory and should be further improved. Some concepts are discussed in [1], but that paper is still connected with general file transfer, so the presented concepts do not accomodate the specific properties of message transfer. Moreover the mail service is expected to involve lots of personal computers which accessability is very low, what should be also taken into consideration while improving the presented functions.

LITERATURE

- [1] Kaszuba R, Some Issues on Management of Application Connections Establishment, Microprocessing and Microprogramming, North-Holland vol 23, 1988, pp. 373 - 378.
- [2] Kaszuba, R.M., and Kwiatkowski, J., Experience in file and job service implementation using the British "coloured books" protocols, in: Boyanov, K. (ed.), Networks in office automation (North-Holland, Amsterdam, 1985, ISBN: 0-444-87715-0) pp 39-45.
- [3] UK High Level Protocol Group, Network Independent File Transfer Protocol, (CNIFTP-BC80), revised by the FTPIG of the DCPUD.

SZEREGOWANIE KOMUNIKATÓW DO TRANSFERU

Sieć komputerowa tworzy naturalne środowisko dla rozproszonego przetwarzania zadań. W środowisku takim niezbędne są funkcje zarządzające kolejnością wykonywania się poszczególnych transferów. W pracy przedstawiono pewne rozwiązania które mogą być wykorzystane przy projektowaniu stacji poziomu aplikacji. Rozważania przeprowadzono na bazie usługi poczty elektronicznej.

УПРАВЛЕНИЕ ПЕРЕДАЧЕЙ СООБЩЕНИЙ

Вычислительная сеть образует естественную среду для передачи сообщений и их обработки. В этой среде необходимы функции управления физическими соединениями. В статье представлены некоторые решения, которые могут быть использованы в проектировании станции передачи сообщений.

Piotr KOCIATKIEWICZ*

sieć komputerowa.
teleprzetwarzanie danych

STAN OBECNY I ROZWÓJ
SYSTEMÓW TELEPRZETWARZANIA DANYCH
W INSTYTUCIE KOMPUTEROWYCH SYSTEMÓW AUTOMATYKI I POMIARÓW

W referacie scharakteryzowano architekturę, sprzęt i oprogramowanie opracowanego w IKSAiP systemu teleprzetwarzania TELE JS-M. Podano możliwości konfiguracyjne sieci komputerowych tworzonych w oparciu o system TELE JS-M. Omówiono główne kierunki rozwoju systemu w zakresie sprzętu i oprogramowania.

1. WPROWADZENIE

Obecnie eksploatowane w wielu dziedzinach gospodarki systemy teleprzetwarzania zapewniają użytkownikom, rozmieszczonym na pewnym obszarze geograficznym, dostęp do zasobów jednego, centralnego systemu komputerowego.

Typowymi przykładami systemów teleprzetwarzania są systemy budowane w oparciu o opracowany w IKSAiP podsystem TELE JS. Podsystem TELE JS jest produkowany i rozpowszechniany przez Zakłady Elektroniczne ELWRO.

Analiza kierunków rozwoju zastosowań systemów komputerowych wskazuje na potrzebę łączenia ich w systemy wielokomputerowe o różnym stopniu sprzężenia. Jedną z dróg do tego prowadzących jest opracowanie sprzętu i oprogramowania, które umożliwia łączenie tradycyjnych systemów teleprzetwarzania z jednym komputerem centralnym w systemy sieciowe z wieloma komputerami centralnymi.

Przedmiotem referatu jest podsystem TELE JS-M. Podsystem TELE JS-M jest zbiorem sprzętu i oprogramowania realizującym elementy architektury sieciowej. W połączeniu z komputerami obliczeniowymi serii Riad-2 lub Riad-3 (JS EMC), które są wyposażone w niezbędne oprogramowanie sieciowe

* Instytut Komputerowych Systemów Automatyki i Pomiarów, Wrocław

(system operacyjny, metody dostępu) służy do tworzenia sieci komputerowych - wielodomenowych systemów teleprzetwarzania.

Sieć komputerowa budowana w oparciu o podsystem TELE JS-M, dalej nazywana systemem TELE JS-M zapewnia użytkownikom współużytkowanie zasobów wielu systemów komputerowych rozmieszczonych na pewnym obszarze geograficznym (miasto, region lub cały kraj). Główną funkcją tego typu systemów jest przyjmowanie, przechowywanie, przetwarzanie, przesyłanie i wydawanie różnego rodzaju informacji.

Obserwowany obecnie rozwój techniki mikrokomputerowej nie zagraża pozycji dużych komputerów i budowanym na ich bazie sieciom komputerowym. Przeciwnie, przewiduje się, że sieci komputerowe będą niezbędnym wsparciem dla rozpowszechnionych systemów mikrokomputerowych, zapewniając im dostęp do baz danych i baz wiedzy, dostęp do mocy obliczeniowej, usług informacyjnych, poczty elektronicznej itp.

2. KONCEPCJA SYSTEMU TELEPRZETWARZANIA TELE JS-M

Ramy niniejszego referatu nie pozwalają na szczegółowe przedstawienie architektury logicznej i komunikacyjnej systemu TELE JS-M. Zatem rozważania zostaną ograniczone do zarysowania koncepcji tego systemu.

Koncepcję systemu TELE JS-M można najłatwiej zrozumieć z punktu widzenia użytkownika końcowego. Termin użytkownik końcowy odnosi się do ludzi bezpośrednio współpracujących z systemem oraz do programów użytkowych (aplikacji) rezydujących w komputerach obliczeniowych i inteligentnych urządzeniach terminalowych. Użytkownik końcowy nie jest częścią systemu, stąd w systemie muszą istnieć mechanizmy stanowiące pomost pomiędzy użytkownikiem końcowym a systemem. Tym pomostem są jednostki logiczne. Jednostki logiczne są implementowane programowo lub mikroprogramowo i są dla użytkowników końcowych punktami dostępu do systemu.

Nawiązanie komunikacji między dwoma użytkownikami końcowymi jest związane z przejściem jednostek logicznych reprezentujących tych użytkowników w stan wzajemnej zależności, który jest nazywany sesją.

Każda jednostka logiczna ma przyporządkowaną nazwę sieciową, w czasie inicjacji sesji system wyznacza adres sieciowy odpowiadający nazwie sieciowej. Taka organizacja wymiany informacji pozwala użytkownikowi końcowemu na komunikację z innym użytkownikiem końcowym bez konieczności wskazania, gdzie żądany użytkownik końcowy się znajduje.

Adresy sieciowe są związane z tak zwanymi sieciowymi jednostkami adresowymi, to jest zbiorem elementów systemu, który dostarcza użytkownikom usług umożliwiających wysyłanie danych przez sieć. Sieciowe jednostki adresowe komunikują się ze sobą przez sieciowe sterowanie drogą (trasą).

Wyróżnia się trzy rodzaje sieciowych jednostek adresowych:

- jednostkę logiczną, omówioną wyżej, reprezentującą użytkownika końcowego,
- jednostkę fizyczną, to jest element zapewniający usługi sterowania łączeniami, terminalami i procesorami w systemie,
- punkt sterowania usługami systemowymi, to jest zbiór elementów zarządzających całym systemem lub jego częścią.

Podobnie jak między jednostkami logicznymi, również między pozostałymi sieciowymi jednostkami adresowymi może być nawiązana komunikacja (sesja).

Wymienione wyżej elementy systemu TELE JS-M są realizowane programowo lub sprzętowo w urządzeniach, z których jest zbudowany system. Urządzenia te mogą być zlokalizowane na pewnym obszarze geograficznym zgodnie z zapotrzebowaniem na usługi przetwarzania danych. Własności konfiguracyjne systemu określają tak zwane węzły. Są to komputery obliczeniowe, procesory sieciowe i urządzenia terminalowe. Każdy węzeł posiada własną jednostkę fizyczną, reprezentującą jego zasoby w sieci. Istnieją następujące kategorie węzłów:

- węzeł komputera obliczeniowego - węzeł sterujący komponentami systemu we wszystkich pozostałych węzłach,
- węzeł procesora sieciowego - węzeł pracujący pod kontrolą węzła komputera obliczeniowego, sterujący liniami komunikacyjnymi i urządzeniami zewnętrznymi,
- węzeł peryferyjny - węzeł skojarzony z urządzeniem terminalowym lub jednostką sterującą grupą urządzeń terminalowych.

Fizycznie węzły są łączone ze sobą w różny sposób. Może to być kanał jednostki centralnej lub linia telekomunikacyjna.

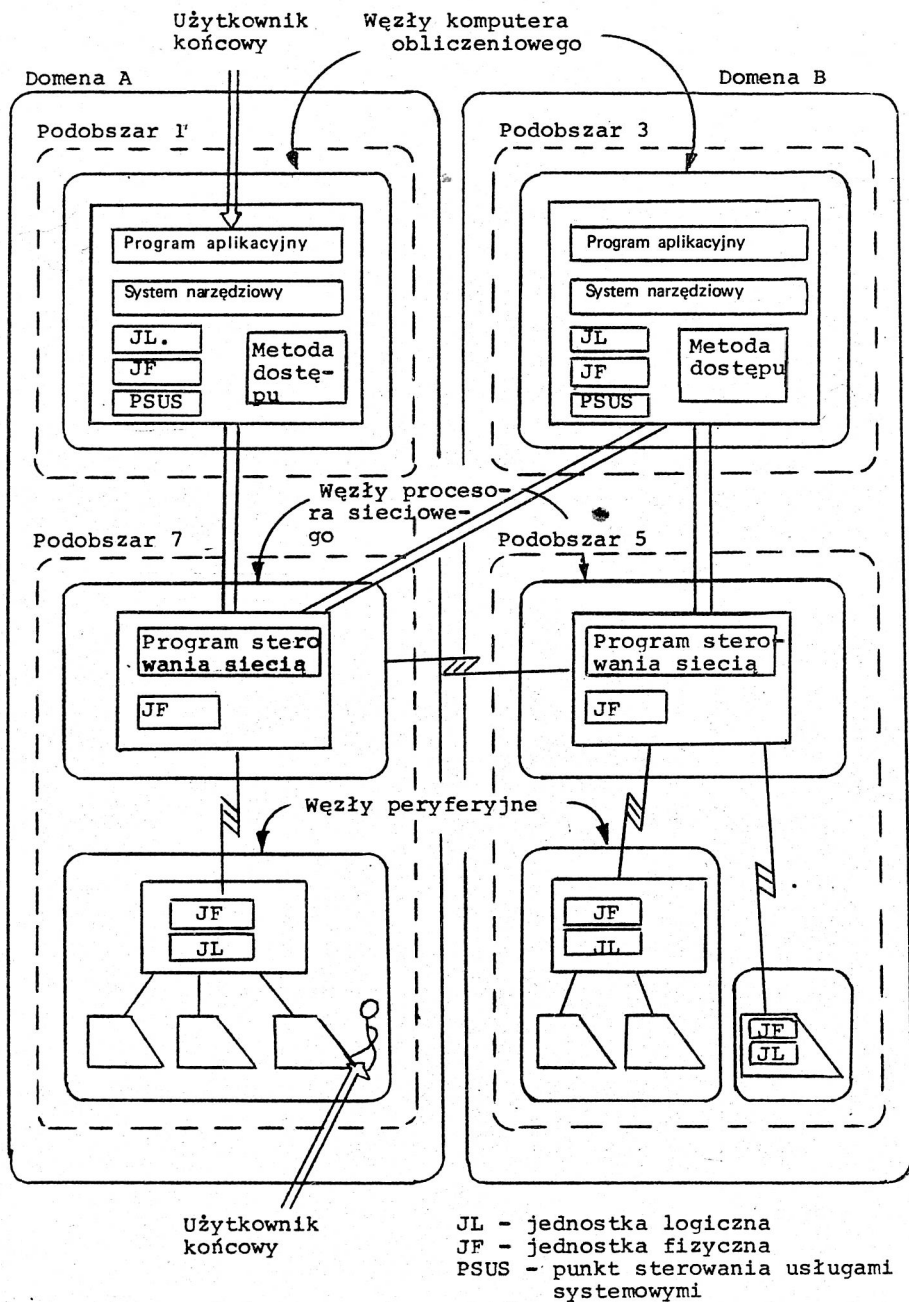
Komponenty systemu są grupowane w podobszary. Każdy podobszar składa się z węzła komputera obliczeniowego lub węzła procesora sieciowego z węzłami peryferyjnymi podłączonymi do niego - każdy podobszar ma własny numer służący do identyfikacji komponentów sieci.

Węzeł podobszaru posiadający punkt sterowania usługami systemowymi wraz z innymi węzłami podobszarów sterowanymi przez ten punkt tworzą domene.

Podczas sesji dane są przesyłane trasami. Trasę tworzą węzły i linie komunikacyjne. Między dwoma użytkownikami końcowymi mogą być tworzone trasy alternatywne.

Relacje między elementami systemu TELE JS-M przedstawiono na rys.1.

Sieciowe sterowanie drogą jest odpowiedzialne za transport komunikatów pomiędzy siecią jednostką adresową nadającą, a siecią jednostką adresową odbierającą.



Rys. 1. Relacje między elementami systemu TELE JS-M

Fig. 1. Relation between elements of TELE JS-M

Format komunikatów jest następujący:

komunikat::= <sja:o,sja:n,parametry,dane>

sja: o - sieciowa jednostka adresowa odbierająca,

sja: n - sieciowa jednostka adresowa nadająca.

Sieciowe sterowanie drogą składa się z dwu komponentów: sterowania drogą i sterowania łączem. Są one odpowiedzialne za zarządzanie przepływem komunikatów wzdłuż całej trasy łączącej zaangażowane sieciowe jednostki adresowe.

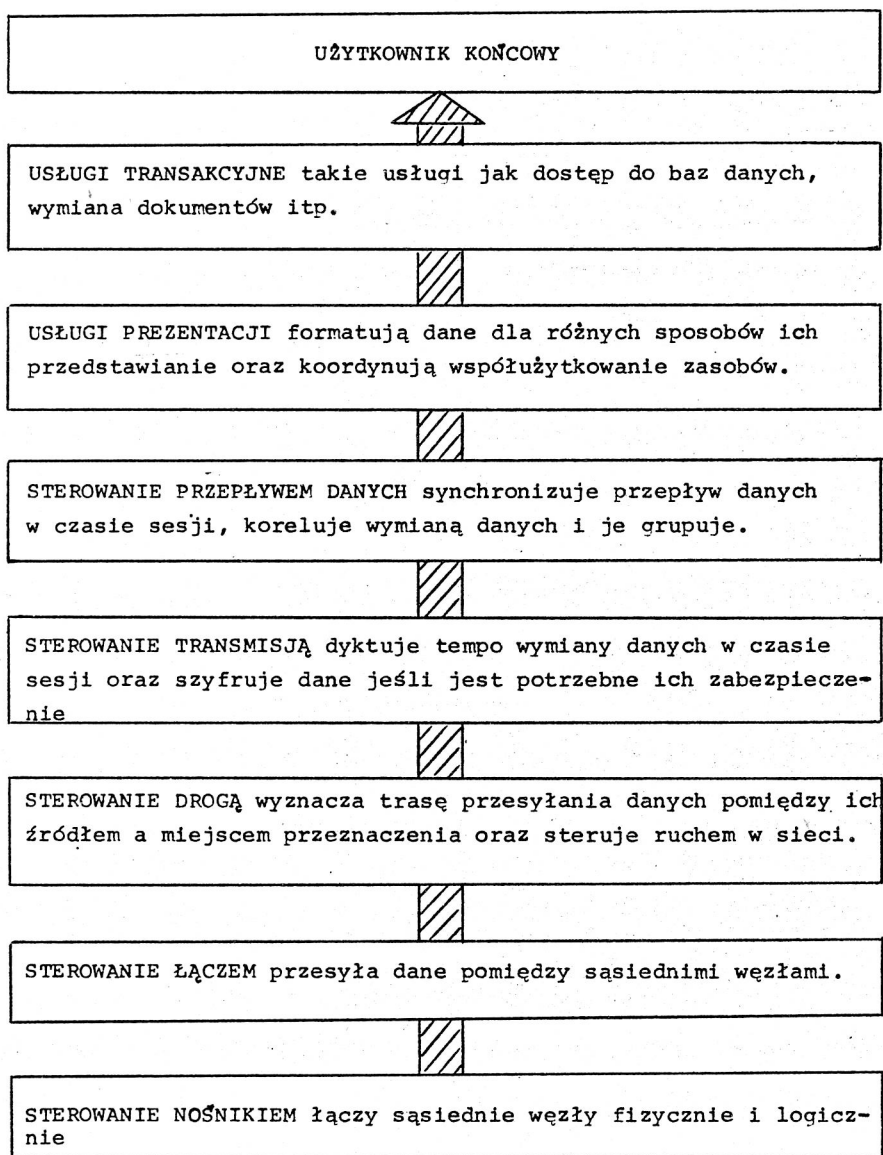
Zbiór zdefiniowanych dla systemu TELE JS-M funkcji jest podzielony na hierarchiczne warstwy, począwszy od warstwy nośnika fizycznego a kończąc na warstwie usług transakcyjnych. Rys. 2 ilustruje warstwy i funkcje przez nie wykonywane.

3. KOMPONENTY SYSTEMU TELE JS-M

Na rys. 3 przedstawiono główne komponenty sprzętowe i programowe systemu. Funkcje poszczególnych komponentów są następujące:

- a) Komputer obliczeniowy - steruje całością lub fragmentem sieci. Zapewnia realizację obliczeń, wykonywanie programów, dostęp do baz danych i usługi zarządzania siecią.
- b) Procesor sieciowy - zarządza fizyczną siecią, steruje łączami i wyznacza drogi przesyłania danych.
- c) Jednostki sterujące - sterują operacjami WE/WY urządzeń terminalowych podłączonych do nich.
- d) Urządzenia terminalowe - zapewniają użytkownikom dostęp do zasobów systemu (wysyłanie i przyjmowanie informacji).
- e) Sieciowe metody dostępu - spełniają następujące funkcje:
 - sterowanie przepływem danych w systemie (na poziomie logicznym),
 - zapewnienie sprzęgu między podsystemami narzędziowymi a siecią,
 - zabezpieczenie podsystemów użytkowych przed nieautoryzowanym dostępem.

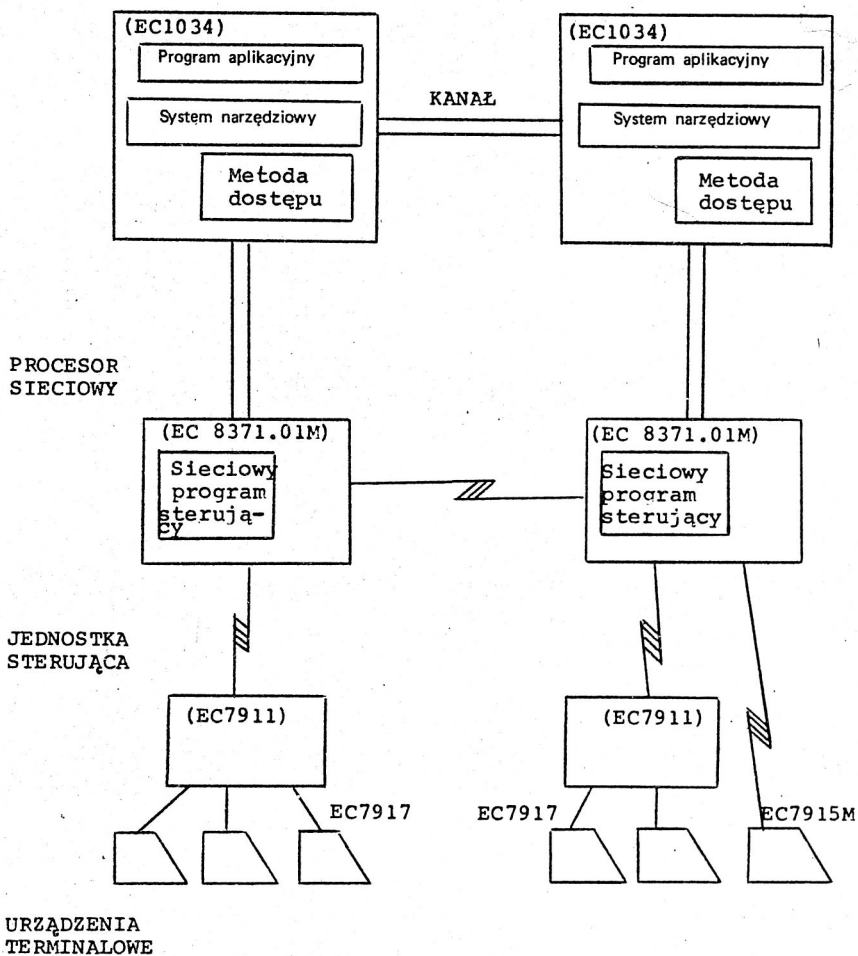
Sieciowe metody dostępu rezydują w komputerze obliczeniowym.
- f) Podsystemy narzędziowe - należą do nich systemy wspomagające takie czynności jak rozwój programów konwersacyjnych, przeglądanie i aktualizacja informacji, zdalne przetwarzanie wsadowe. Przykładami takich podsystemów są SKOT/VS i HADES/VS.
- g) Programy użytkowe - wykonują funkcje wymagane przez użytkowników. Należą do nich obliczenia naukowe, usługi typu transakcji lub edycja tekstów. Programy użytkowe rezydują w komputerach obliczeniowych lub programowych urządzeniach terminalowych.



Rys. 2. Warstwowa struktura systemu TELE JS-M

Fig. 2. Layered structure of TELE JS-M system

KOMPUTER
OBLICZENIOWY



Rys. 3. Komponenty systemu TELE JS-M

Fig. 3. Components of TELE JS-M system

- h) Program sterowania siecią - wyznacza trasy przesyłania danych oraz steruje transmisją. Program sterowania siecią rezyduje w procesorze sieciowym.

4. SPRZĘT SYSTEMU TELE JS-M

W skład zestawu urządzeń systemu TELE JS-M wchodzi następujące urządzenia:

- komputer obliczeniowy,
- procesor sieciowy,
- komunikacyjne urządzenia transmisji danych,
- urządzenia terminalowe.

4.1. Komputer obliczeniowy

Funkcje komputerów obliczeniowych pełnić mogą systemy komputerowe Jednolitego Systemu EMC serii Riad-2 lub Riad-3. Warunkiem stosowania tych systemów jest akceptowanie przez jednostki centralne systemu operacyjnego OS/JS 7.2.

Wymagania odnośnie konfiguracji komputera obliczeniowego i parametrów jednostki centralnej zależą od rodzaju aplikacji eksploatowanych w sieci, wymaganych czasów reakcji, niezawodności itp., to znaczy od sfery użytkowej instalacji.

4.2. Procesor sieciowy

Funkcje procesora sieciowego w systemie TELE JS-M pełni procesor EC 8371.01M. Jest on specjalizowanym komputerem ukierunkowanym na obsługę transmisji danych między komputerem obliczeniowym a urządzeniami terminalowymi. Swoje funkcje wykonuje w oparciu o program sterujący przechowywany w pamięci operacyjnej.

Procesor sieciowy składa się z następujących bloków funkcjonalnych:

- Jednostka sterująca - zarządzająca pracą procesora i organizująca współdziałanie pozostałych bloków.
- Pamięć operacyjna - przeznaczona do przechowywania programów sterujących i danych. Wykonana w technice półprzewodnikowej.
- Adapter kanałowy - obsługuje transmisję kanałową między procesorem a komputerem obliczeniowym. Istnieją dwa typy adapterów kanałowych:
 - AK14 - pracujący z kanałem bajtmultipleksorowym
 - AK 2 - pracujący z kanałem blokmultipleksorowym.
- Skaner komunikacyjny - obsługuje transmisje liniami komunikacyjnymi łączącymi procesor sieciowy z urządzeniami terminalowymi. Opracowano dwa typy skanerów:

SK 2 pracujący w trybie znakowym,

SK 3 pracujący w trybie blokowym.

- Blok liniowy - realizujący nadawanie i odbiór kolejnych bitów znaku oraz sterowanie obwodami styku S2 linii komunikacyjnych.
- Adapter dyskowy AD-2 - moduł sterujący pamięcią na dyskach elastycznych. Pamięć ta jest przeznaczona do autonomicznego ładowania programów testujących lub programu sterującego procesora sieciowego pracującego w trybie zdalnym (ładowanie przez łącza komunikacyjne).
- Pulpit techniczny - umożliwiający obsługę operatorską i szybką diagnostykę stanów awaryjnych procesora.
- System zasilania - zasilający wszystkie moduły procesora.

Opisane wyżej bloki funkcjonalne są montowane w zespoły konstrukcyjne - tak zwane moduły. W zależności od typów i liczby modułów istnieją różne warianty konfiguracji procesora. Moduły: jednostka sterująca, pamięć operacyjna, adapter dyskowy, pulpit techniczny i zespół zasilania wchodzi w skład każdej konfiguracji. Pozostałe moduły są opcjonalne. Kolejne wykonania (tzw. konfiguracje) procesora sieciowego scharakteryzowano w Tabeli 1.

Tabela 1

Nr konfigu- racji \ Typ modułu	AK-14	AK-2	SK-3/B0	SK-2/B0	Wersja
1	-	-	1	-	zdalna
2	-	-	-	1	zdalna
3	-	-	1	1	zdalna
4	1	-	1	-	lokalna
5	1	-	-	1	lokalna
6	1	-	1	1	lokalna
7	-	1	1	-	lokalna

Nr konfigu- racji \ Typ modułu	AK-14	AK-2	SK-3/B0	SK-2/B0	Wersja
8	-	1	-	1	lokalna
9	1	1	1	-	lokalna
10	1	1	-	1	lokalna
11	1	2	1	-	lokalna
12	1	1	1	1	lokalna
13	-	2	1	-	lokalna
14	-	2	-	1	lokalna
15	-	2	1	1	lokalna

Podstawowe parametry techniczne procesora sieciowego są następujące:

Pojemność pamięci operacyjnej	≤ 512 KB
Cykl instrukcji	1,6 μs
Maksymalna liczba linii telekomunikacyjnych	128
Liczba podłączonych kanałów	4 (jednoczesna praca 2)
Tryb pracy	S/S, BSC, SDLC

4.3. Komunikacyjne urządzenie transmisji danych

Podstawowy zestaw urządzeń transmisji danych systemu TELE JS-M stanowią następujące urządzenia:

- modemy EC 8006 i EC 8013
- konwertery: telegraficzny TGF
podstawowy KP 4800 bps
- autowzywak telefoniczny EC 8062

Mogą być również stosowane inne urządzenia spełniające wymagania CCITT.

4.4. Urządzenia terminalowe

Podstawowy zestaw urządzeń terminalowych systemu TELE JS-M obejmuje:

- system monitorów zdalnych:

- . monitor ekranowy zdalny niezależny EC 7915M
 - . jednostka sterująca zdalna EC 7911
 - . monitor ekranowy zdalny zależny EC 7917
 - . drukarka trwałej kopii EC 7914M
 - punkt abonencki EC 8575M
 - mikrokomputer ELWRO 801AT
- oraz inne urządzenia terminalowe, które spełniają wymagania protokołów komunikacyjnych przyjętych w systemie TELE JS-M.

5. OPROGRAMOWANIE SYSTEMU TELE JS-M

5.1. Oprogramowanie komputera obliczeniowego

Zasadniczym elementem oprogramowania komputera obliczeniowego jest system operacyjny. Dla potrzeb systemu TELE JS-M jest dostarczany system operacyjny OS/JS 7.2.

System TELE JS-M może również pracować z systemem operacyjnym OS/VS1 7.0D. W tym celu należy wyposażyć ten system w sieciową metodę dostępu VTAM/MR, która jest przedmiotem dostaw w ramach podsystemu TELE JS-M.

5.2. Oprogramowanie procesora sieciowego

a) Sieciowy program sterujący NCP/MR/8371

Sieciowy program sterujący NCP/MR/8371 steruje transmisją danych pomiędzy komputerem obliczeniowym a terminalami podłączonymi do procesora sieciowego. Procesory sieciowe mogą być podłączane do komputerów obliczeniowych kanałowo lub liniami telekomunikacyjnymi przez inne procesory sieciowe. Program NCP/MR/8371 może być wygenerowany w trzech postaciach:

- dla pracy w trybie emulacyjnym (EP),
- dla pracy w trybie sieciowym (NCP),
- dla pracy w trybie dzielonym (PEP).

Funkcje programu można sklasyfikować w dwóch grupach: funkcje standardowe i funkcje opcjonalne.

Do funkcji standardowych należą:

- sterowanie transmisją danych,
- obsługa znaków sterujących,
- obsługa buforów,
- sterowanie przepływem danych,
- obsługa łączy równoległych pomiędzy sąsiednimi procesorami sieciowymi,
- odnowa sesji,
- sterowanie łączami danych pomiędzy wieloma procesorami w sieci,
- rejestracja błędów.

Do funkcji opcjonalnych należą:

- przetwarzanie bloków danych,
- odnowa po błędach i diagnostyka obejmująca:

- . powiadomienie o sytuacjach krytycznych,
- . śledzenie adresów pamięci i rejestrów,
- . testowanie urządzeń terminalowych w trybie on-line,
- . testowanie łącza SDLC w trybie on-line,
- . wznowienie po przerwach w transmisji,
- weryfikacja identyfikatorów stacji BSC i SDLC,
- sterowanie dynamiczne.

b) Programy wspomagające

Do programów wspomagających NCP/MR/8371 należą następujące programy:

- assembler procesora sieciowego,
- program ładujący,
- program statycznej fotografii pamięci procesora sieciowego,
- program dynamicznej fotografii pamięci procesora sieciowego.

Assembler procesora sieciowego

Assembler procesora sieciowego jest wykonywany w komputerze obliczeniowym. Służy on do kompilacji programów napisanych w języku assembler procesora sieciowego.

Program ładujący

Program ładujący jest wykonywany częściowo w komputerze obliczeniowym i częściowo w procesorze sieciowym. Program służy do ładowania do pamięci procesora sieciowego programu NCP/MR/8371.

Program statycznej fotografii pamięci procesora sieciowego

Program statycznej fotografii pamięci wprowadza do komputera obliczeniowego obraz pamięci procesora sieciowego. Program posiada dwie opcje:

- ograniczenia obszaru fotografowanej pamięci,
- organizowania formatu wydruku pamięci.

Program dynamicznej fotografii pamięci procesora sieciowego

Program dynamicznej fotografii pamięci wprowadza do komputera obliczeniowego obraz pamięci procesora, nie przerywając wykonywania programu sterującego tego procesora. Program ten może być wykonywany wyłącznie z programem NCP/MR/8371 pracującym w trybie EP lub PEP.

c) Moduł wirtualnego terminala sieciowego - VTO/8371

Zadaniem modułu wirtualnego terminala sieciowego VTO/8371 zapewnić obsługę terminali starego typu w charakterze pełnoprawnych terminali sieciowych systemu TELE JS-M. Moduł ten jest opcjonalnym, funkcjonalnym

rozszerzeniem sieciowego programu sterującego NCP/MR/8371 realizującym symulację obsługi urządzeń i linii start-stopowych. Moduł zapewnia obsługę następujących urządzeń terminalowych:

- EC 8575M,
- T 100 lub T 63

6. PODSTAWOWE KONFIGURACJE SYSTEMU TELE JS-M

W zależności od typu systemu operacyjnego zainstalowanego w komputerze obliczeniowym wyróżnia się dwa warianty konfiguracji systemu TELE JS-M:

- wariant 1 z systemem operacyjnym OS/JS 7.2.
- wariant 2 z systemem operacyjnym OS/VS1 7.0D

Własności obu wariantów zestawiono w tabeli 2.

Tabela 2.

Parametr	Wariant 2	Wariant 1
System operacyjny	OS/VS1 7.0D	OS/JS 7.2
Metoda dostępu	VTAM/MR	TCAM/ANF
Liczba komputerów obliczeniowych	≥ 1	≥ 1
Liczba lokalnie podłączonych procesorów sieciowych	≥ 1	≥ 1
Liczba zdalnych procesorów sieciowych podłączonych do procesora sieciowego lokalnego	≥ 1	1
Liczba procesorów zdalnych podłączonych do procesora zdalnego	≥ 1	0
Liczba połączeń zdalnego procesora sieciowego do innych procesorów sieciowych	≥ 1	1
Łącza grupowe między procesorami sieciowymi zdalnymi	TAK	NIE
Łącza pomiędzy lokalnymi procesorami sieciowymi	TAK	TAK
Rodzaj połączeń pomiędzy lokalnymi procesorami sieciowymi	dowolny	sekwencyjny lub drzewo
Tryb pracy systemów monitorów lokalnych	sieciowy lub lokalny	tylko lokalny

7. ZAKOŃCZENIE

Zaprojektowany w IKSAiP procesor sieciowy EC 8371.01M wraz z opracowanym oprogramowaniem podsystemu TELE JS-M został przebadany i będzie wdrożony do produkcji w ZE ELWRO. Przekazanie podsystemu do dystrybucji nastąpi w 1990 r.

Podsystem TELE JS-M będzie stale rozwijany. Rozpoczęto prace nad podsystemem TELE JS-2, który będzie podstawą tworzenia instalacji sieciowych (systemów TELE JS-2). Koncepcja tego systemu będzie zgodna z opisaną wyżej koncepcją systemu TELE JS-M. Nacisk zostanie położony na polepszenie parametrów technicznych i własności eksploatacyjnych. Złożą się na to:

a. Opracowanie procesora sieciowego nowej generacji (EC 8377). Przewidywane parametry procesora:

- | | |
|---|----------|
| - pojemność pamięci operacyjnej | 0,5+2 MB |
| - liczba podłączonych linii telekomunikacyjnych | ≤ 256 |
| - liczba kanałów do podłączenia jednostek centralnych | 8 |
| - tryb pracy: s/s, BSC, SDLC, X.25 | |

Nowa baza elementowa z obwodami wysokiej skali integracji przyczyni się do osiągnięcia wyższego w porównaniu z EC 8371.01M poziomu niezawodności.

b. Opracowanie oprogramowania ułatwiającego utrzymanie systemu w ruchu. Będą to programy:

- operatorskiego sterowania komunikacją w sieci,
- określania problemów sieci.

c. Opracowanie programu umożliwiającego wykorzystywanie przez system TELE JS-2 autonomicznych sieci transmisji danych z komutacją pakietów z zewnętrznym stykiem zgodnym z rekomendacją X.25.

d. Opracowanie sprzętu i oprogramowania umożliwiającego podłączenie do systemu TELE JS-2 lokalnych sieci komputerowych.

Actual state and development trends of teleprocessing system
designed in the Institute of Computer Systems for Automation and Meas-
urement

In the paper, the TELE JS-M teleprocessing system developed in the Institute of Computer Systems for Automation and Measurement is described. Architecture, hardware, and software of this system are characterized. Some configuration possibilities of computer network based on this system are presented. The development trends of hardware and software for TELE JS system are discussed.

Настоящее состояние и развитие систем телеобработки данных в Инсти-
туте компьютерных систем автоматики и измерений

В статье рассматривается система телеобработки данных ТЕЛЕ ЕС-М. Особое внимание уделяется архитектуре, программному обеспечению и техническим средствам этой системы. Представлены конфигурационные возможности создания вычислительной сети на базе систем ТЕЛЕ ЕС-М вместе с характеристикой направления развития этой системы.

systemy rozproszone, COSID
optymalizacja oprogramowania

Koczur Krzysztof *
Kotulski Leszek *

OPTIMALIZACJA DYSTRYBUCJI OPROGRAMOWANIA W LOKALNYCH SIECIACH KOMPUTEROWYCH

Efektywność pracy systemu współbieżnego zależy między innymi od funkcji przypisania węzłów obliczeniowych lokalnej sieci komputerowej poszczególnym modułom programowym. W pracy sformułowano kryterium pozwalające teoretycznie porównać efektywność poszczególnych funkcji przypisania. Sformułowano też założenia systemu sugerującego bardziej "efektywne" przypisanie na podstawie analizy wybranych parametrów pracy systemu współbieżnego.

1. WSTEP

W ramach prac nad systemem wspomagania implementacji oprogramowania współbieżnego w lokalnych sieciach komputerowych (COSID [4,5]) pojawił się problem takiej alokacji modułów programowych, aby praca sieci była możliwie efektywna. Kryterium pomiaru stopnia efektywności systemu współbieżnego może być suma czasów bezczynności procesorów obsługujących lokalną sieć komputerową [4]. W systemach rozproszonych należy uwzględnić dodatkowe opóźnienie wynikające z komunikacji pomiędzy odległymi węzłami obliczeniowymi i wtedy ważnym kryterium analizy efektywności pracy sieci komputerowej jest częstotliwość wymiany informacji pomiędzy węzłami obliczeniowymi. Dla uniknięcia strat związanych z czasem oczekiwania na realizację tych transmisji należałoby dążyć do lokalizacji najczęściej współpracujących ze sobą modułów programowych w tym samym węźle obliczeniowym. W pracy przedstawiono założenia systemu DYNOPT wspomagającego procedury alokacji w celu uzyskania możliwie efektywnej pracy systemu współbieżnego. Zdefiniowano funkcję efektywności systemu oprogramowania. Dla potrzeb systemu DYNOPT w czasie działania systemu współbieżnego obliczane są wybrane parametry

* Katedra Informatyki UJ Kraków

statystyczne pracy systemu współbieżnego. Formalnym opisem alokacji modułów programowych w sieci komputerowej jest EDG-graf. W grafie tym etykietowane wierzchołki oznaczają komponenty sieci (węzły obliczeniowe, moduły programowe), zaś etykietowane krawędzie oznaczają relacje pomiędzy komponentami sieci (alokacja modułu w węźle obliczeniowym, połączenia pomiędzy węzłami, współpracujące moduły itd.). Z każdym z wierzchołków możemy też związać określone atrybuty. Dokładny opis wraz ze szczegółowymi przykładami można znaleźć w [3,4].

Przyjęto, że zmianie alokacji modułów programowych odpowiada rekonfiguracja EDG-grafu. Rekonfiguracja ta odbywa się pod kontrolą zmodyfikowanej gramatyki edNIC [1] przy pomocy systemu ASNET [2]. Problem optymalnej dystrybucji oprogramowania w lokalnych sieciach komputerowych jest problemem bardzo złożonym, dlatego autorzy traktują zaproponowany system DYNOPT jako próbę znalezienia pewnych kryteriów pozwalających uczynić systemy oprogramowania współbieżnego bardziej efektywnymi.

2. FUNKCJA EFEKTYWNOŚCI SYSTEMU PROGRAMOWANIA WSPÓLBIEŻNEGO

Z każdym wierzchołkiem grafu alokacji związany jest atrybut coop (cooperation probability). Każda krawędź (v, c, w) (oznacza to krawędź łączącą moduły współpracujące ze sobą) atrybutowana jest liczbą $B_{(v, c, w)}^{(v, c, w)}$ z przedziału $[0, 1]$.

Funkcję efektywności systemu S z alokacją G w czasie Δt określamy jako

$$FE(S, G) = \sum_{i \in \text{Nodes}} (\text{ready}_i + \rho * (1 - \text{activ}_i : \Delta t) * W(i))$$

gdzie:

- Nodes - oznacza zbiór węzłów obliczeniowych systemu S
- ready_i - suma czasów bezczynności aktywnych modułów w węźle i w rozpatrywanym przedziale czasu Δt
- activ_i - iloraz sumy wykorzystanego czasu procesorów w węźle i przez liczbę procesorów dostępnych w tym węźle
- ρ - opóźnienie wymiany informacji w systemie rozproszonym.

$$W(i) = \sum_{v \in M_i} \sum_{w \in I} \text{coop}_v * B_{(v, c, w)} * \text{Exter}(i, w)$$

gdzie

- coop_v - jest wartością atrybutu wierzchołka v
- $B_{(v, c, w)}$ - wartością atrybutu krawędzi (v, c, w)
- $\text{Exter}(i, w)$ - przyjmuje wartość 0 jeżeli moduł pośredniczący jest zaalokowany w węźle i oraz 1 w przeciwnym przypadku
- M_i - zbiór procesów i managerów zaalokowanych w węźle i
- I - zbiór modułów pośredniczących systemu

Niech K oznacza zbiór wszystkich grafów alokacji systemu S .

$$K = \{G_i, i = 1, \dots, n \mid G_i \neq G_j \text{ dla } i \neq j, G_i \text{ graf alokacji dla } S\}$$

W tym przypadku jako kryterium efektywności systemu przyjmujemy funkcję

$$\min_{G \in K} \{FE(S, G)\}$$

Duża złożoność obliczeniowa powoduje, że przedstawione powyżej kryterium efektywności ma znaczenie bardziej teoretyczne.

3. PARAMETRY STATYSTYCZNE PRACY SIECI KOMPUTEROWEJ

Dla sieci S z n węzłami obliczeniowymi i p modułami oraz grafem alokacji G modułów programowych w trakcie pracy systemu ustalane są wartości następujących zmiennych:

t_i - czas wykorzystania procesorów w i -tym węźle obliczeniowym ($i=1 \dots k$)

k_i - średnia długość kolejki gotowych modułów i -tym węźle obliczeniowym

$C_{(v,c,w)}$ - ilość wołań na krawędzi (v,c,w) .

g_i^j - czas oczekiwania gotowego modułu j w i -tym węźle obliczeniowym

Wtedy

$activ_i = t_i : lp_i$ ($i=1 \dots k$, lp_i oznacza liczbę procesorów w węźle i)

$$ready_i = g_i^1 + \dots + g_i^n$$

$coop_{v,w} = \left(\sum_{c \in V} C_{(v,c,w)} \right) / \left(\sum_{x \in V} C_{(x,c,w)} \right)$ gdy brak połączenia pomiędzy wierzchołkami v, w to $C_{(v,\gamma,w)} = 0$

$$B(v,c,w) = C_{(v,c,w)} / \left(\sum_{x \in V} C_{(x,\gamma,w)} \right)$$

Wartości tych zmiennych posłużą jako parametry dla systemu DYNOPT.

4. PODSTAWOWE ZAŁOŻENIA SYSTEMU DYNOPT

Dla systemu S po upływie odcinka czasu Δt_i ($i=0,1,\dots,k$ za każdym razem identycznego) uruchamiany jest system DYNOPT. Z każdym odcinkiem czasu związany jest graf alokacji G_i . Jeżeli po upływie czasu Δt_i $FE(S, G_i) \leq FE(S, G_{i-1})$ ($i > 0$, w czasie Δt_0 ustalana jest początkowa wartość $FE(S, G_0)$) to przywracana jest konfiguracja systemu wyznaczona przez graf G_{i-1} , w przeciwnym przypadku podejmowana jest akcja rekonfiguracji systemu. Główne założenie systemu DYNOPT jest takie, że moduły komunikujące się najczęściej ze sobą powinny być zlokalizowane w tym samym węźle obliczeniowym. Niech graf G_i będzie grafem alokacji po czasie Δt_i . Znajdujemy krawędź $C_{(v,c,w)}$ (gdzie v, w należą węzłów obliczeniowych $1, k$ $k \neq 1$) o maksymalnej wartości, taką, że możliwa jest alokacja modułu v w węźle obliczeniowym k . W węźle obliczeniowym k znajdujemy moduł z dla którego $\sum_{x \in k} C_{(z,c,x)}$ jest najmniejsza. Moduł z alokujemy w węźle obliczeniowym j wybranym z tych węzłów obliczeniowych $\{i_1, \dots, i_s\}$ w których możliwa jest alokacja modułu z , aby iloczyn $lp_j * (\Delta t_i - t_j) * k_j$ był najmniejszy spośród wszystkich $lp_m * (\Delta t_i - t_m) * k_m$ dla $m=1 \dots i_n$ $m \neq k$ (lp_t - liczba procesorów w węźle t).

Drugim kryterium rekonfiguracji systemu jest poprawa wykorzystania czasu pracy procesorów i zmniejszenie długości kolejek gotowych modułów. Zakładamy, że raz relokowane moduły nie podlegają powtórnej relokacji podczas tego samego przebiegu systemu DYNOPT. Spośród wszystkich węzłów obliczeniowych $\{1, \dots, m\}$ wybieramy węzeł obliczeniowy k dla którego wartość iloczynu $lp_k * (\Delta t_1 - t_k) * k_j$ jest największa spośród wszystkich iloczynów $lp_m * (\Delta t_1 - t_m) * k_m$ i węzeł l dla którego iloczyn $lp_l * (\Delta t_1 - t_l) * k_l$ jest najmniejszy. Z węzła k wybieramy moduł r możliwy do zaalokowania w węźle l dla którego wartość g_k^r jest największa i alokujemy w węźle l .

5. LITERATURA

- [1] Flasiński M.: Parsing of edNLC-graph grammar for scene analysis. Pattern Recognition, 1988, no 3.
- [2] Flasiński M, Hliniak G: Realization of the system controlling distributed software allocation.
przyjęte do Universitas Jagiellonica Scientarium Litterarumque Schedulae Informaticae.
- [3] Flasiński M., Kotulski L.: On the use of graph grammar for the control of a distributed software allocation.
przyjęte do druku w Computer Journal.
- [4] Flasiński M, Kotulski L: On the Linguistic Approach to Controlling a Distributed Software Allocation.
W materiałach konferencji "Computer Networks 89".
- [5] Kotulski L: System wspomagania implementacji oprogramowania współbieżnego w lokalnych sieciach komputerowych.
Raport KI UJ Kraków 1988.
- [6] Kotulski L: A two level approach to the implementation LAN's software.
W przygotowaniu.

Эффективность работы параллельной системы зависит от того как мы распределим программы в вычислительной сети. В статье мы тоже показали критерий который разрешает сравнить эффективность этого распределения. Предлагаем тестировать, некоторые параметры параллельного процесса. Разрешает его предложить метод который предлагает "лучше" распределить программы в вычислительную сеть.

Efficiency of the concurrent system works depends on the function describing associations among the computing nodes of a local area network and program modules. The theoretical criterion which allows one to compare the efficiency of these functions is presented in this paper. A few parameters of the concurrent system work are proposed to be chosen and tested. This allows one to suggest a more "efficient" software distribution

Jerzy KONORSKI*

sieci pakietowe,
zakleszczenia,
modele kolejkowe

PRZEPUSTOWOŚĆ SIECI PAKIETOWEJ Z DZIAŁAJĄCYM MECHANIZMEM UNIKANIA ZAKLESZCZEŃ

W pracy przeprowadzono analizę jakości wybranych mechanizmów unikania zakleszczeń w sieciach pakietowych z punktu widzenia przepustowości sieci. Zastosowano metody sieci kolejkowych z ograniczeniami populacji i przyjęto jednorodny model stochastyczny sieci, pozwalający zmniejszyć nakłady obliczeniowe i nie zakładać określonej topologii sieci. Do obliczeń wykorzystano zmodyfikowany algorytm spłotowy Reisera-Kobayashi. Przedstawiono wyniki numeryczne ilustrujące rolę doboru mechanizmu kontroli przeciążeń przy ocenie mechanizmu unikania zakleszczeń.

1. WPROWADZENIE

Zakleszczenia (store-and-forward deadlocks) są potencjalną konsekwencją stosowania protokołów przesyłania pakietów między węzłami z zachowaniem kopii pakietu wysłanego i retransmisją pakietu odrzuconego przez węzeł sąsiedni i były badane już w latach 70-tych. Systematyczna konstrukcja mechanizmów unikania zakleszczeń (MUZ) rozpoczęła się u progu lat 80-tych [1]. Wspólną cechą MUZ jest możliwość odrzucania pakietów przez węzeł mimo niecałkowicie zapełnionej pamięci buforowej, co na ogół prowadzi do obniżki przepustowości sieci. Postulat minimalizacji wspomnianej obniżki oznacza poszukiwanie możliwie liberalnych MUZ (tj. z łagodnymi warunkami akceptacji pakietu); liberalność definiowana jest przy tym w terminach zbiorów charakterystycznych funkcji decyzyjnych opisujących MUZ w poszczególnych węzłach [3,6,9]. Przydatność tego deterministycznego kryterium oceny MUZ nie zawsze znajduje potwierdzenie w analizie stochastycznej (por. [8]). W niniejszej pracy zanalizujemy sieć pa-

* Instytut Telekomunikacji Politechniki Gdańskiej

kietową z MUZ działającym równolegle z mechanizmem kontroli przeciążeń (MKP) celem zademonstrowania wpływu MKP na przepustowość sieci z działającym MUZ, a w szczególności tego, że niewłaściwy dobór MKP czyni kryterium liberalności MUZ bezwartościowym lub nawet zwodniczym.

2. WYBRANE MECHANIZMY UNIKANIA ZAKLESZCZEN

W tym punkcie omówimy krótko wybrane MUZ będące przedmiotem dalszej analizy. W dwóch pierwszych decyzja o akceptacji/odrzućeniu pakietu przez węzeł oparta jest na rozpoznaniu klasy lokalnej pakietu, określanej jako liczba kanałów sieci pozostających pakietowi do przebycia przed osiągnięciem węzła docelowego (ang. hop-count). MUZ oparte na krążących tokenach definiują specjalny rodzaj pozwoleń (tokenów), które mogą krążyć między węzłami bądź samodzielnie (jako wiadomości kontrolno-sterujące), bądź w nagłówkach pakietów, nie podlegając zniszczeniu ani powieleniu (dla dokładniejszych opisów i dowodów poprawności por. [3]). Niech H będzie maksymalną długością trasy pakietu, zaś B liczbą buforów w węźle (każdy bufor mieści 1 pakiet). Ponadto przez x_h oznaczmy liczbę pakietów klasy lokalnej h znajdujących się aktualnie w pamięci buforowej węzła ($0 \leq h \leq H$), zaś przez f - liczbę aktualnie wolnych buforów w węźle. Rozłączne Pule Buforów (RP) [1,6]. Wybieramy liczby naturalne $B_0, \dots, B_H$ takie, że $B_0 + \dots + B_H = B$. Pakiet klasy lokalnej h zaakceptowany zostaje, gdy $x_h < B_h$.

Badanie Zawartości Pamięci Buforowej (BZ) [9]. Pakiet klasy lokalnej h zostaje zaakceptowany, gdy dla każdego $k = 0, \dots, h$ zachodzi $f + x_0 + \dots + x_{k-1} > k$. Łatwo pokazać, że dla dowolnych $B_0, \dots, B_H$ mechanizm BZ jest bardziej liberalny niż RP (w istocie jest on najbardziej liberalnym mechanizmem typu hop-count).

Jeden Krążący Token (KT1) [2,3]. Pakiet klasy lokalnej $h=0$ zostaje zaakceptowany przy $f > 0$, zaś klasy lokalnej $h > 0$ - gdy $f > 1$, bądź, przy $f=1$, gdy $x_0 > 0$ lub gdy token znajduje się w węźle bądź w nagłówku pakietu będącego przedmiotem decyzji. Można pokazać [2], że mechanizm KT1 jest bardziej liberalny niż BZ; dotyczy to również mechanizmów KT2, KT3 niżej.

Dwa/Trzy Krążące Tokeny (KT2, KT3) [2,3]. Zasady akceptacji/odrzućania pakietów są podobne jak dla KT1; przy podejmowaniu decyzji należy dodatkowo sprawdzać liczbę aktualnie niepustych kolejek wyjściowych w węźle (tj. kolejek pakietów skierowanych do poszczególnych węzłów sąsiednich).

Gdy liczba ta wynosi 1 (dla KT2), bądź jest ≤ 2 (dla KT3), uruchomiona zostaje tzw. procedura zapytania-odpowiedzi (PZO) celem ustalenia, czy akceptacja przybyłego pakietu grozi wystąpieniem zakleszczenia. Wymaga się tutaj, aby kierowanie pakietów do węzłów sąsiednich było nieodwołalne.

3. PRZEPUSTOWOŚĆ SIECI Z DZIAŁAJACYM MUZ I MKP

Modelowany jest N-węzłowy fragment sieci, nie przyjmujący ruchu z pozostałej części sieci. Znormalizowane obciążenie zewnętrzne (poissonowsko-wykładnicze) wynosi g i rozkłada się równomiernie na wszystkie NQ kolejek wyjściowych. Przyjmujemy, że PZO prawie zawsze kończy się akceptacją pakietu. Jako MKP zastosujemy ograniczenie liczby pakietów źródłowych do $B' < B$ w każdym węźle źródłowym [5]. Oznaczając przez a_{ij} udział ruchu pakietów kierowanych na trasy i-kanalowe, które przebywają j kanałów w naszym obszarze ($j \leq i \leq H$), mamy obciążenie każdej kolejki wyjściowej pakietami źródłowymi i tranzytowymi klasy lokalnej h ($1 \leq h \leq H$):

$$r_{hz} = g A_{h,h+1} / \{ NQ [1 - P_t(h-1)] \}, \quad (1)$$

$$r_{ht} = g \sum_{h+1 \leq i \leq H} A_{ih} [1 - P_z(i)] / \{ NQ [1 - P_t(h)] [1 - P_t(h-1)] \}, \quad (2)$$

gdzie $A_{ik} = a_{i,i-k+1} + \dots + a_{ii}$, zaś $P_z(h)$, $P_t(h)$ - prawdopodobieństwa odrzucenia pakietu źródłowego i tranzytowego klasy lokalnej h ; jak wiadomo [4], można je wyrazić poprzez prawdopodobieństwa typu $P(y \in Y_h)$, gdzie $y = (y_{1z} \dots y_{Hz} y_{1t} \dots y_{H-1,t})$ reprezentuje stan węzła, y_{hz} , y_{ht} - liczby pakietów źródłowych i tranzytowych klasy lokalnej h w węźle, zaś Y_h jest pewnym zbiorem stanów węzła, wynikającym z opisu danego MUZ w p. 2 z podstawieniem $y_{hz} + y_{ht} = x_h$. Do obliczenia tych ostatnich prawdopodobieństw potrzebne są współczynniki wielowymiarowego rozwinięcia potęgowej funkcji $V(u) = [1 - \sum_{1 \leq h \leq H} r_{hz} u_{hz} - \sum_{1 \leq h \leq H-1} r_{ht} u_{ht}]^{-Q}$, które można obliczyć przy pomocy algorytmu splotowego [7]. Dzięki jednorodności modelu sieci nakłady obliczeniowe można tu zmniejszyć Q -krotnie, okazuje się bowiem, że [3]

$$v(y) = v^*(y) \binom{B-f+Q}{B-f} / Q^{B-f}, \quad (3)$$

gdzie $v(y)$ - współczynnik rozwinięcia stojący przy $u_{1z}^{y_{1z}} \dots u_{Hz}^{y_{Hz}} u_{1t}^{y_{1t}} \dots u_{H-1,t}^{y_{H-1,t}}$, zaś $v^*(y)$ - analogiczny współczynnik otrzymany przy podstawieniu $Q := 1$, $r_{hz} := Q r_{hz}$, $r_{ht} := Q r_{ht}$. Przepustowość sieci wynosi

$$S = g \sum_{1 \leq h \leq H} A_{h,h+1} [1 - P_z(h)]. \quad (4)$$

W tabeli poniżej przepustowość sieci i obciążenie zewnętrzne znormalizowano względem maksymalnej przepustowości sieci bez działającego MUZ i odpowiadającego jej obciążenia zewnętrznego, S_0 i g_0 ; przyjęto $N=6$, $H=3$, $B=5$, $Q=3$, $a_{ij}=0.2$ ($1 \leq j \leq i$, $2 \leq i \leq 3$). Wyniki pokazują znaczenie właściwego doboru MKP (tj. wartości B') dla zdyskontowania wzrostu liberalności MUZ.

g/g_0 [%]	60	70	80	S/S_0 [%]							
				90	100	110	120	130	140	150	160
RP, $B'=2$	48	51	54	57	58	60	61	62	63	64	65
BZ, $B'=4$	69	76	81	84	86	87	87	87	86	85	83
KT1, $B'=4$	74	82	88	90	89	85	77	69	60	52	46
KT3, $B'=4$	75	84	90	94	95	92	88	82	75	69	63
BZ, $B'=2$	65	71	76	80	83	85	87	88	89	90	90
KT1, $B'=2$	66	72	78	82	86	89	91	92	94	94	95
KT3, $B'=2$	66	73	78	83	86	89	91	93	94	95	96

4. LITERATURA

1. Günther K., Prevention of buffer deadlocks in packet-switched data transport systems, IEEE Trans. Comm., COM-29, 4, 1981, 512-524
2. Konorski J., S/F deadlock prevention: a circulating token approach and performance considerations, Proc. ICC, Toronto 1986, 4.6.1-4
3. Konorski J., Badanie możliwości zastosowania krążących tokenów do przeciwdziałania zakleszczeniom w sieciach pakietowych, Polit. Gdańska 1986
4. Lam S.S., Queueing networks with population size constraints, IBM J. Res. Devel., 21, 7, 1977, 370-378
5. Lam S.S., Reiser M., Congestion control of S/F networks by input buffer limits, IEEE Trans. Comm., COM-27, 1, 1979, 127-133
6. Merlin P.M., Schweitzer P., Deadlock avoidance in S/F networks, IEEE Trans. Comm., COM-28, 3, 1980, 345-360
7. Reiser M., Kobayashi H., Queueing networks with multiple closed chains: theory and computational algorithms, IBM J. Res. Devel., 19, 5, 1975, 283-294
8. Swiderski J., Efficiency of S/F deadlock prevention controllers, Proc. 12th ITC, Torino 1988, 4.4a.5.1-7
9. Toueg S., Ullman J.D., Deadlock-free packet-switching networks, SIAM J. Comput., 10, 3, 1981, 594-611

NETWORK THROUGHPUT UNDER DEADLOCK PREVENTION SCHEMES

Store-and-forward deadlock prevention based on buffer classes and on circulating tokens is considered in the context of network throughput analysis via a Lam-type queueing model. The influence of congestion control parameters on the efficiency of deadlock prevention is pointed out.

ПРОПУСКНАЯ СПОСОБНОСТЬ СЕТИ С АНТИ-ТУПИКОВЫМ МЕХАНИЗМОМ

В статье представлен анализ сети с пакетной коммутацией, в которой применены некоторые анти-тупиковые механизмы, ограничивающие доступ пакетов к памяти блоков в узлах сети. Рассматривается влияние параметров контроля перегрузок на пригодность детерминистского критерия оценки анти-тупиковых механизмов.

protocol implementation,
ESTELLE,
PDL

Hartmut Koenig*
Olaf Krohmann*

AN APPROACH TO AUTOMATED PROTOCOL IMPLEMENTATION

SUMMARY: The paper deals with problems of automated protocol implementation. Among others two questions are discussed especially: how is a protocol implementation to look like and to what extend the implementation process can be automated? Experience with the PDL and the ESTELLE implementation at the TU Magdeburg are presented.

The large gap the protocol developer has to overcome from the specification to the practical realization of protocols has brought forth the desire to automate this process. Therefore, automated protocol implementation has become one of the main topics in the field of protocol engineering. Besides efficiency automated protocol implementation brings further remarkable advantages as the reduction of implementation errors and of incompatibilities between different implementations, better maintenance and independence of a certain implementor /Aggarwal 83/. The experience shows further that automatically generated implementations are better structured than hand-coded ones.

Systems for automated protocol implementation have been reported in literature since the beginning of the eighties. They are mostly based on state-transition specification languages. The principal implementation approach is similar in many cases. The specification text is transformed into a high-level language code, often PASCAL, C or CHILL, which is connected with hand-

* Technical University "Otto von Guericke" Magdeburg Department of Informatics PSF 124 Magdeburg 3010 GDR

coded procedures. The latter usually comprise such components as the process organization applied for the protocol implementation, the buffer and queue management, and the time control as well. The statements about the portion of this part differ between 30 and 60 per cent.

Without doubt automated protocol implementation is still a topic for further research. Among others the following questions are of interest:

- (1) What is the relation between specification and implementation or in other words: how is the implementation to look like?
- (2) To what extent the implementation process can be automated?

The paper will focus on these questions. It presents our experience gathered by the implementation of the specification language PDL /Koenig 86/ which is being adapted now to the implementation of ESTELLE /Koenig 88/, /Krohmann 88/ (see fig. 1).

Automated protocol implementation is connected to a certain implementation solution. That means the designer of the transformation has to work out a well-fitted solution which the user must accept (or not). This solution comprises the manner how the specification structure can be mapped into the implementation, i.e. which process structure will be chosen, which components are included into the run time kernel and others. In our approach (a similar solution you can find in /Serre 86/) we chose a straightforward transformation of the specification into the implementation, i.e. the latter reflects the structure of the PDL description principle (see fig. 2 and 3). In compliance with this solution a PDL-specification is transformed into three processes which handle the interfaces to the upper and lower layer (upper and lower sap handler) and the protocols (protocol handler). The process scheduling itself is part of the run time kernel (see below).

The advantage of this approach consists in a relatively simple realization of the transformator from the specification into the executable code. Moreover the implementation reflects the specification structure thus simplifying later maintenance.

Discussing the question to what degree this process can be automated we must start from the fact that the information contents of a specification text is confined. The representation is very formal abstracting from many details necessary for the implementation such as event management, process structure and buffer organization. Furthermore, the specification usually considers only one (virtuel) connection but a working entity handles simultaneously several ones. All these components can scarcely be derived from the specification. They must be provided by hand. This is the reason why the relations between automatically generated and hand-made code mentioned above will not change principally.

One way to increase the automation degree consists in designing the hand-coded portion in such a manner that it can be applied to many implementations. In our realization we use the implementation environnement developed in /Zschiesche 88/. It contains procedures for process scheduling, buffer and queue organization, and time control. The environnement, which is available on 16- and 32-bit computers, summarizes experience from different hand-coded protocol implementations.

For automated implementation the environnement has been extended by an envelope including the connection and the event management /Koenig 87/. The interface of the extended environnement was designed relatively common. It contains such functions as (guarded) waiting with selective and parallel executing of events, including of events into waiting queues of other processes, opening and closing of connections as well as several functions of the internal kernel which can be used directly.

This approach is now applied to the implementation of ESTELLE /Krohmann 88/. For this we also want to use the realized environnement (may be with a redefinition of the interface). Then both compilers will be applied to practical protocol implementations to optimize the implementation structure.

Summarizing our experience we want to underline that there is a strong need for a broader discussion of protocol implementation problems. The questions discussed above are only two ones of interest here. There is still a too little knowledge about how to implement efficiently protocols. It seems to be necessary to work out a methodology of implementing protocols.

References

- /Aggarwal 85/ Aggarwal,S.; Kurshan,R.P.: Automated Implementation from Formal Specification. in Yemini,Y.; Strom,R.; Yemini,S.(eds): Protocol Specification, Testing and Verification,IV. North Holland Coop., Amsterdam, 1985, pp. 127-136
- /Koenig 86/ Koenig,H.; Heiner,M.: The PDL-System - An Unified Approach to the Specification, Verification and Implementation of Protocols. in Csaba,L.; Tarnay,K.; Szentivanyi,T. (eds.): COMPUTER NETWORK USAGE: Recent Experiences. North-Holland Comp., Amsterdam, 1986, pp.567-582
- /Koenig 87/ Koenig,H.: Kommunikationsprotokolle - Prinzip und Entwicklung. Diss. B, TU Dresden, Informatikzentrum, 1987
- /Koenig 88/ Koenig,H.;Tarnay,K.: A Work-station for Protocol Engineering. Proceedings IFIP '88 Symposium on Computer Networks, Part 2, pp. 213-224, Sofia,1988
- /Krohmann 88/ Krohmann,O.; Koenig,H.: Rechnergestuetzte Implementierung von Kommunikationsprotokollen auf der Basis von ESTELLE. to be published in Wiss. Zeitschrift d. TU Magdeburg
- /Serre 86/ Serre,J.M.; Cerny,E.; Bochmann,G.v.: A Methodology for Implementing High Level Communication Protocols, Université de Montréal, Département d'informatique et de recherche operationelle, Publication #555, 1986
- /Zschiesche 88/ Zschiesche,M.;Pfueller,G;Wiens,B.: Prozess-Supervisor & Pufferverwaltung. Anwendungsbeschreibung für 16- und 32-Bit-Technik. Version 1.2. TU Dresden, Informatikzentrum, Juli 1988

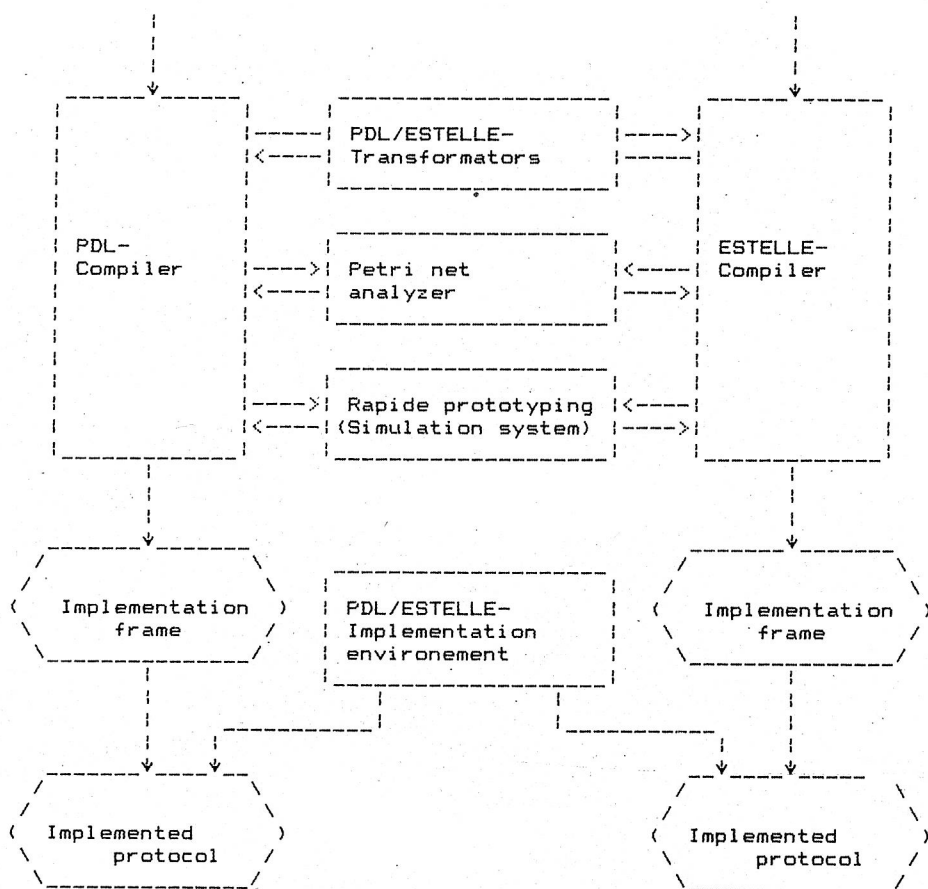


Fig. 1: Structure of the work-station under realization

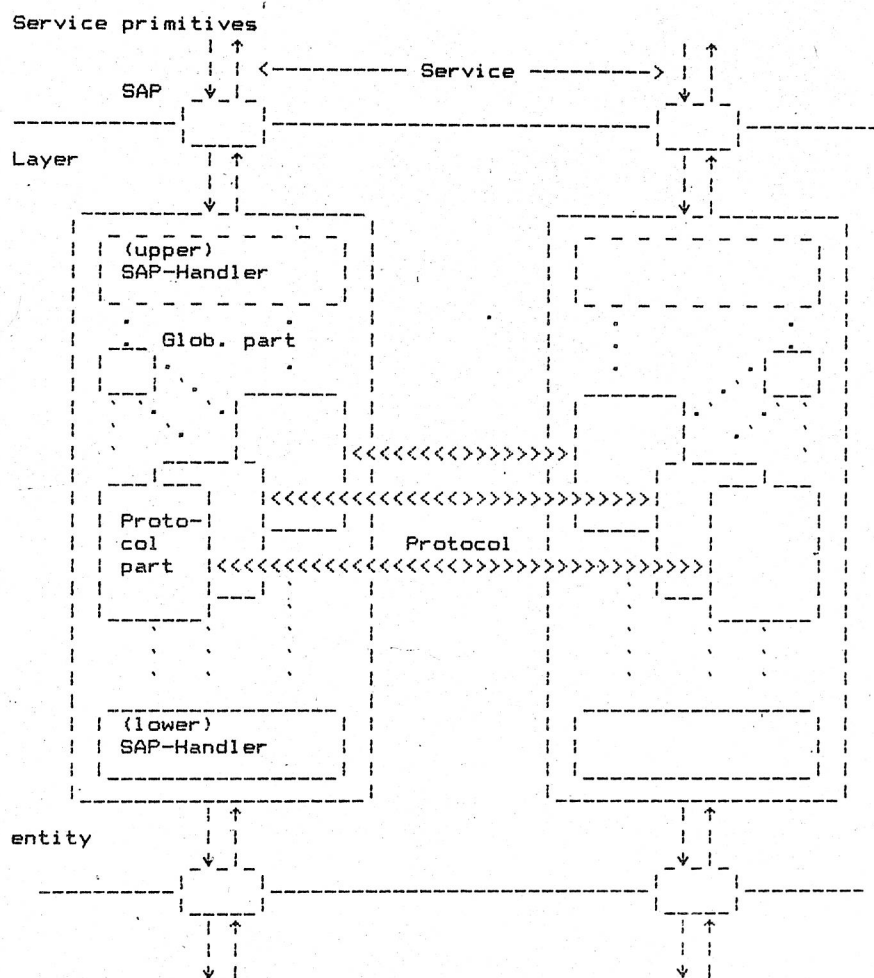
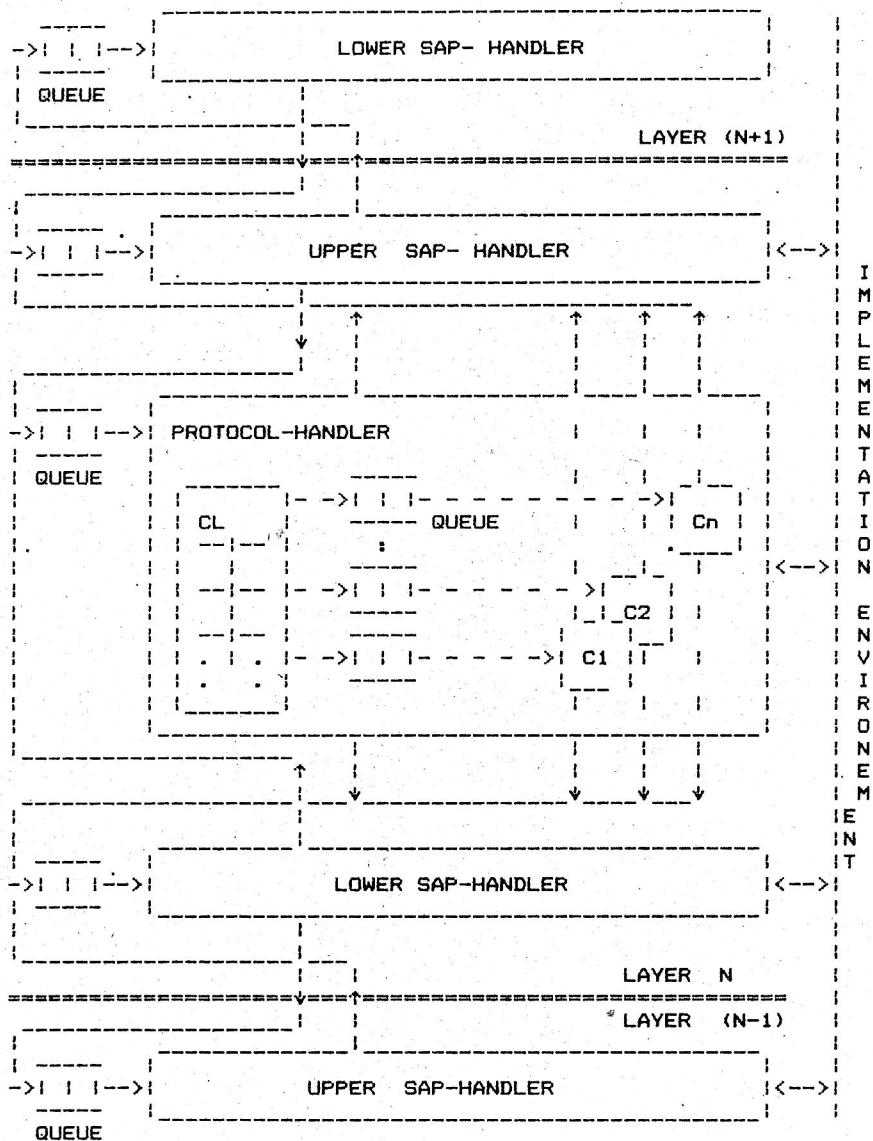


Fig. 2: The PDL-description principle



CL - Connection list

C1..n - Connection 1..n

Fig. 3: Structure of PDL-implementation approach

Piotr KREMIENOWSKI*

sieć komputerowa,
zarządzanie,
określanie problemów

FUNKCJE ZARZĄDZANIA W SIECIACH KOMPUTEROWYCH TYPU SNA

W referacie omówiono funkcje wykorzystywane do celów zarządzania w sieciach komputerowych typu SNA. Podano krótką charakterystykę programów służących do realizacji tych funkcji, ze szczególnym uwzględnieniem problematyki określania problemów. Krótko scharakteryzowano nowy produkt NetView zastępujący wcześniejsze programy służące do zarządzania w sieciach typu SNA.

Podjęcie tego tematu jest celowym z następującego powodu. W ramach Jednolitego Systemu EMC prowadzone są prace nad opracowaniem sieci komputerowych zgodnych z SNA (Systems Network Architecture) firmy IBM [2,3]. Przykładem takiej sieci może być system TELE JS-M opracowywany w IKSAiP we Wrocławiu, który to system jest przedmiotem odrębnego referatu na tej konferencji.

Kategorie funkcji zarządzania w sieci SNA

Sieci komputerowe typu SNA są sieciami rozległymi, obejmującymi lokalny sprzęt komputerów obliczeniowych, procesory sieciowe, linie komunikacyjne, zdalne procesory sieciowe, sterowniki lokalne i zdalne oraz różnorodne terminale. Ponadto w skład sieci SNA mogą wchodzić sieci lokalne. Zarządzanie taką siecią obejmuje wiele różnorodnych funkcji. Można je pogrupować w cztery główne kategorie:

. Zarządzanie problemami

Zarządzanie problemami jest procesem mającym na celu zapewnienie wysokiej dostępności systemu. Obejmuje ono określanie problemów, diagnozowanie problemów, obejście i odzyskiwanie, rozwiązanie problemu oraz śledzenie i nadzór.

* Instytut Komputerowych Systemów Automatyki i Pomiarów, Wrocław.

Określanie problemu jest to wykrywanie utraty (lub zagrożenia utratą) dostępności zasobów systemowych dla użytkowników końcowych. Jest to postępowanie zmierzające do ustalenia jaki zasób, element sieci komputerowej przestał działać poprawnie, stał się przyczyną degradacji pracy sieci. Określanie problemów jest procesem intelektualnym, który może być wspomagany środkami technicznymi i programowymi. Przyczyną powstania problemu mogą być zasoby sprzętowe jak i programowe sieci. Najistotniejszymi programami wspomagającymi zarządzanie problemami, zwłaszcza z naszego krajowego punktu widzenia, są programy określania problemów.

Diagnozowanie problemu jest ustalaniem specyficznych przyczyn powstania problemu i sposobów jego rozwiązania. Do tego celu wykorzystuje się dane sygnalizujące powstanie problemu. Niekiedy potrzebne są dodatkowe informacje dla pełnego diagnozowania problemu.

Obejście i odzyskiwanie jest to zapewnienie środków zastępczych (np. linii) do czasu rozwiązania problemu lub przywrócenie do użytkowania utraconych zasobów programowych.

Rozwiązanie problemu jest akcją korygowania problemu. Po skorygowaniu przyczyny problemu podejmowana jest akcja przywrócenia zasobu do użytkowania.

Śledzenie i nadzór jest to dogłębne oglądanie problemu od jego wykrycia aż do końcowego rozwiązania. Rozwiązywany problem może powodować wiele różnych symptomów i problemów pochodnych.

. Zarządzanie zmianami

W dużych złożonych sieciach zmiany następują stale. Powodowane są zmianami konfiguracji komputera obliczeniowego, sprzętu komunikacyjnego, oprogramowania systemowego a także zmianami aplikacji. Proces ten obejmuje planowanie, dystrybucję, instalowanie oraz śledzenie zmian.

. Zarządzanie konfiguracją

Ta kategoria zarządzania jest związana z utworzeniem i utrzymaniem bazy danych o konfiguracji, która zawiera wszystkie informacje o fizycznych i logicznych zasobach sieci i ich powiązaniach. Funkcje tej kategorii pokrywają się częściowo z funkcjami zarządzania zmianami, lecz informacje w bazie danych dotyczą lokalizacji komponentów sieciowych, ich atrybutów identyfikujących, statusu i procesu zbierania danych o konfiguracji.

. Ocena działania i rozliczanie

Funkcja ta obejmuje procesy pomocne w kwantyfikowaniu, raportowaniu dostępności, monitorowaniu poziomu usług a także rozliczania za wykorzystanie zasobów.

Implementacja funkcji zarządzania w produktach programowych

Pierwsze sieci typu SNA nie posiadały żadnych środków służących do zarządzania. Po uruchomieniu sieci, ich eksploatacja nastroczała wiele trudności, pojawiły się problemy utrudniające lub uniemożliwiające eksploatację. Ażeby wiedzieć co się dzieje w sieci, zaczęto zbierać dane od użytkowników końcówek a także opracowano szereg produktów programowych wspomagających funkcje zarządzania. Omówienie wszystkich wymagałoby zbyt wiele miejsca, zatem ograniczę się do omówienia najważniejszych programów.

. Program operatorskiego sterowania siecią

Umożliwia on:

- sterowanie i nadzorowanie pracy sieci z jednego miejsca,
- automatyzację czynności operatorskich, z możliwością tworzenia tzw. list komend,
- podział sieci na obszary nadzorowane przez wyznaczonych operatorów.

. Program określania problemów sprzętowych

Przeznaczony jest do zbierania danych o tym co się dzieje w zasobach sieci komunikacyjnej danej domeny oraz w zasobach lokalnych komputera obliczeniowego, porządkowania tych danych i udostępniania ich w taki sposób, by wspomagać proces identyfikacji i ustalenia zasobu, który jest przyczyną aktualnego problemu. Użytkownik tego programu musi mieć dostęp do wszystkich zebranych danych, by móc powiązać je logicznie ze sobą i mieć pełny obraz tego co się dzieje w sieci.

Program określania problemów otrzymuje dane w sposób automatyczny, bez zwracania się o nie. Jest to niezwykle istotne. Zatem dane mogą być nadawane tylko przez zasoby "inteligentne", które są w stanie interpretować swoje stany i kwalifikować je jako źródła danych dla programu określania problemów.

Na rysunku 1 pokazane są zasoby, które mogą być źródłem danych dla programu określania problemów sprzętowych.

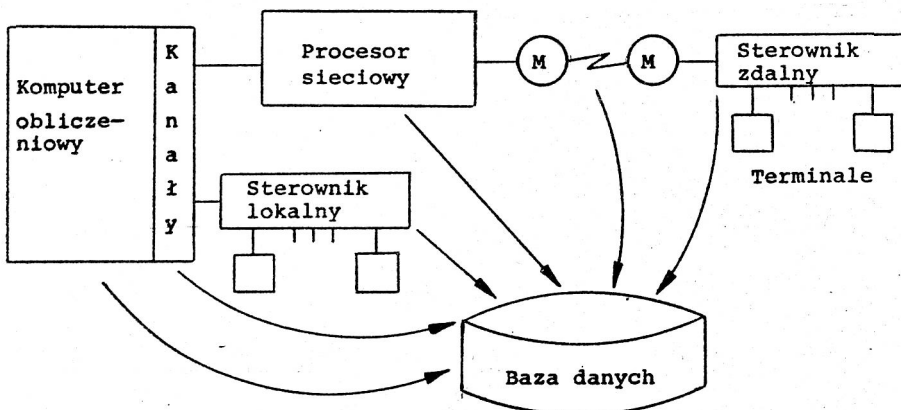
. Program określania problemów logicznych

Spełnia on następujące funkcje:

- . Dostarcza informacji niezbędnych do rozwiązywania problemów nie-sprzętowych.
- . Zbiera i kojarzy informacje o przebiegu sesji.
- . Zbiera i wyświetla informacje o aktualnych czasach odpowiedzi dla poszczególnych końcówek.

Ułatwia identyfikowanie i izolowanie problemów, które nie generują komunikatów o błędach.

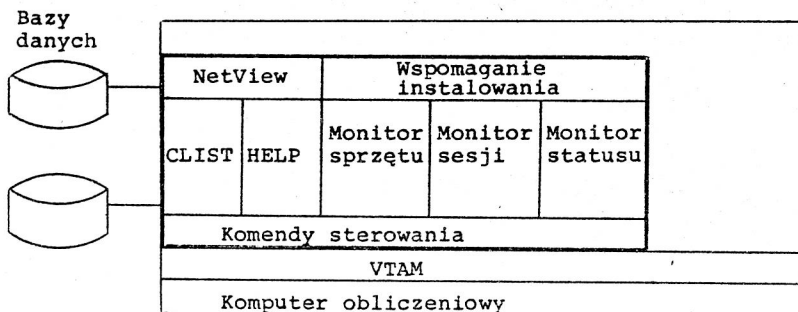
Istnieją inne programy służące do wspomagania funkcji zarządzania, jednakże ich znaczenie jest mniejsze, zatem nie będą tu omawiane.



Rys. 1. Zasoby stanowiące źródła danych
Fig. 1. Resources that are sources of data

Zintegrowane podejście do funkcji zarządzania

W wyniku nacisków na IBM w 1986 roku pojawia się program NetView, łączący w sobie funkcje pięciu wcześniejszych programów. Według dostępnych informacji [1,4,5] jest to realizacja nowej strategii w dziedzinie zarządzania sieciami typu SNA. Jest to produkt mający usprawnić centralne zarządzanie, diagnozowanie i sterowanie sieciami SNA w tym sieciami lokalnymi poprzez wersje NetView/PC. Program ten realizuje wiele funkcji zarządzania. Na rysunku 2 przedstawiono najważniejsze komponenty tworzące ten program.



Rys. 2. Komponenty tworzące NetView
Fig. 2. NetView's components

Istotną informacją jest to, że NetView współpracuje tylko z metodą dostępu VTAM. Poniżej krótko omówione zostaną funkcje poszczególnych komponentów.

- Komendy sterowania – służą do scentralizowanego zarządzania siecią. Istnieje bogaty wybór komend i makrokomend służących do przeglądania baz danych, obsługi plansz, przesyłania komunikatów.
- Monitor sesji – zbiera dane logiczne o przebiegu sesji, kojarzy je i udostępnia do oglądania.
- Monitor sprzętu – zbiera zdarzenia i statystyki oraz generuje alerty.
- Monitor statusu – zbiera i udostępnia na planszy aktualny status wszystkich urządzeń w domenie. Jest to nowa i cenna funkcja umożliwiająca szybkie kontrolowanie statusu zasobów.

W każdej z wymienionych kategorii zarządzania NetView rozszerza spełniając funkcje w stosunku do programów, które on zastępuje.

Zakończenie

Celem referatu było zaprezentowanie punktu widzenia firmy IBM na zagadnienia zarządzania sieciami typu SNA. Firma ta poprzez produkty NetView i NetView/PC nakreśliła strategię swojego rozwoju w tym zakresie. Należy dodać, że w ramach ISO/OSI prowadzone są prace nad standaryzacją zarządzania.

Literatura

- [1] Capps J., Martin R.: Partners – NetView and NetView/PC Linchpins in IBM's network management scheme, Data Communications, March 1987
- [2] Cypser R.J.: Communication Architecture for Distributed Systems, 1978
- [3] Guruge A.: SNA Theory and Practice, 1984
- [4] Kaynuth D.: An integrated network management product, IBM Systems Journal, 1/1988
- [5] Rose D.B., Munn J.E.: SNA network management directions, IBM Systems Journal, 1/1988

MANAGEMENT FUNCTIONS IN THE SNA TYPE COMPUTER NETWORKS

In the paper, the SNA network management functions and some programs that implement that functions are presented. More attention is paid to the problem determination. Next, the NetView program – new product that combines and enhances the functions of previous network management programs is described.

ФУНКЦИИ УПРАВЛЕНИЯ В ВЫЧИСЛИТЕЛЬНЫХ СЕТЯХ ТИПА SNA

В статье описаны функции использованные для управления в вычислительных сетях типа SNA и программы к ним применяемые. Обращается внимание на программу анализа проблем в сети. Кратко описана новая программа NetView, которая выполняет функции управления в сети.

integracja obsługi,
transmisja sygnałów,
sieć komputerowa

Andrzej KUSIUK *

TRANSMISJA SYGNAŁÓW KOMPUTEROWYCH W CYFROWYCH KANAŁACH TELEFONICZNYCH SYSTEMU PCM

W artykule przedstawiono rozwiązanie pozwalające na wykorzystywanie telefonicznych kanałów cyfrowych systemu PCM do równoczesnej transmisji sygnałów telefonicznych i komputerowych. Przedstawione rozwiązanie polega na zastosowaniu hybrydowej komutacji. Dla transmisji sygnałów telefonicznych zachowano powszechnie stosowaną komutację kanałów, natomiast dla transmisji sygnałów komputerowych zaproponowano zastosowanie komutacji wiadomości.

1. Wstęp

Powszechnie stosowanym sposobem wykorzystywania kanałów cyfrowych do transmisji sygnałów telefonicznych i nitelefonicznych, np komputerowych, jest sposób, w którym stosowana jest reguła, według której kanał przesyłowy przydzielany jest na czas trwania transmisji bądź użytkownikowi telefonicznemu, bądź użytkownikowi innego typu. Taki sposób wykorzystywania kanałów nie pozwala na uzyskanie dużego współczynnika ich wykorzystania do transmisji informacji. Uwzględnienie bowiem faktu występowania przerw pomiędzy segmentami mowy (a więc braku transmisji informacji) [8] w czasie trwania rozmowy telefonicznej, w czasie których kanał jest niedostępny dla innych użytkowników, wpływa na znaczne zmniejszenie stopnia jego wykorzystania.

Jednym z rozwiązań umożliwiających zwiększenie stopnia wykorzystania kanału jest zastosowanie komutacji pakietów zarówno do transmisji sygnałów telefonicznych (pakietyzacja mowy) [7] jak i nitelefonicznych. Rozwiązanie to wiąże się jednak ze znaczną komplikacją układową. W powszechnie użytkowanych kanałach cyfrowych stosowana jest komutacja kanałów. Z omawianym rozwiązaniem wiąże się natomiast wprowadzenie komutacji pakietów do transmisji sygnałów zarówno

* Instytut Telekomunikacji, Politechnika Gdańska, Gdańsk

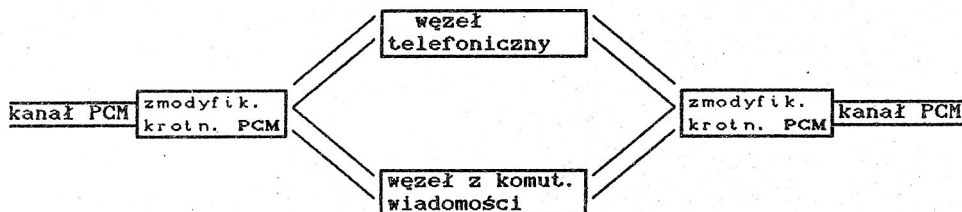
telefonicznych jak i komputerowych a także konieczność stosowania detektorów mowy [6]. Wywiera to wpływ na koszt (indywidualne wyposażenie do każdego kanału telefonicznego) a także na jakość transmisji sygnałów telefonicznych (straty próbek mowy na początku segmentu mowy).

Innym sposobem wykorzystywania cyfrowych kanałów do równoczesnej transmisji sygnałów telefonicznych i niefonnych jest tak zwana komutacja hybrydowa [1]. Sygnały telefoniczne przesyła się wykorzystując komutację kanałów, natomiast sygnały niefonne (komputerowe) przesyła się w oparciu o komutację pakietów. Pewną odmianą komutacji hybrydowej będzie też stosowanie w tej samej sieci komutacji kanałów do transmisji sygnałów telefonicznych i komutacji wiadomości do transmisji sygnałów komputerowych. Przedstawione w pracach [2], [3], [4] rozwiązanie, zwane warunkową komutacją wiadomości (CMSS), pozwala na znaczne zwiększenie stopnia wykorzystania kanałów (do około 0.8) przy stosunkowo niewielkiej komplikacji układowej. Sposób ten sprowadza się do wykorzystywania przerw występujących pomiędzy segmentami mowy, a także pomiędzy rozmowami do transmisji sygnałów niefonnych (transmisja sygnałów niefonnych realizowana jest z niższym priorytetem, a więc do transmisji sygnałów komputerowych wykorzystuje się jedynie nie zajęte kanały telefoniczne).

2. Transmisja sygnałów komputerowych w przerwach pomiędzy segmentami mowy.

W omówionym w pracach [2], [3] i [4] sposobie wykorzystywania kanałów cyfrowych wykorzystuje się własności statystyczne sygnałów telefonicznych przekazywanych kanałem cyfrowym. W szczególności w prosty sposób można wykorzystywać występującą powtarzalność kolejnych próbek telefonicznych w kanale. W tym przypadku transmituje się pierwszą próbkę ciągu identycznych próbek a następnie jedynie informację o braku zmiany próbki. W pracy [3] sposób ten nazwano warunkową komutacją wiadomości ze statyczną alokacją kanałów (CMSS). Z uwagi na fakt, że powtarzalność kolejnych próbek podczas transmisji sygnałów telefonicznych jest bardzo duża [8], opłacalny jest taki sposób transmisji. Uzyskane bowiem wolne szczeliny czasowe można wykorzystać do transmisji innych sygnałów (sygnałów komputerowych). W takim rozwiązaniu kanał cyfrowy wykorzystywany jest do transmisji sygnałów telefonicznych w oparciu o komutację kanałów, natomiast do transmisji sygnałów komputerowych w oparciu o komutację wiadomości. Rozwiązanie to jest więc sposobem integracji obsługi ruchu telefonicznego i komputerowego w kanałach cyfrowych. Odrębne natomiast pozostają same węzły. Tak więc użytkownicy telefoniczni obsługiwani są w węzle telefonicznym, natomiast użytkowników komputerowych obsługuje węzeł sieci z komutacją wiadomości.

Na rys. 1 przedstawiona została struktura przepływu informacji w podstawowym ogniwie takiej sieci, a więc w kanale i w węźle. Elementem wprowadzającym sygnały obu typów do kanału jest krotnica zmodyfikowana w stosunku do powszechnie stosowanej. Rozwiązanie układowe takiej krotnicy PCM przedstawione zostało w pracy [4].



Rys. 1. Przepływ informacji telefonicznej i komputerowej w podstawowym ogniwie sieci.

Z przedstawionego rozwiązania wynika w szczególności, że węzeł telefoniczny wykorzystuje do transmisji rozmów wszystkie możliwe do wykorzystania kanały telefoniczne PCM, a więc liczba kanałów telefonicznych w systemie pomniejszona jest o liczbę kanałów wziętych na stałe do sterowania przepływem informacji w kanale. Dla systemu PCM 30/32 z 30 kanałów telefonicznych do transmisji sygnałów telefonicznych równocześnie może być wykorzystywanych 26. Zmniejszenie o cztery liczby kanałów dostępnych do transmisji rozmów telefonicznych wynika z konieczności przesyłania w każdej ramce informacji czy uległa zmianie próbka mowy (jeden bit na kanał). To zmniejszenie o 13% liczby kanałów telefonicznych daje możliwość transmisji sygnałów komputerowych i to z dużą szybkością. Uzyskana przepustowość kanału do transmisji sygnałów komputerowych jest co prawda zmienia się w czasie, w zależności od liczby prowadzonych jednocześnie rozmów telefonicznych, jednakże wartość tej przepustowości jest bardzo duża (około 1.2 Mbit/s). W rozwiązaniu tym konieczny jest przed przesłaniem odbiór całej wiadomości komputerowej w węźle. Wpływa to na wzrost sumarycznego opóźnienia transmisji wiadomości komputerowej.

3. Zakończenie

Przedstawione rozwiązanie może być bardzo atrakcyjne do zastosowań z uwagi na fakt, że może ono zostać zrealizowane w kanałach cyfrowych systemu PCM, którego rozwój jest zapowiadany. Ponadto rozwiązanie to jest proste w realizacji. Sprowadza się ono do pewnej modyfikacji stosowanej krotnicy telefonicznej systemu PCM. Oddzielenie węzła telefonicznego od węzła sieci z komutacją wiadomości powoduje, że niezależnie od stosowanych typów central telefonicznych (elektromechaniczne, elektroniczne) sygnały komputerowe nie przechodzą

przez centrale telefoniczne (nie występuje zakłócanie sygnału komputerowego przez elementy węzła telefonicznego). Węzeł sieci z komutacją wiadomości jest natomiast węzłem w pełni elektronicznym.

Wadą tego rozwiązania jest oczywiście fakt, że liczba jednocześnie obsługiwanych użytkowników telefonicznych jest mniejsza niż w powszechnie stosowanym systemie PCM (30). Warto tu jednak podkreślić, że zmniejszenie liczby kanałów telefonicznych z 30 na 26 ma miejsce jedynie w przypadku potrzeby wykorzystywania wszystkich kanałów telefonicznych do transmisji sygnałów komputerowych (uzyskany wówczas kanał do transmisji sygnałów komputerowych ma przepustowość ok. 1200 kbit/s). Prezentowany sposób umożliwia wykorzystywanie części kanałów telefonicznych do transmisji sygnałów komputerowych. W szczególności wykorzystując jedynie 8 kanałów do transmisji sygnałów komputerowych, liczba kanałów telefonicznych jest pomniejszona jedynie o jeden. Oczywiście również średnia przepustowość kanału do transmisji sygnałów komputerowych jest odpowiednio mniejsza.

4. Literatura

- [1] K. Kummerle, "Multiplexer Performance for Integrated Line and Packed Switched Traffic", Proceedings ICC'74, Stockholm, Sweden 1974.
- [2] A. Kusiuk, "Analiza pracy węzła sieci telegramowej z komutacją wiadomości", Praca doktorska, Politechnika Gdańska 1988.
- [3] A. Kusiuk, W. Szpankowski, "Efficient Algorithms for Hybrid Data/Voice Transmissions", CSD-TR-805 Purdue University, USA, August 1988.
- [4] A. Kusiuk, M. Zientalski, "Sposób wykorzystania synchronicznych kanałów cyfrowych w systemach wielokrotnych i układ do realizacji tego sposobu", Zgłoszenie Patentowe nr P-266081 z dnia 1987.06.03, Instytut Telekomunikacji Politechniki Gdańskiej.
- [5] A. Kusiuk, M. Zientalski, "Analiza asynchronicznego nadawania w węzle telegramowym z komutacją wiadomości", Materiały Konferencyjne KST'87, ATR Bydgoszcz.
- [6] K. Spiram, P.K. Vershney, J.G. Shanthikumar, "Discrete-time analysis of integrated voice/data multiplexers with and without speech activity detectors," IEEE J. Select. Areas Commun., Vol. SAC-1, No 6, Dec. 1983.
- [7] T. Suda, H. Miyahara, T. Hesegawa, "Performance evaluation of packetized voice systems in simulation study," IEEE Trans. Commun., Vol. COM-32, No 1, Jan. 1984.
- [8] G.J. Weinstein, "Fractional Speech Loss and Talker Activity Model for TASI and for Packed - Switched Speech," IEEE Trans. Commun., Vol. COM-26, No 8, pp 1253-1257, Aug. 1978.

TRANSMISSION COMPUTER SIGNALS IN THE DIGITAL TELEPHONE CHANNELS

In the paper is presented the method utilization of the digital channels PCM to transmission voice and data signals. In this solution the hybrid switching system is used. To voice transmission is proposed circuit switching system but to data signals - message switching system.

sieć lokalna,
model OSI,
usługi sieci

Marek KWIATKOWSKI *

USŁUGI LOKALNEJ SIECI KOMPUTEROWEJ HADES

W referacie przedstawiono usługi lokalnej sieci HADES łączącej komputery ODRA 1305, R-32 i ICL 1904S. Usługi obejmują transfer zbiorów i dostęp do systemu MOP komputerów ODRA/ICL. Architektura sieci wykorzystuje model ISO/OSI. Realizacja usług odbywa się przy wykorzystaniu specjalnego języka makroinstrukcji dostępu do usług warstwy transportowej.

1. UWAGI WSTĘPNE

Lokalna sieć komputerowa HADES służy do szybkiej wymiany danych między wybranymi komputerami znajdującymi się w centrum obliczeniowym COIG.

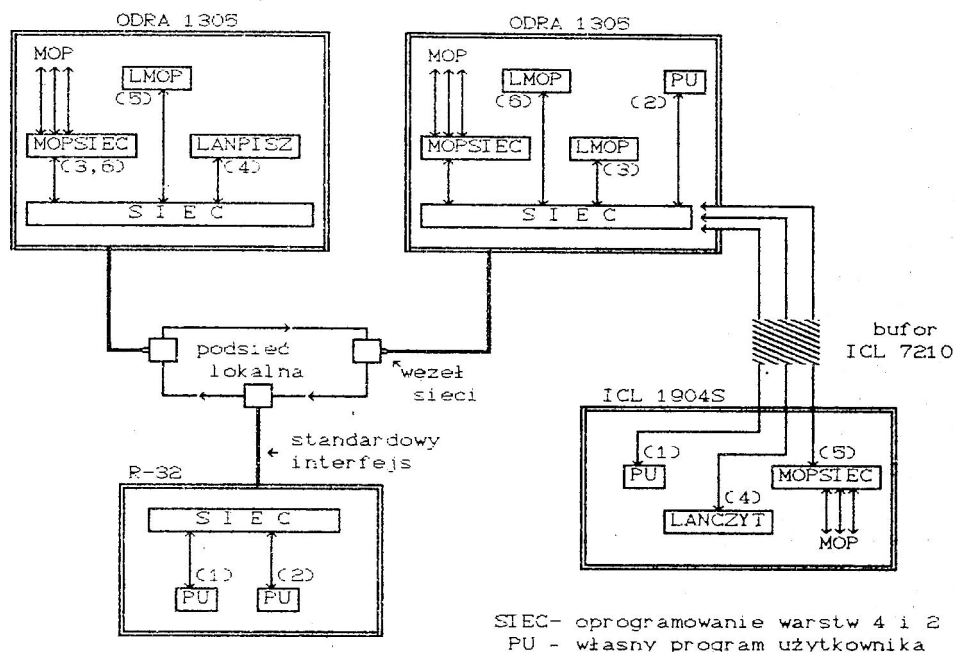
Sieć posiada konfigurację pierścieniową. Dla sterowania transmisją danych wykorzystano metodę znacznika zwanego tokenem (ang. token ring) [2].

Aktualna konfiguracja sieci HADES została przedstawiona na rysunku 1. Do sieci bezpośrednio podłączone są komputery ODRA 1305 i R-32 natomiast pośrednio komputer ICL 1904s.

Architektura logiczna sieci wykorzystuje, z pewnymi modyfikacjami, model systemów otwartych OSI (ang. Open Systems Interconnection [1]) w zakresie dolnych czterech warstw przewidzianych dla lokalnych sieci komputerowych. Rozwiązania implementacyjne poszczególnych warstw zostały przedstawione w [2,6].

W sieci HADES nie jest implementowana warstwa sieci. Funkcje warstw 5-7 modelu OSI są obecnie realizowane w programach użytkowych przy wykorzystaniu niestandardowych protokołów opracowanych w COIG.

*Centralny Ośrodek Informatyki Górnictwa, Katowice



Rys.1. Aktualna konfiguracja i rozmieszczenie standardowego oprogramowania komunikacyjnego w sieci HADES. Numery w nawiasach odpowiadają przykładowym parom komunikujących się programów użytkownika.

Fig.1. HADES network topology and communication software components. Numbers in parentheses correspond to application programs pairs.

2. DOSTĘP DO USŁUG TRANSPORTOWYCH

Dostęp do usług warstwy transportowej realizowany jest za pośrednictwem punktów dostępu do usług transportowych TSAP (ang. Transport Service Access Points). Każdy taki punkt jest jednoznacznie adresowalny w całej sieci lokalnej.

Zakres usług transportowych sieci HADES stanowi podzbiór usług opisanych w dokumencie ISO DIS 8072 [4]. Obejmują one:

- środki dla ustanawiania i kasowania połączenia transportowego między parą punktów TSAP sieci,
- środki dla transparentnej wymiany danych w obu kierunkach między parą punktów TSAP,
- środki, przy użyciu których odbiorca może wpływać na szybkość wysyłania przez nadawcę danych połączeniem transportowym,
- środki dla przyspieszonego transferu krótkich komunikatów danych,
- środki dla bezwarunkowego kasowania połączenia transportowego.

Warstwa transportowa nie zapewnia możliwości negocjowania jakości świadczonych usług oraz ustanawiania więcej niż jednego połączenia transportowego między parą punktów TSAP.

Dostęp do usług transportowych odbywa się w programach użytkowych przez wywołanie standardowych makroinstrukcji (napisanych w językach assemblera komputerów ODRA/ICL i R-32) wraz z odpowiednimi parametrami (np. nazwa lokalnego punktu TSAP, długość komunikatu do wysłania). Makroinstrukcje te przekazują sterowanie do procedur komunikacyjnych dołączanych do programu użytkowego na etapie jego kompilacji. Procedury te wymieniają informacje kontrolne i dane z programem komunikacji z siecią (oznaczonym jako SIEC na rys.1.). Wymiana ta odbywa się za pomocą mechanizmów łączności międzyprogramowej (kanałów WMC i IMC w komputerach ODRA i ICL oraz makroinstrukcji POST w wersji międzystrefowej w komputerze R-32).

Do najważniejszych makroinstrukcji należą :

STINP- żądanie utworzenia lokalnego TSAP i ustanowienia połączenia z TSAP w odległym komputerze,
 STONP- żądanie utworzenia lokalnego TSAP i określenie wstępnej gotowości nawiązania połączenia z dowolnym innym TSAP w sieci,
 STACK- akceptacja , wydanego przez TSAP w odległym komputerze, żądania nawiązania połączenia z lokalnym TSAP,
 STNAK- odrzucenie, wydanego przez TSAP w odległym komputerze, żądania nawiązania połączenia z lokalnym TSAP,
 STPDB- żądanie pobrania komunikatu danych od lokalnego TSAP,
 STWYB- żądanie wysłania komunikatu danych do TSAP w odległym komputerze,
 STEXB- żądanie przyspieszonego wysłania krótkiego komunikatu do TSAP w odległym komputerze.

Program użytkowy posiada możliwość korzystania jednocześnie z kilku punktów TSAP.

Wykorzystując zestaw opisanych makroinstrukcji przygotowano dla użytkowników sieci HADES specjalizowane programy użytkowe realizujące typowe usługi. Obejmują one [3]:

- pracę z terminali dialogowych jednego komputera w systemie MOP pozostałych komputerów sieci,
- transmisję zbiorów między pamięciami zbiorowymi (PZS) komputerów ODRA i ICL.

Użytkownicy sieci mogą tworzyć własne programy użytkowe realizujące przetwarzanie rozproszone.

3. PRACA Z TERMINALI DIALOGOWYCH W WYBRANYM SYSTEMIE MOP

Warunkiem umożliwiającym użytkownikom terminali jednego komputera pracę w systemie MOP innego komputera sieci jest uruchomienie w nim specjalizowanego programu użytkowego oznaczonego jako MOPSIEC na rys.1. Program ten zapewnia równocześnie wielu użytkownikom sieci dynamiczne otwieranie, realizację i zamykanie sesji MOP. Dla realizacji tych funkcji wykorzystano pseudourządzenia typu CI (ang.Command Issuer - wydawca komend).

Użytkownik terminala dialogowego chcąc pracować w systemie MOP w odległym komputerze powinien uruchomić w systemie MOP lokalnego komputera program użytkowy LMOP przez wywołanie makroinstrukcji:

LMOP xx

gdzie xx-adres komputera adresata.

Od tego momentu zadanie użytkownika znajduje się pod kontrolą systemu MOP odległego komputera (dotyczy to wysyłanych komend i przerw).

Chcąc zakończyć sesję MOP w odległym komputerze należy doprowadzić do kontekstu NO USER co powoduje powrót do zadania w lokalnym komputerze.

4. TRANSMISJA ZBIORÓW

Oprogramowanie komunikacyjne sieci umożliwia przesyłanie następujących typów zbiorów między PZS-ami komputerów sieci:

- zbiory podstawowe (typu GRAPHIC, ALLCHAR, NORMAL),
- zbiory amorficzne,
- programy binarne formatu dyskowego.

Przebieg transmisji jest nadzorowany przez parę programów użytkowych uruchamianych makroinstrukcjami LANPISZ i LANCZYT (patrz rys.1.). Ogólna postać wywołania makroinstrukcji jest następująca:

LANCZYT &xx,*nnnn,*nazwa,...,*nazwa

LANPISZ *nnnn,*nazwa,...,*nazwa

gdzie:

- xx - adres komputera, do którego mają być wysłane zbiory,
- nnnn - czteroznakowa nazwa połączenia wybrana przez użytkownika,
- nazwa - nazwa zbioru; dla LANCZYT oznacza zbiór czytany z PZS-u,
dla LANPISZ zbiór zapisywany do PZS-u.

Makroinstrukcje LANCZYT i LANPISZ umożliwiają za pomocą jednego wywołania transmisję do 24 zbiorów różnego typu. Dla danej transmisji nazwa połączenia podana w obu makroinstrukcjach musi być taka sama.

Odczyt i zapis zbiorów z/do PZS-u odbywa się przy wykorzystaniu pseudourządzenia typu File Handler.

5. UWAGI KOŃCOWE

Przewiduje się, że przedstawione w niniejszym referacie specjalizowane programy użytkowe zostaną zaimplementowane w komputerze R-32. Umożliwią one:

- transmisję zbiorów znakowych (wraz z przekodowaniem) między komputerami R-32 i ODRA/ICL,
- pracę z terminali RIAD-32 w wybranym systemie MOP.

Planuje się również podłączenie do sieci HADES komputerów ICL 2957 i IBM-PC oraz budowę kilku sieci lokalnych tego typu i połączenie ich za pośrednictwem zdalnej sieci transmisji danych [5]. Przedstawione w niniejszym referacie usługi użytkowe będą obejmować wszystkie komputery podłączone do tych sieci.

6. LITERATURA

1. Open Systems Interconnection - Basic Reference Model, dokument ISO DIS 7498, 1983.
2. Bulandra M., Kwiatkowski M., Maj W., Lokalna sieć komputerowa HADES, konferencja: "Sieci komputerowe, Usługi, Protokoły, Modele", Wrocław 1985.
3. Instrukcja użytkownika lokalnej sieci komputerowej HADES, wydawnictwo COIG, 1988.
4. Dokument ISO DIS 8072, Transport Service Definition.
5. Kwiatkowski M., Concept of Internetwork Gateway, Zeszyty Naukowe AGH, seria AUTOMATYKA, 1987.
6. Domagała J., Opis funkcjonalny podsieci komunikacyjnej lokalnej sieci komputerowej HADES, konferencja: "Sieci komputerowe, Usługi, Protokoły, Modele", Wrocław, 1985.

USER SERVICES OF HADES LOCAL AREA NETWORK

The paper presents user services of token ring local network which uses ISO/OSI model. Software implementations for file transfer and terminal access to MOP service facilities are briefly described.

УСЛУГИ ЛОКАЛЬНОЙ СЕТИ ЭВМ- HADES

В статье представлено услуги кольцевой локальной сети, применяющей модель открытых систем. Описывается программное обеспечение переноса файлов в сети и реализация сетевого доступа к сервису MOP в машинах ODRA и ICL.

LAN,
interconnection,
router.
bridge

Dr Nigel LINGE*

THE INTERCONNECTION OF LOCAL AREA NETWORKS

With the increasing use of local area networks (LANs) there has developed a need for LAN interconnection to overcome various limitations with current technologies. This paper describes four basic classifications of equipment for network interconnection namely, repeaters, bridges, routers and gateways. Particular attention is paid to bridges due to their enormous commercial potential however, a need for high speed routers is also identified.

INTRODUCTION

With the increasing use of local area networks (LANs)[18] there has developed the requirement for LAN interconnection to overcome limitations in topology, traffic loading capacities and the number of unique end-stations which can be supported. As an example CSMA/CD[13] (IEEE 802.3) is currently leading the field for office networking whilst, Token Bus[14] (IEEE 802.4), under the MAP (Manufacturing Automation Protocol) initiative, is favoured for industrial shop floor networking. Clearly, a means of connecting between these two technologies is required. Additionally, connection to wide area network services is equally important. Companies with offices throughout the world employ wide area networks with LANs providing the networking requirements of individual sites.

When considering network interconnection [4,21,19] there are a number of possible solutions which essentially provide four distinct levels of service. Using IEEE 802 terminology these correspond to repeaters, bridges, routers, and gateways.

A repeater provides the simplest method of interconnection. Operating at the physical layer a repeater is traffic and protocol transparent and merely provides signal amplification and conditioning. As an example the CSMA/CD (IEEE 802.3) LAN permits a maximum of four repeaters between any two communicating end-stations which results in a total end-to-end cable length of 2500 metres.

Bridges operate at the MAC sub-layer of the Data link layer. Although still unable to process protocol information bridges do provide a level of traffic partitioning or filtering in that they are able to analyse individual frame addresses and perform a simple yes/no relaying decision. In principle the IEEE 802 LAN models should allow bridges to connect dissimilar LANs however, in reality incompatibilities between LAN

* University of Salford, Salford, UK

technologies over complicate the situation. Both bridges and repeaters rely on unique MAC layer addressing and also require communicating end-stations to operate identical higher layer protocols.

A router operates at the network layer and represents a more realist method for connecting dissimilar LANs or for providing links to wide area networks. Routers must be addressed by end-stations and are able to provide selective information routeing over mesh connect networks although end-stations must still maintain identical transport and higher layer protocols.

Finally gateways operate at the application layer and prove useful in situations where networks contain end-stations which employ totally dissimilar protocol structures.

This paper describes the operation of gateways, routers and bridges and details their advantages, disadvantages and methods of implementation.

GATEWAYS

Gateways [6,10] represent the most complex method of achieving internetworking in that they act as protocol translators in order to achieve connection between networks which support toally incompatible protocol structures, e.g. interconnection of ISO standard protocols with the IBM proprietary SNA protocol architecture. Such an internetworking task can only be performed by ensuring that the gateway supports the full suite of protocols resident on both networks. In other words all received data must firstly be processed through the entire protocol architecture corresponding to the first network, restructured into a new format and then reprocessed through the other networks' protocol structure before being finally retransmitted. In this way a gateway can be considered as providing internetworking at a level corresponding to the Application layer of the ISO seven layer model and can also contribute a considerable delay towards end to end propagation times.

ROUTERS

In situations where the network protocols contain an element of commonality the design of a gateway can be simplified. To exploit this condition common internet protocols, intended for use by all end-stations within the concatenated networks, have been developed. A gateway, in these circumstances need only implement the common internet protocol to achieve interconnection. The reduced gateway or router [20] therefore provides a protocol specific interconnection operating at the Network layer. Routers must be directly addressed by the network end-stations and are able to provide selective routeing based on various performance criteria such as shortest path or least congestion.

An example of a network layer router which is currently experiencing significant industrial and commercial support and already a recognised de-facto standard is TCP/IP[23]. Originally developed as part of the ARPAnet project it has been subsequently supported by the USA Department of Defense.

The TCP/IP structure which is illustrated in figure 1, comprises three layers namely, the data link layer of the interconnected LANs, an internet protocol (IP) and a transmission control protocol (TCP). The latter two layers correspond to the Network and Transport layers of the OSI model respectively. User application utilities such as file transfer (FTP) and terminal access (Telnet) are provided for above TCP.

The internet protocol provides a connectionless unreliable datagram service in which frames are transmitted without acknowledgement and error control procedures. Addressing is achieved by means of a unique host address and network number. Routers which implement the internet protocol are subsequently responsible for interpreting frame addressing information and with the aid of internal routing tables directing traffic along the most appropriate route.

Additionally, the internet protocol serves to fragment and reassemble a data stream of octets presented from the transmission control protocol above. The TCP protocol is responsible for establishing a connection orientated end to end reliable service. This includes positive acknowledgement and retransmission, sequencing, multiplexing and flow control using window based regulation.

The equivalent connectionless internet protocol is being developed by the ISO[12].

BRIDGES

When the communication protocols used on the interconnected networks are totally compatible internetworking can be achieved by the use of bridges[2,3]. A bridge therefore does not have to provide large quantities of protocol processing and this results in an inherent high speed operational capability. Hence, a bridge represents a form of network connection at the Data-Link layer as shown in figure 2. Operating at such a low level within the ISO seven layered model imposes the restriction that bridge functionality is restricted to an investigation of frame addressing information only. Bridges therefore monitor all LAN traffic and only physically re-transmit that percentage which contains addressing information relating to end-stations known to be on the other side of the bridge. Problems can arise however, if the extended LAN environment contains multiple routes between any given pair of communicating end-stations. Under such conditions bridges are unable to distinguish between different frames travelling between the same pair of end-stations or duplicates of the same frame.

Unless controlled bridges will successively retransmit a single frame along every possible route between the source and destination end-station which causes the destination to receive mis-sequenced and duplicate frames, both of which can cause protocol failure. Additionally such frames circulate forever in the extended LAN environment ultimately leading to network congestion.

Bridges must therefore provide two main functions namely, selective frame relaying and multiple route elimination. Two schemes are currently destined to dominate the commercial market. These are the IEEE 802 Spanning Tree scheme and IBMs Source

Routing.

IEEE 802 SPANNING TREE SCHEME

Currently the IEEE 802 LAN Bridge standard[11] which was developed from an original DEC proposal[9,1] is in an advanced stage of development. The scheme proposed is intended to provide an interconnection mechanism between any combination of IEEE 802 LANs.

With each of these LAN technologies frames contain a 48 bit destination address, which identifies the intended recipient, and a 48 bit source address, which indicates the originator of the frame. Hence, the destination address is used to effect a selective frame relaying decision whilst the source address provides a means of locating the whereabouts of particular end-stations.

The IEEE 802 bridge implements a single internal address list, called the filtering database, within the bridge which contains a list of received source addresses and information regarding the LAN from which each entry was received. The decision to retransmit a frame is determined by comparing a received frames destination address with the contents of this address list. Additionally, a further process is performed to determine whether the frame being tested was received on the same side of the bridge as the entry in the filtering database. If a matching address entry is found, the frame will be discarded if both the entry and the frame were received on the same side of the bridge otherwise, the frame will be retransmitted on the side of the bridge indicated within the address list. A false comparison results in the frame being retransmitted onto every LAN to which the bridge is connected except for the LAN on which the frame was received. The filtering database is constantly updated by adding each received frame's source address and corresponding information regarding the LAN from which they were received irrespective of the retransmission decision. Care is however taken to ensure that the duplicate entries do not occur within the filtering database.

End-station movement is detected by one of two mechanisms either, through the ageing out of entries in the filtering database by timer action or receipt of a frame in which the source end-station appears on a different LAN to that indicated within the filtering database. In the latter case the new information regarding the location of the source end-station is used to overwrite the current entry within the filtering database.

Multiple routes are eliminated by configuring any arbitrary extended LAN into a topology which supports a single active route. This final topology has a pyramid structure with a single bridge, nominated the root and determined purely by its physical address, at the apex. Such a topology, which is termed a spanning tree, is configured through the use of a dedicated bridge protocol which operates transparently to the network end-stations.

Each LAN within the topology also has associated with it a single bridge termed the designated bridge. With respect to protocol operation the designated bridge controls

all other bridges attached to the same LAN. Multiple routes are eliminated by forcing strategically selected bridges to enter a blocking state where they cease frame filtering operations. This feature of the protocol may be used to provide fault tolerance in that blocking bridges may switch back on-line should the topology be forced to undergo a reconfiguration as a direct result of either bridge failure or the addition of extra bridges.

Operation of the protocol is controlled by the exchange of special inter-bridge frames called configuration bridge protocol data unit (BPDU) messages. Each configuration BPDU conveys the transmitting bridge address, the address of the bridge assumed to be root, the distance to the root, the age of the configuration BPDU and a constant which defines the maximum value of an internal bridge timer. Conceptually bridges are considered to comprise a number of interconnected ports. Each port is the physical connection between the bridge and the attached LAN.

When a bridge first comes on line it issues a configuration BPDU onto each LAN to which it is connected in an attempt to become root.

Bridges which receive configuration BPDUs compare the address of the root, indicated within the message, to the currently best known root stored in the bridge. Either the newly received configuration BPDU contains a new root (smaller address in configuration BPDU) or the existing root should be maintained (larger address in configuration BPDU). Possibilities of a tie are disregarded as this would imply a violation of the basic LAN requirements of unique addressing. Where the existing root is maintained the bridge contending to become root is informed of the current topology. It subsequently ceases its attempt to become root. This protocol ensures that topology changes due to the insertion of new bridges are restricted to an area within the control of the nearest designated bridge.

When a new root is detected or a bridge is forced to abandon its contention for root status, currently stored information regarding the root is overwritten with the contents of the newly received configuration BPDU. This new configuration BPDU is then retransmitted with the distance from the root updated by adding one to its value. In so doing a bridge attempts to become the designated bridge on as many LANs as possible. The root is designated on every LAN to which it is connected whilst the designated bridge on all other LANs is the single bridge that is nearest to the root. In circumstances where two bridges interface to the same LAN and are equal distances from the root, the bridge with the smallest physical address always becomes designated. Periodically the root issues configuration BPDUs to confirm its continued existence. These messages are retransmitted by each designated bridge until they have propagated over the entire topology.

Multiple routes are detected by bridges in one of two ways. Either received configuration BPDUs from different LANs proclaim the same root and distance information or a configuration BPDU is received on a port from which a configuration BPDU containing equal root and distance information had been transmitted. Such bridges switch into a backup state and stop relaying normal LAN traffic.

In an attempt to minimise network reconfigurations due to lost configuration BPDUs each designated bridge originates a configuration BPDU if the expected configuration BPDU fails to arrive from the root. This ensures that those bridges 'beneath' the designated bridge, within the spanning tree, do not detect the lost message. If, however, configuration BPDUs from the root are not restored the designated bridge concerned will cease configuration BPDU generation and a full topology reconfiguration will be invoked.

During topology reconfiguration resulting from bridge failure it is possible for end-stations apparently to move from one side of a bridge to the other. If this occurs the contents of the filtering database could be such that an end-station becomes isolated. To overcome this difficulty an additional message transaction is defined to inform the root of a topology change. Reconfiguration BPDUs permeate upwards through the spanning tree with each message being acknowledged by the local designated bridge. The root in turn instructs all other bridges to reduce their filtering database timers for a fixed period before returning them to their original value. This period of reduced timer value is intended to age entries out of the filtering database quickly enough to minimise both the duration and probability of any communication blockages.

In summary, final topology configurations are a function of bridge addresses and the relative location of the bridges and are independent of LAN traffic loading characteristics. In this way the spanning tree scheme exhibits a deterministic behaviour however, the final topology does force all LAN traffic over the same route.

IBM SOURCE ROUTEING

Source routing is a scheme in which bridge complexity can be reduced in favour of forcing the majority of the multiple route elimination functions to be the sole responsibility of communicating end-stations.

Originally described by Sunshine[22], Farber[8] and Kleinrock[16], source routing has now been adopted by IBM[5] as the basis for their token ring[7] local area network bridging scheme. The scheme relies on two factors. Firstly, each interconnected LAN must be assigned a unique identifier, called the segment number, and secondly, it is the responsibility of the source end-station to determine a suitable route to the destination hence, source routing. Each frame transferred between two communicating end-stations contains in addition to the existing 48 bit source and destination addresses, a special routing information (RI) field comprising a sequence of segment numbers which explicitly define the route to be taken through the topology. Bridges achieve the required frame relaying functions by an interpretation of the contents of the frames routing information field only.

Within topologies comprising bridges, communicating end-stations can either reside on the same segment (LAN) or different segments. Clearly, in the former case the routing information is trivial since, existing frame destination and source address

information proves sufficient to establish a communication, i.e. frames travelling between two end-stations in these situations would not be relayed by bridges. However, in the latter case each frame must contain the additional routing information field in order to force intermediate bridges to route frames through the topology along the chosen path. To distinguish between these two types of frame, use has been made of the source group address bit. Current IEEE 802 LAN standards prohibit the use of this bit however, it has been defined in order to maintain compatibility with the destination address format. Existence of the routing information (RI) field is indicated by setting the source group address bit to one whilst, the absence of the RI field is indicated by the bit being cleared.

The RI field, which is inserted into the frame immediately following the source address, has variable length and comprises two portions, the routing control (RC) and a list of segment numbers which identify the LANs on which the frame has or is about to traverse. Interpretation of the routing information is gained via an investigation of the RC field which totals 16 bits of which only seven have significance, i.e. the broadcast (B), length (LTH), and direction (D) bits. The broadcast bit indicates that the frame is an 'all routes broadcast', i.e. bridges should re-transmit the frame onto all LANs to which they are connected. The length bits correspond to the size of the RI field, including the RC, whilst, the direction bit shows the direction in which the frame is travelling, i.e. from source to destination or vice versa.

Operation of the source routing scheme can be explained with reference to the functions performed by the communicating end-stations and those performed by the bridges. Assuming that a source end-station is about to initiate a communication, it can either issue a frame without an RI field, i.e. relying on the destination end-station being on the same segment, or it can issue a frame containing a RI field, known as a resolve frame. For the purposes of describing the details of the scheme, only the latter case will be considered further. Within the RI field the broadcast bit is set, the length field is set to two to account for the RC (the length field is manipulated by bridges only since end-stations are unaware of segment number allocations) and the direction bit adjusted to indicate that the frame is travelling from the source to destination end-station. As the frame propagates through the topology bridges append new segment numbers to the RI field. Hence, eventually the destination end-station receives a copy of the original resolve corresponding to each possible route between the end-stations. To each received resolve, the destination end-station issues a resolve response with a modified RC where the broadcast bit is reset, thus forcing intermediate bridges to relay the frame along the route described by the RI field segment numbers only, and the direction bit adjusted to show that the frame is travelling from destination to source. Alteration of the direction bit permits the segment number list to remain intact but forces bridges to interpret it in the reverse direction, i.e. the last segment on which the resolve was present now corresponds to the first segment onto which the resolve response should appear.

Following these transactions the source end-station will receive multiple resolve response frames (assuming that more than one possible route exists between the source and destination end-stations) from which it must choose one route for use in future communications. This decision is not explicitly defined in that the first response could be chosen to reflect the shortest path in terms of time, or the response with the smallest number of segment numbers could be chosen for the shortest path length. Any remaining resolve response frames however, are stored for future use to overcome failure of the chosen route, i.e. in cases of failure the stored response frames may contain information regarding a route which is still intact and available for use. All frames transmitted by the source and destination end-stations, following the decision on which route to adopt, are non-broadcast, i.e. the broadcast bit is always reset, forcing all such frames to be confined to the single route defined within the RI field. The destination end-station learns of the selected route from the first non-broadcast frame received from the source.

Within the bridge received frames are analysed by firstly establishing whether a RI field is included. Frames which do not contain a RI field are always discarded whilst those which contain a RI field are further processed by examining the RC information to ascertain whether the frame is designated 'all routes broadcast', i.e. the broadcast bit set. In situations where the broadcast bit is set, the bridge scans the RI field. If the RI field currently does not contain any segment numbers, it appends two numbers, one corresponding to the LAN from which the frame was received and the other identifying the LAN onto which the frame is about to be retransmitted. The length field contained in the RC is also incremented. If on the other hand, the RI field already contains segment number information, the bridge scans the list for the number of the segment onto which the frame will be retransmitted. If found the frame is discarded since, it has already traversed that segment, i.e. this action prevents looping. Otherwise, the segment number is added to the list and the length field suitably incremented. As one final test each bridge maintains a value known as the 'Hop count' which determines the maximum number of segment numbers allowed in the RI field. Any frame which exceeds this number is also discarded.

For received frames designated non-broadcast the analysis differs. Firstly, the direction bit is examined to determine how to interpret the segment number list contained in the RI field, i.e. from last entry to first or vice versa. If the bridge is located within the route specified by the RI field, the frame will be retransmitted unaltered otherwise, it will be discarded. If the segment number list contains an ordered pair of segment numbers to which the bridge is attached, corresponding to the direction in which the frame is travelling, the bridge is located within the route. All other possibilities from this test result in the frame being discarded.

Hence in contrast to the IEEE 802 solution source routing configures a single route for each pair of communicating end-stations thereby, allowing traffic load sharing throughout the extended LAN. The major criticism of the scheme within the standards bodies was the requirement to define new LAN frames which violate current

structures defined within the appropriate world standards, i.e. the use of the source address group bit and the inclusion of the routing information field.

THE INTERCONNECTION OF DIS-SIMILAR LAN TECHNOLOGIES

The decision made by the IEEE 802 LAN standards committee to divide the ISO Data-Link layer into technology dependent (MAC) and technology independent (LLC) sub-layers provides the unique situation of enabling bridges to achieve the inter-connection of dissimilar LANs. Under these circumstances a bridge operates at the MAC to LLC interface within the IEEE 802 LAN architectural model. Bridges are however, a new innovation and hence, for historical reasons a number of inconsistencies have emerged from the various IEEE LAN technologies which introduce serious difficulties into the realisation of such bridges.

Each of the three LAN standards produced from the IEEE work define a different maximum limit on the amount of data which can be transmitted in an individual frame, i.e. CSMA/CD allows 1500 bytes, Token Bus permits 8192 bytes, whilst Token Ring can support 2500 bytes. In circumstances where a bridge is used to interconnect say, a Token Bus LAN to a CSMA/CD LAN, the inability to provide frame fragmentation and reassembly at the Data-Link layer prevents a bridge from resolving the situation where a maximum size frame received from the Token Bus has to be retransmitted onto the CSMA/CD LAN.

Further problems arise from the fact that each LAN technology supports its own frame format which forces bridges to provide appropriate 'frame mapping' functions in order to convert between the formats[5]. Additionally, each byte comprising the Token Ring frame is physically reversed in relation to the CSMA/CD and Token Bus frames. Difficulties can still arise when bridging between similar technologies due to the fact that the standards permit a range of possible data rates. Hence, a bridge may be required to connect two LANs operating the same access protocol, say CSMA/CD, but providing significantly different cable data rates, typically 10 Mbit/sec and 1 Mbit/sec. A bridge does not possess sufficient protocol intelligence to provide flow control facilities to transmitting end-stations, i.e. all flow control must have end-to-end significance, leading to potential bridge overload when transferring a large number of frames from the high to low speed LAN. As such routers provide a more realistic solution to these internetworking problems.

SUMMARY

The development of LANs is such that interworking is now essential. This paper has outlined four basic levels at which networks may be interconnected namely, repeaters, bridges, routers and gateways. Currently bridges represent a major commercial area with a number of products in the market place. However, the development of high speed repeaters will almost certainly gain in importance over the next few years.

One of the main reasons for this is the difficulties bridges face when operating in an extended LAN environment in which multiple routes exist between end-stations.

The IEEE 802 bridge provides a solution which forces all traffic over a single route whilst, the IBM source routing scheme provides traffic load sharing at the expense of creating non-standard LAN frames. Currently the University of Salford, UK, and SERC Daresbury Laboratory, UK, are active on a research programme to develop a totally new type of bridge known as the protocol-less bridge[7]. Funded by the SERC this bridge is designed to provide a level of interconnection service which essentially combines the qualities of the two bridge schemes described in this paper.

REFERENCES

- 1) Backes F
Transparent Bridges for Interconnection of IEEE 802 LANs
IEEE Network, Jan 1988, Vol 2, No 1, pp 5-9.
- 2) Ball E., Linge N., Kummer P., and Tasker R.
Local Area Network Bridges
Networks '87, Proc European Computer Communications Conference,
June 1987, pp 203-211.
- 3) Ball E., Linge N., Kummer P S, and Tasker R, (1988)
Local Area Network Bridges
Computer Communications, 11, pp 115-117.
- 4) Benhamou E
Multilevel Internetworking Gateways: Architecture and Applications
Computer, Srpt 1983, pp 27-34.
- 5) Berntsen J A., Davin J R., Pitt D, and Sullivan N G.
MAC Layer Interconnection of IEEE 802 local area networks
Computer Networks and ISDN Systems, Vol 10, 1985, pp 259-273.
- 6) Cerf V, and Kirstein P.
Issues in packet network interconnection
Proc IEEE, Vol 66, Nov 1978, pp 1386-1408.
- 7) Dixon R C, and Pitt D A.
Addressing, Bridging and Source Routing
IEEE Network, Jan 1988, Vol 2, No 1, pp 25-32.
- 8) Farber D J, and Vittal J J.
Extendability Considerations in the Design of the Distributed
Computer System (DCS)
IEEE National Telecommunications Conference, 1973, Vol 1, pp 15E-1 - 15E-6.
- 9) Hawe B, Kirby A and Stewart B.
Transparent Interconnection of Local Area Networks with Bridges
J. of Telecommunications Networks, Vol 3, No.2, Summer 1984, pp 116-130.
- 10) Hinden R., Haverly J., Sheltzer A.
The DARPA Internet: Interconnecting heterogenous computer networks
with gateways.
Computer, Sept. 1983, pp 38-48.
- 11) Institute of Electrical and Electronic Engineers Inc.
Draft IEEE Standard 802.1:Part D (Rev 6), Sept. 1988
IEEE Working paper:- P802.1D/D6 88/48
MAC Bridges.

- 12) International Standards Organisation, ISO 8473 Connectionless network service.
- 13) ISO/IEC IS 8802-3 Carrier Sense Multiple Access with Collision Detection (CSMA/CD) access method and physical layer specification.
- 14) ISO/IEC IS 8802-4 Token-Passing Bus access method and physical layer specification.
- 15) ISO/IEC IS 8802-5 Token-Passing Ring access method and physical layer specification.
- 16) Kleinrock L, and Kamain F.
Hierarchical Routeing for larger networks. Performance evaluation and optimization.
Computer Networks, Vol.1, 1977, pp 155-174.
- 17) Kummer P., Tasker R., Linge N, and Ball E.
A Protocol-less Scheme for Bridging between IEEE 802 Local Area Networks
Computer Networks and ISDN Systems
Vol.12, No.2, 1987, pp 81-87.
- 18) Linge N
Emerging Local Area Network Technologies
Microprocessors and Microsystems, Vol. 9, No.9, Nov. 1985, pp 452-457.
- 19) Perlman R., Harvey A, and Varghese G.
Choosing the Appropriate ISO Layer for LAN Interconnection
IEEE Network, Jan. 1988, Vol. 2, No.1, pp 81-86.
- 20) Seifert W M
Bridges and Routers
IEEE Network, Jan. 1988, Vol.2, No.1, pp 57-64.
- 21) Sunshine C
Interconnection of computer networks
Computer networks, Vol. 1, pp 175-195, Jan. 1977.
- 22) Sunshine C A
Source Routeing in Computer Networks
ACM Computer Communication Review, Vol.7, No.1, Jan. 1977, pp 29-33.
- 23) Transmission Control Protocol, Military Standard, MIL-STD-1778,
U.S. Department of Defense, May 20 1983.

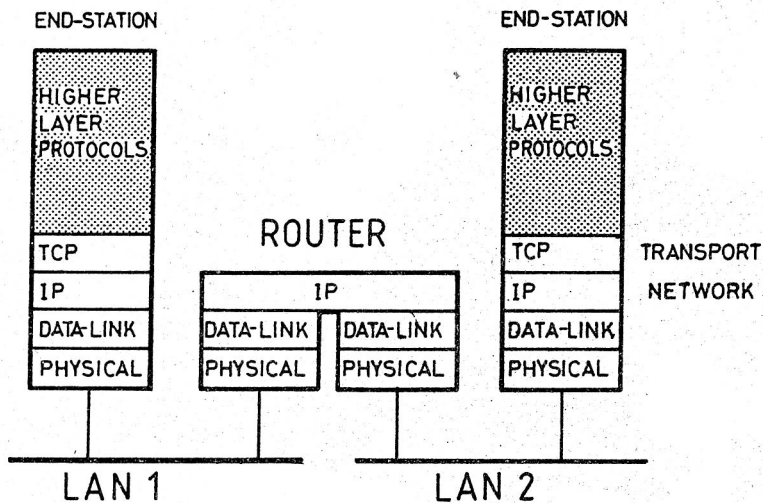


Fig.1 TCP/IP Protocol Structure

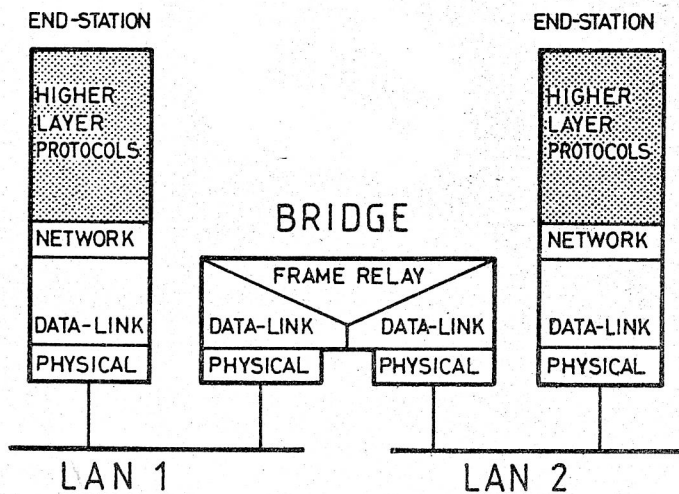


Fig. 2 Bridge Architecture

distributed data processing,
computer network

Y. LYAMIN, V. YALOVETSKY *

A TOOLS FOR DISTRIBUTED DATA PROCESSING FOR COMPUTER NETS

Management of the economical life and everyday functions of a large city with its complex economical relations first of all is based on different city resources: labour power, transport, buildings, communication facilities, etc. Thus, city managing needs precise and off-hand information on the state of city resources to make well grounded decisions.

The task to provide all the municipal bodies with such an information can be easier solved with the help of a Distributed Data Processing System (DDPS), which gives the informational basis for the city computer-aided management systems. This principle is realized in a city management complex "Moscow".

Any DDPS comprises the following subsystems:

- Data Communication System based, firstly, on the net of bus digital channels with the rate of data transmissions up to 64 Kbit/sec, connecting the largest Multiaccess Computer Centers (MCC) of the "Moscow", and, secondly, on the direct line net, providing the remote access to the Net resources with the rate up to 4800 bit/sec. To be exact, the Data Communication System connects 6 large centers and over 2400 user nodes;
- The chain of computer-aided Data Banks (DB) located on the Net nodes with the Multiaccess Computer Centers, which store the information on city resources;
- Instrumental System for DDPS support.

The general principle of our DDPS functioning is based on the principle of practical satisfaction of the first-order demands of final

* 113054, Bahrushina st., 18, NPO ASU "MOSKVA", Moscow, USSR

users in the first turn and provides them with the following means:

- access from any terminal to any DB in the Net; files communication between MCCs;
- programs communication with DBs in a homogeneous net;
- informational services for the end users and application programmers;

An operational environment for DDPS is the system of virtual machines for IBM/370 plug compatible computers. In the given operational system we have developed and used the Distributed Database Management System (DDBMS) called "TRIADA-R", which provides means to connect up to 127 local DBs, located at the abovementioned computers, organized in a net. A user or a program makes a query to DB, calling only the external name of a DB. A query comes to the TRIADA-R's kernel at his/her own computer, that is the computer, connected to the terminal or program, which has made a query. The kernel identifies the query, distinguishes an address of a net node, where the necessary DB is located and, through the means of file communication, sends a query and receives an answer. If it's necessary to transport a DB from one node to another, it can be done without any change in the software. The scheme we have suggested proved to be very useful while working with not so large a number of DBs and users (4-6 DBs at 4 computers), but as the number of users and DBs grows larger, we have to choose more efficient management of the Distributed DBs in the Net and develop a new technology for their design and operation.

That's why in the frame of "Moscow" - complex we are working at the subcomplex of instrumental dialogue systems aimed to support users' communication with the Distributed Information Retrieval Systems /1,2/. The complex of Dialogue Systems maintains all the technological stages of the design and implementation of means for information processing in the nodes of the net. Taking into consideration functional requirements of technological stages, the architecture of the complex must be a set of informationally compatible program systems. The basic level of subcomplex is presented by the DDBMS TRIADA-R. According to a particular informational technology the instrumental tools can be installed in any node in the net. The scheme of tools distribution over nodes follows the logical structure of the informational technology. The independence of all instrumental systems in a set allows to combine arbitrary technological schemes. Moreover, they can be used to construct virtual Personal Information Processing Systems, which can enter the system environment of the DDPS (Distributed Data Processing System) under the control of the Net Operational System.

Design and implementation of application program systems can't do without their main tool - specification language for the information resources control in the Distributed Information Processing System. Specifications denote functional structure of informational technology, data structure, interfaces between data structures and nodes of the net, form of the dialogue. The independent character of language specifications along with the interpretation routines make it possible to have a single correspondence of the project decisions at all the stages of the design and implementation.

The clue concept in our approach is the "independent representation". It means the independence of the application systems model from their realization, DBs development from their running, distribution of program systems over the net nodes from interfaces and access techniques.

Taking into consideration that the model of the application systems is at the same time the model of the problem area, and that the aim of such systems is management of informational resources of the object under control, a DB can be viewed as the object of the problem area. That's the reason to have means of meta-management of data, the role of those means can be played by the dictionary/directories /4/. The dictionary/directory take the function of generalization and management on the metalevel in the DDBMS. Metabase of the dictionaries/directory has the structure, following the logical structure of the DDPS. The elements of such a structure are the nodes of the dictionary/directory.

Integration is the very important characteristic of instrumental systems. It suggests an interface between programs, which provides the unified informational environment. This concept is basic for the design of the complex of instrumental problem-oriented systems (that is shell systems), tunable to particular functions and data structures.

The complex includes the following modules:

1. Technological system of data processing, which has both batch and dialogue techniques for input, controle and correction of documental information of arbitrary structure; besides it has means of developing and text-generation of any output documents.
2. DB quick screen system (DBQSS), which is represented by the dialogue interface with arbitrary DBs without an internal representation. A menu with a password provides entering any node of the DDB.
3. DB EDITOR (DBEDIT). It supports functions of retrieval and editing of arbitrary DBs in a distributed environment. Interface language of the EDITOR is based on the lexics of the XEDIT VM-370 text editor. The DBEDIT permits the controlled backtracking to the initial state of the DB.

4. Dialogue system of the DB developing and running supports the following design functions: DB logical structure design, DB location parameters design, DB loading design, DB modification design.

The abovementioned systems are the instrumental environment of the DDBMS TRIADA-R. The complex of the instrumental dialogue systems is open for the extension by new functional components.

The main architectural concept of the unified net of data communication is represented by the Integrated Service Digital Network (ISDN) /3 /. From the point of view of ISDN utilization in the "Moscow" complex one can distinguish the following perspectives of work:

1. Implementation of local office nets, providing the communication of an office with the integrated service, combining the telephone communication with the computer mail and data processing systems. It can be enforced by the possibilities of internet communications (not necessarily digital net).

2. Implementation of the regional computer net using the TV cable.

3. Implementation of the Unified City Net of data communication with the integrated services.

The perspectives of our work embrace the following aspects: supplying of the ISDN by new technologies, such as broadband batch data communication and opto-electronic communication; realization of the unified mechanism of control over calls for speech communication, data and image communication, based on the channel and batch communication services.

ISDN based informational technology comprises the electronic mail, Teletext and Videotext, Multi-Functional Workstation. The major aim of the discussed complex project is the design of the informational infrastructure of a large city.

REFERENCES

1. V.I.Yalovetsky, Yu.M.Cherkasov. Instrumental Systems for Computer-aided Managerial System Design. Proceedings of the Conference on Governmental and Municipal Information System, Budapest, IFIP TC8, 1987, pp.80-84. Reprinted in Elsevier Science Publishers, Amsterdam, North-Holland, 1988, pp.167-171.
2. Yu.M.Cherkasov, V.A.Grinshtein, Yu.B.Radashevich, V.I.Yalovetsky. Automatization in the Design of the Automated Control Systems with Batch Processing. Moscow: Energoizdat, 1987, 328 p. (in Russian).
3. E.A.Yakubaytis. Network Architecture Integral. - Riga, 1988, 64 pp. (in Russian).
4. B.Leong-Hong, B.Plagman. Data Dictionary-Directory Systems. Administration, Implementation and Usage. Moscow: Finansy i Statistika, 1986, 311 p. (in Russian).

publiczna sieć teleinformatyczna,
SYNCOM, POLPAK

Krzysztof Odolak*
Andrzej Pękalski*

KIERUNKI ROZWOJU TECHNICZNEGO PUBLICZNEJ SIECI TELEINFORMATYCZNEJ W POLSCE DO ROKU 1995

Dotychczas wykorzystywane środki (komutowana sieć telefoniczna i łącza dzierżawione) nie dają perspektyw rozwoju usług teledancyjnych w ciągu najbliższych lat. Jedynym rozwiązaniem otwierającym możliwości rozwoju jest utworzenie, sieci publicznej wyspecjalizowanej do świadczenia usług teleinformatycznych. W tym celu opracowywane są urządzenia sieci SYNKOM. Planuje się import urządzeń do utworzenia sieci POLPAK oraz opracowanie urządzeń do jej rozbudowy.

1. STAN OBECNY.

Stan usług teleinformatycznych w r. 1988 nie oddaje rzeczywistego zainteresowania usługami teledancyjnymi ze względu na istniejące ograniczenia w ich rozwoju.

Stan ilościowy i jakość komutowanej sieci telefonicznej sprawia, że jej wykorzystanie nie jest efektywnym kierunkiem rozwoju tych usług. Komutowana sieć telefoniczna umożliwia transmisję danych z elementową stopą błędów $<10^{-3}$ dla szybkości transmisji do 1200 bitów/s. Czas zestawiania połączenia wynosi do kilkunastu sekund a ze względu na niską dostępność sieci i błędy w zestawianiu połączeń w wielu relacjach znacznie się wydłuża. Dla wielu zastosowań (np. transakcyjnych, konwersacyjnych) są to parametry nie do przyjęcia.

Telefoniczne łącza trwale umożliwiają transmisję danych z elementową stopą błędów rzędu 10^{-5} dla szybkości do 9600 bitów/s. Osiągnięcie wyższych szybkości transmisji wymaga dzierżawienia grup pierwotnych. PPTiT dzierżawi łącza tylko w szczególnie uzasadnionych przypadkach ze względu na poważny niedostatek tych środków dla innych rodzajów usług (zwłaszcza dla telefonii). Mimo, że parametry techniczne łączy dzierżawionych odpowiadają potrzebom teleinformatyki ich wykorzystanie jest ograniczone względami administracyjnymi (brak rezerw) i ekonomicznymi (wysokie opłaty). Nie jest więc to kierunek swobodnego rozwoju teleinformatyki.

*Instytut Łączności, Warszawa.

Dostępne dzisiaj środki teledacyjne (komutowana sieć telefoniczna i łącza dzierżawione) służą do tworzenia tylko łączy podkładowych i nie zapewniają żadnych udogodnień typowych dla teledacji.

W Polsce nie działa wyspecjalizowana sieć telekomunikacyjna oferująca usługi teledacyjne dla teleinformatyki i telematyki a także nie sposób określić kiedy z całą pewnością taka sieć powstanie.

Rozwijane są sieci komputerowe i systemy o ograniczonym zasięgu terytorialnym i zamkniętym kręgu użytkowników (w tym dynamicznie rozwijające się sieci lokalne - LAN) ale tworzenie rozległych systemów informacyjnych i automatyzacji jest niemożliwe bez korzystania z publicznych sieci telekomunikacyjnych. Obecny stan usług teledacyjnych hamuje nowoczesne zastosowania wymagające transmisji danych.

2. PERSPEKTYWA ROZWOJU DO 2010 R.

W prognozie COPIOZ PPTiT oceniono, że możliwości realizacji usług teledacyjnych będą niższe od zapotrzebowania i wyniosą:

- 42,3 tys. stacji transmisji danych średniej i dużej szybkości,
- 16 tys. telematycznych urządzeń końcowych,

łącznie będzie obsługiwanych 58,3 tys. stacji, które w znacznej części korzystać będą z publicznej sieci teleinformatycznej. Pewna część tego zapotrzebowania będzie w dalszym ciągu pokrywana przez komutowaną sieć telefoniczną (co w dużym stopniu zależy od jej jakości) oraz w postaci dzierżawy łączy (co zależy od dostępności tej usługi i taryfy).

Publiczna sieć teleinformatyczna będzie siecią świadczącą pełny asortyment usług i udogodnień teledacyjnych. Działać będą systemy usługowe oferujące rozwinięte usługi komunikacyjne na bazie sieci teledacyjnej. Obok systemów publicznych, tworzonych i eksploatowanych przez PPTiT, rozwój systemów aplikacyjnych użytkowników oraz sieci wydzielonych (abonenckich) może znacząco wpływać na rozwój sieci teleinformatycznej.

Sieci teledacyjne oraz systemy użytkowe będą tworzone w oparciu o standardy międzynarodowe (w szczególności zalecenia CCITT w zakresie telekomunikacji oraz standardy ISO w zakresie systemów informacyjnych). Umożliwi to bezkolizyjną współpracę sieci i systemów krajowych między sobą a także z sieciami i systemami tworzącymi światowy system telekomunikacyjny.

Część usług teleinformatycznych będzie realizowana w sieci z integracją usług (ISDN). Istnieć będą pierwsze fragmenty tej sieci oraz odpowiednia infrastruktura do jej intensywnej rozbudowy - tj. przede wszystkim cyfrowa sieć transmisyjna oraz duże nasycenie usługami telekomunikacyjnymi. Sieć ISDN przejmować będzie obsługę abonentów innych sie-

ci wtórnych a w tym sieci teleinformatycznej.

3. KIERUNKI ROZWOJU PUBLICZNEJ SIECI TELEINFORMATYCZNEJ.

Dotychczas wykorzystywane środki nie dają perspektyw rozwoju usług teledacyjnych w ciągu najbliższych kilkunastu lat. Jedynym rozwiązaniem umożliwiającym pokonanie tych problemów i otwierającym możliwości pełnego rozwoju jest utworzenie, na bazie środków krajowej sieci telekomunikacyjnej nowej sieci, wyspecjalizowanej do szerokiego świadczenia usług teledacyjnych - krajowej sieci teleinformatycznej.

Docelowo w publicznej sieci teleinformatycznej powinny być wykonywane dwie grupy usług:

- oparte na komutacji kanałów teledacyjnych,
- oparte na komutacji pakietów.

Obie te techniki wzajemnie się uzupełniają a mogą być realizowane w ramach jednej sieci (jak to jest planowane w sieci SYNKOM) lub też przez dwie odrębne sieci.

Prace badawczo-rozwojowe w obszarze sieci teleinformatycznych obecnie koncentrują się na opracowaniu urządzeń dla sieci SYNKOM.

Sieć SYNKOM jest pomyślana jako pierwszy etap rozwojowy perspektywicznej, wielofunkcyjnej sieci cyfrowej IDN. W polu komutacyjnym centrali będą komutowane standardowe kanały synchroniczne 64 kbit/s, a podstawowym portem centrali będzie wejście/wyjście traktu liniowego PCM 30/32 (2.048 Mbit/s). Łącza abonenckie będą dołączane do koncentratorów, w których będą niwelowane zróżnicowane szybkości transmisji (600, 1200, 2400, 4800, 9600 bit/s).

W łączach abonenckich zostanie zrealizowana transmisja dwuplexowa w pasmie podstawowym oparta na zasadzie cyfrowej eliminacji echa. Podstawowym stykiem abonenckim będzie styk według zalecenia CCITT X.21. Terminale pakietowe będą korzystać z sieci SYNKOM jako sieci transportowej na zasadach określonych w zaleceniu CCITT X.32.

Do r. 1995 nie należy się spodziewać w większej skali efektów prowadzonych obecnie prac badawczo-rozwojowych. Jeśli usługi teledacyjne mają być w tym okresie udostępnione szerzej niż dotychczas, to jedynym wyjściem jest zakup za granicą kompletu urządzeń umożliwiających stworzenie sieci załatwiającej najpilniejsze potrzeby użytkowników. Ze względu na mniejsze zaawansowanie w kraju prac nad techniką komutacji pakietów oraz spodziewane niższe ceny urządzeń sieci pakietowej, ewentualne zakupy powinny dotyczyć tej klasy sprzętu. Sieć pakietowa (nazwana POLPAK) rozbudowywana byłaby w oparciu o urządzenia opracowane i wyprodukowane w kraju.

Urządzenia sieci POLPAK powinny spełniać wymagania zawarte w zalece-

niach CCITT, szczególnie serii X, V i F. Usługi sieci POLPAK będą realizowane w trybie komutacji pakietów.

Do sieci będą mieć dostęp urządzenia końcowe:

- pakietowe przez łącza bezpośrednie, zgodnie z zaleceniem CCITT X.25 albo przez komutowaną sieć telefoniczną lub sieć teledacyjną z komutacją kanałów, zgodnie z zaleceniem CCITT X.32;
- asynchroniczne przez łącza bezpośrednie lub komutowaną sieć telefoniczną, zgodnie z zaleceniem CCITT X.28.

Sieć będzie oferować połączenia wirtualne i stałe połączenia wirtualne.

Docelowo obydwie sieci (SYNKOM i POLPAK) będą się przekształcać w jedną sieć świadczącą dwa rodzaje usług - opartych na komutacji kanałów oraz komutacji pakietów.

4. UWARUNKOWANIA ROZWOJU PUBLICZNEJ SIECI TELEINFORMATYCZNEJ.

Kierunki rozwoju teleinformatyki zależą od następujących czynników:

- dotychczas nie podjęto strategicznych decyzji stwierdzających potrzebę rozwijania teleinformatyki jako jednej ze służb telekomunikacji oraz określających warunki tego rozwoju;
- potrzeba świadczyć zupełnie nowe usługi a więc trzeba ustalić zakres i formy ich świadczenia, zasady organizacji i ekonomiki oraz sposoby współpracy z istniejącymi służbami w kraju i za granicą;
- brak jest rzetelnego rozeznania co do obecnego i przyszłego zapotrzebowania na te usługi a informacje takie staną się dostępne dopiero gdy zostaną otwarte możliwości normalnego rozwoju;
- w kraju oraz w krajach RWPG nie produkuje się stosownych urządzeń i brak wiążących informacji o planach uruchomienia takiej produkcji;
- urządzenia produkowane w krajach bardziej zaawansowanych technicznie obłożone są ograniczeniami wywozowymi;
- zakup tych urządzeń za granicą wymaga znacznych środków dewizowych (rzędu 2000 \$ za jedno zakończenie łącza);
- na świecie usługi teleinformatyczne i telematyczne są intensywnie rozwijane i w literaturze podawane są różne wnioski dotyczące szczegółowych rozwiązań, przy czym rozbieżności wynikają zarówno z lokalnych warunków jak i bardzo szybkiego rozwoju techniki;
- standaryzacja usług i rozwiązań technicznych nie została zakończona a tendencje do integracji technik i usług zmieniają relacje między różnymi sprawdzonymi praktycznie rozwiązaniami;

5. PRZEWIDYWANE SCENARIUSZE ROZWOJU SIECI TELEINFORMATYCZNEJ.

Szanse najszybszego i najmniej ryzykownego rozwoju usług teleinfor-

matycznych daje wariant, według którego w latach 1989-90 zostanie zrealizowany I etap budowy sieci POLPAK. Pozwoli on na wejście w lata 1991-95 z zaczątkiem publicznej sieci teleinformatycznej. Jego realizacja wymaga jednak szybkich decyzji i sprawnego działania w latach 1989-90. W latach 1991-95 sieć POLPAK będzie rozbudowywana w oparciu o urządzenia opracowane i wyprodukowane w kraju a także powstanie sieć pilotowa SYNKOM.

Prawdopodobny jest wariant, według którego decyzja o zakupie urządzeń z importu będzie odkładana, gdyż warunki zakupu mogą być niesprzyjające a jednocześnie tempo opracowań krajowych nie pozwoli na zaspokojenie rosnących potrzeb w latach 1991-95. Jednak opóźnianie decyzji w sprawie zakupu urządzeń i przeciąganie procesu wdrażania do eksploatacji pierwszej fazy sieci POLPAK może w ogóle podważyć sens budowy tej sieci.

Wariant ostatni, według którego zrezygnuje się z zakupu urządzeń z importu i pozostanie przy programie tworzenia sieci SYNKOM stanowi minimum tego co można obecnie planować. Taka koncepcja jest faktycznie obecnie realizowana gdyż koncepcja sieci POLPAK pojawiła się w obecnej postaci dopiero w połowie 1988 r. Grozi to ryzykiem odsunięcia rozwiązania problemu usług teleinformatycznych oraz telematycznych praktycznie na lata 1995-2000.

* * *

TECHNOLOGY TRENDS IN PUBLIC DATA NETWORK AREA IN POLAND UP TO 1995.

Today's media (switched telephone network and leased lines) have not perspective in expansion of data services. One way to start progress in this area is erection of the public network specialized for data services provision. Equipment of SYNKOM network is developed for this purpose. There is a plan to purchase equipment for POLPAK network introduction and to develop equipment for its extension.

terminal system,

REG-System

Martin SIEBERT*

Klaus NOLTE *

Uwe HOSTMANN *

EMPIRIC PREDICTION MODELS OF THE REG-SYSTEM

Certain empiric models were developed to assess and predict the performance of the REG-system, a terminal system for the bank branches of Berlin. The growth of rush hour workloads was forecasted by use of special models reflecting system extensions and the development of the counter traffic. Linear and nonlinear regressions were used to model the dependence of the CPU loads on workloads and numbers of terminals in service. By use of analyses of variances, factors influencing the response time behaviour (e.g., message lengths, type of order, number of terminals at the line, line speed, CPU load) were investigated.

1. INTRODUCTION

The REG-System is being built up for the bank branches of Berlin and is still under development. Up to now about 350 PC-like terminals (teller machines and automated teller machines) are connected via local concentrators, nonswitched lines and front-end processors to one or two RIAD1056. The teller machine (K8924) is a special version of a PC (BC5120). The hardware of the local concentrator is a KON20. The following different types of lines are used:

- half duplex - 2.4 kbps,
- duplex - 2.4 kbps and
- duplex - 4.8 kbps.

Up to 4 front-end processors of the type TSRA are connected to one RIAD1056.

The RIAD1056 uses OS-SVS 7.1 as operating system. The application program is based on DAKS. BSC is used as data transmission procedure.

*Academy of Sciences of the GDR, Institute of Informatics and Computing Technique Rudower Chaussee 5-6 Berlin, 1199

The local concentrator works without block buffering. Therefore, it works nearly without additional time delay.

Response time behaviour (terminal → CPU → terminal) was adopted as a criterion for the performance of the REG-system. Previous investigations [1] induced that the CPU of RIAD1056 is a bottleneck of the system. Hence, it is essential

- to determine the effects of the CPU load and of other influencing factors to the response times (see 4.) and
- to predict the development of the CPU load.

Two steps were performed to predict CPU loads:

- modelling of the dependence between CPU loads and workloads (see 3.) and
- forecasting of workloads (see 2.).

The following measurement means have been used:

- For every transaction induced by customer orders or management activities a record was written into the so called time measurement file (TMF) by a software monitor installed on the RIAD1056. This record includes
 - the time of registration of the transaction at the CPU,
 - type of order (disbursement, print of abstract of account (PAA), ...),
 - terminal identifier,
 - message lengths (of request and answer),
 - identifier of the data file of account and
 - response time for the previous transaction at the same terminal (precision: whole seconds; not for every transaction).
- Moreover, a hardware monitor measures the CPU time consumption for every minute (accuracy of 0.01%).

2. WORKLOAD PREDICTION

2.1. A detailed model

Models were created for rush hours only. A rough model description has the following form:

$$PL_{RH} = C MR_{RH} + E E D G V_{E RT} + C M M_{RH} + E M D G M V_{E M} \quad (1) \quad \text{with}$$

PL_{RH} = vector; components are the numbers of orders of certain types for the rush hour in the prediction period,

MR_{RH} = quantity of measured real-time orders in the rush hour,

EE = estimated quantity of real-time orders (RTO), which are to be processed additionally because of system extensions,

G_E = estimated growth rate for the prediction period (concerning real-time orders),

V_{RT} = vector; components are the measured relative proportions of different real-time order types (sum of components = 1),

MM_{RH} = quantity of measured management orders (transactions) in the rush hour,

EM = estimated number of management orders, which are to be processed additionally because of system extensions,

GM_E = estimated growth rate for the prediction period (concerning management orders),

V_M = vector; components are the measured relative proportions of different management orders (sum of components = 1).

The unknown coefficients of (1) are to be estimated by use of

- statistical data collected in the last years and
- TMF data of a certain observation day.

Due to the aim of prediction and to the date of observation, the components of PL_{RH} and V_M , the length of the prediction period and the way of estimation of the unknown coefficients in (1) have to be specified.

E.g., predictions were computed to forecast

- what happens when PAA's are available in all branches (at measurement time PAA's were not available in all branches),
- the effect of integration of savings book orders,
- the development of the workload at all,
- the development of workloads distinguished for several types of instructions.

A single prediction vector PL_{RH} strongly depends on the data of one observation day. Using the data of several observation days, several predictions PL_{RH} were computed for the same magnitude(s) of interest. To assess the variability of these predictions, the standard deviations, resp. the coefficients of variation, were computed by use of the several predicted values for the same magnitude of interest. For special examples coefficients of variation ranged about 10..15%.

Furthermore, predicted values were compared with observed values for special cases. This induced two essential conclusions:

- There are unconsidered influencing factors. Presumable, there are seasonal influences with essential effects. Unfortunately, the amount of data was too small for investigating such factors.
- Nevertheless, it has been shown that the prediction model is usable for long-term forecasting with acceptable results.

2.2. A regression model

The model discussed up to now lacks a quick response to short-term measurements (as necessary in discussions with partners) because of laborious calculation. Therefore, a simple regression model was developed. It has the following form:

$$RTO_i = a + b i + u_i \quad (2) \quad \text{with}$$

RTO_i = number of measured real-time orders in the rush hour of week i ,

i = number of weeks after January 6th 1987,

u_i = error terms and

a, b = regression coefficients.

On one hand, the factor "time" turned out as particularly qualified for such a model, because the system extensions depend on the time, too. On the other hand, it is probable that in consequence of changing the extension strategy the regression model is worthless for forecasting. *Besides strictly speaking, it is not permitted to use such models outside the observation time.* Nevertheless, we developed such a model with the aim to use it for short-term predictions.

Results were quite satisfactory. Squared multiple correlation coefficients were very high (greater than 0.95).

3. ANALYSIS OF THE CPU LOAD

3.1. Linear models

Linear regression models reflect the CPU load behaviour in a satisfactory manner for low and medium CPU loads (cf. [2]). Because of rising CPU loads and certain system changes during 1987 it became necessary to prove the applicability of linear models and to revise coefficient estimates by using data from Oct.-Dec. 1987 and from 1988. Additionally, improved models were developed considering the number of terminals in service as a magnitude of influence. Finally, there was used the following linear model

$$Y_j = a_0 + \sum_{i=1}^5 a_i X_{ij} + e_j, \quad j=1, \dots, K, \quad (3)$$

where K is the number of 3-minutes-intervals, Y_j - the CPU load (in seconds), X_{ij} , $i=1, \dots, 4$, - the number of transactions of kind i , X_{i5} - the (approximate) number of terminals in service for the j .th 3-minutes-interval, a_i - coefficients, e_j - random error terms.

There are four kinds of events:

- 1 special system operations,
- 2 real-time disbursements,
- 3 real-time payments,
- 4 other direct access operations.

By use of ordinary least squares estimation (OLSE) coefficient estimates were computed based on data from Oct. 6th, Dec. 8th and 22nd in 1987 and May 24th in 1988. Comparisons of the results from May 24th with the other results reflected improvements of the REG-system in the beginning of 1988. In the following table estimates are given for the data from

1987:	6.10.	8.12.	22.12.			
K	171	171	141			
mean CPU load	67.4%	75.2%	80.2%			
squ. multiple						
corr. coeff.	0.987	0.960	0.961			
SE	2.56	3.25	1.77			
i	a_i	SEa_i	a_i	SEa_i	a_i	SEa_i
0	21.46	1.20	42.90	2.05	50.30	1.99
1	0.207	0.028	0.151	0.038	0.143	0.023
2	0.120	0.011	0.103	0.016	0.072	0.0074
3	0.338	0.055	0.270	0.74	0.090	0.041
4	0.209	0.0091	0.177	0.0091	0.157	0.0058
5	0.136	0.0046	0.109	0.0074	0.081	0.0072

(SE-standard error).

Cross validations gave acceptable results. Nevertheless, detailed examinations of the results from linear regressions gave rise to use a certain nonlinear model.

3.2. A nonlinear model

There are certain reasons to assume that the data communications system works more effectively for CPU loads between 80 and 90% than for lower CPU loads (between 60 and 70%), i.e., the ratio between the number of transactions in a time interval and the corresponding CPU load is greater for the higher CPU loads. It is given the following nonlinear model:

$$Y_j = \exp(k_0 L_j) a_0 + \exp(k_1 L_j) \sum_{i=1}^5 a_i X_{ij} + e_j \quad j=1 \dots K. \quad (4)$$

where L_j is the sum of all events in the j .th time-interval and k_i , $i=0,1$, are growth coefficients.

First, there were computed common estimations for k_0 and k_1 for the three days from 1987 by minimization of the residual sum of squares using a gradient method. There were found

$$k_0 = 0.000808 \quad k_1 = 0.000471.$$

Using these values, linear coefficients were estimated by OLSE

separately for the different days. There were found

	6.10.1987	8.12.1987	22.12.1987
a ₀	22.608	21.479	22.243
a ₁	.135	.415	.305
a ₂	.189	.167	.108
a ₃	.239	.287	.183
a ₄	.258	.263	.280
a ₅	.127	.134	.159
SE	2.812	2.916	1.887

Let us consider the following workloads for a 3-minutes-interval:

kind	1	2	3	4	5
workload	57	284	39	426	270

(prognosis for Nov. 15th 1988).

Using these values, the following predictions for the CPU load were computed (see (4)):

	linear model	nonlinear model
6.10.	114.6	101.7
8.12.	109.4	105.6
22.12.	100.8	102.2

Thus, the nonlinear model gives approximately the same results for the different days, whereas predictions by using the linear model are rather different. (Use of data from May 24th 1988 gave lower predictions.)

4. RESPONSE TIME BEHAVIOUR OF REAL-TIME TRANSACTIONS

4.1. Empirical statistics

Investigations were carried out using data from December 1987, April and May 1988. Mean CPU loads for the rush hours ranged between 80 and 85%. There were computed empirical statistics and analyses of variances (ANOVA). Besides the determination of the magnitudes of main influence and quantitative assessments of these influences, it is of interest to investigate when mean response time values exceed the 3-seconds-limit and when (empirical) 95%-percentiles exceed the 5-seconds-limit. Empirical statistics (and corresponding 95%-confidence bounds) were computed for a large number of subgroups of the data. Especially for the following situations critical response time behaviour, i.e., mean values significantly exceed the 3-seconds-bound, was observed:

PAA's

- for PAA's at all (partially, the 4-seconds-limit was exceeded; only for 4.8-kbps-lines the 3-seconds-limit was kept approximately also for higher TADLV-values (numbers of active terminals at the line; max.

TADLV-value was 700,
disbursements

- for 2.4-kbps-lines and high TADLV-values (TADLV=6,7).

Differences between mean response times when distinguishing between CPU loads between 70 and 75% and CPU loads between 85 and 90% in most cases did not exceed 0.5 seconds. For CPU loads greater than 90% numbers of observations were too small for getting reliable conclusions.

Observed standard deviations in most cases ranged between 0.8 and 1.4 seconds. Altogether, higher mean values of response times were connected with higher standard deviations.

95%-percentiles were estimated by means of linear interpolation. Estimated 95%-percentiles did not exceed the 5-seconds-limit when the corresponding mean values were less than 3 seconds.

4.2. Analyses of variances

By the following reasons the use of ordinary least squares estimators (OLSE) is problematic:

- distributions of the error terms are non-normal,
- error variances are not homogeneous (higher standard deviations for higher mean values).

Nevertheless, in most cases OLSE was used. Several trials with weighted least squares estimation in order to compensate the effects of heterogeneous error variances gave hints that the reliability of coefficient estimators is quite satisfactory, whereas F-statistics, standard error estimators and confidence bounds should be regarded cautiously.

Standard error estimators were relatively high. The main reason for this can be seen in the relatively low accuracy of the observed response time values (accuracy of seconds).

When selecting appropriate models, the aim was to find several ones reflecting in each case special points of view rather than to find one unique "all-including" model.

Main results were the following:

- about 2.5..3.5 ms delay per byte message CPU -> terminal and about 3..7 ms delay per byte message terminal -> CPU;
- 750..900 ms delay for PAA's compared with other real-time transactions using access to a data file of accounts;
- 400..600 ms delay due to access to a data file of accounts;
- 150..300 ms delay per additional terminal at the same line;
- about 450..900 ms delay by use of a 2.4-kbps- instead of a

4.8-kbps-line;

- about 100 up to several hundreds ms delay per transaction processed at the same line during the preceeding 2.5 s;
- several up to several dozens of ms delay per real-time transaction processed at the CPU during the preceeding 2.5 s;
- 170..300 ms higher response times for CPU loads between 85 and 90% in comparison with CPU loads between 70 and 75%;
- about 40..250 ms delay due to the difference between half duplex and duplex lines.

The smallness of the observed deterioration of the response time behaviour which corresponds to CPU loads rising from 70% up to 90% was an unexpected result.

Special interest was dedicated to several interaction effects. Altogether, certain main effects and covariates turned out as being the most important components of appropriate ANOVA models, whereas all regarded interaction effects were assessed as being less important. Nevertheless, there were found out and interpreted certain influences due to interaction effects.

5. REFERENCES

- [1] D.Carl, M.Siebert, H.Siegert, C.Tessmer
Research paper "Analyse von Architekturvarianten fuer das Rechnernetz zur Rationalisierung geldwirtschaftlicher Prozesse in Berlin unter Nutzung eines Datenkommunikationssystems"
Acad. of Sc., IIR-RK/NA, Berlin 1984
- [2] B.Gnuechwitz, U.Hostmann, M.Siebert, H.Siegert, I.Wenhold
Research paper "Untersuchung der transaktionsorientierten Anwendung REG",
Acad. of Sc., IIR-RK/NA, Berlin 1987
- [3] U.Hostmann, K. Nolte, M. Siebert
Empiric Study of the REG-System,
Proc. of the 2nd International Seminar on Modelling and Performance Evaluation of Distributed Computer Systems,
Wendisch-Rietz, Nov. 1988
- [4] U.Hostmann
Internal paper "Antwortzeitverhalten im REG-System",
IIR-RK/NA, Berlin 1988
- [5] K.Nolte
Internal paper "Modell der CPU-Belastung",
IIR-RK/NA, Berlin 1988
- [6] M.Siebert
Internal paper "Arbeitslastmodell fuer die zentrale Verarbeitungsebene des REG-Systems",
IIR-RK/NA, Berlin 1988

multidatabase management

Witold STANISZKIS*

AN OVERVIEW OF THE MULTIDATABASE MANAGEMENT TECHNOLOGY

ABSTRACT

The highlights of the multidatabase management technology are presented in terms of principal user requirements and system characteristics. The most significant systems, both research prototypes and commercial products, are presented. Future trends and outstanding problems in the area of the heterogeneous database integration are briefly discussed.

1. INTRODUCTION

The growing maturity of database management system and techniques has led to a substantial growth in investment in corporate data resources. The current potential of information processing systems and the sophistication of user requirements have created a strong need for integration of data resources at the corporate as well as government administration levels. The growing backlog of data processing applications precludes the possibility of reimplementing existing systems, in order to create new centralized databases supporting the required integration of data.

Typically, a large organization has a number of different centralized database management systems and many geographically distributed databases. In some cases, computer networks based on heterogeneous hardware are used within the same organization. Hence, integration of data resources requires distributed data management capabilities that are sufficiently powerful to cope with the above situation.

Intensive research and development work in the area of multidatabase systems, including those to be implemented in the heterogeneous hardware environment, has been stimulated by the growing user demand.

2. MULTIDATABASE MANAGEMENT SYSTEM ARCHITECTURE

As the result of the above user requirements the multidatabase systems have become increasingly popular. The main interest has been directed towards integrating heterogeneous databases, but systems integrating homogeneous data models were also developed. One of the most serious outstanding problems is transaction processing. This is due to the lack of compatibility of the concurrency control mechanisms of the different database management systems as well as the use of data manipulation or query languages (user interface levels) as access mechanisms by the multidatabase management systems.

A multidatabase management system may be defined in terms of the following characteristics:

Integration of heterogeneous databases - this implies that the underlying databases, accessible through the system, are managed by different DBMSes and may represent diverse logical data models.

Distribution - the fact that data are not resident at the same site (processor) and that all sites are linked via a geographical or local network.

Logical correlation - the fact that data have some properties that tie them together, so that they may be represented by a meaningful, common data model.

Distribution may not always be a necessary condition to justify an application of a multidatabase management system. Clearly, there exist cases, where different centralized database management systems are present at a single site, and a multidatabase system may be used to provide for the required integration of data. However, the lack of distribution capabilities would be a serious limitation with respect to most applications.

Logical correlation implies existence of a common logical data model with its corresponding data language. Since many different data models already exist, corresponding to the various centralized databases accessible in the realm of a multidatabase system, the necessary mappings between those data models and the common data model of a multidatabase management system must be provided. This is the major distinguishing factor between the multidatabase and the distributed database management systems. In the case of the latter class, a common logical data model is applied and the data are physically distributed.

Presently, there exists little user experience in the area of the multidatabase systems, and the existing applications may be considered, at best, to be experimental prototypes. However, the limited experience suggests a number of principal user requirements.

* CRAI, Rende(CS), Italy

The following requirements have already been reported as desirable in [STA83,DAY82,DAY83]:

1. The data model that supports a non-procedural, end-user friendly data language.
2. The complete distribution transparency providing for the manipulation of data, as if the user were accessing a unique centralized database.
3. The local data model transparency relieving the user of the complexity of dealing at the same time with many, diverse logical data models.
4. The explicit support for definition of data abstractions providing means for correct representation of the multidatabase system semantics.
5. Minimal interference with the local processing of the centralized databases included in a multidatabase system.

Current objectives of multidatabase systems, exemplified by the above user requirements, show that interrogation functions take precedence over the distributed update capabilities. In many cases, allowing the multidatabase system users to update the underlying centralized databases would be considered too much of an interference in the local applications. However, transaction processing based update capabilities may be useful, and they are comprised in some of the current multidatabase management system prototypes. This creates serious technical problems, particularly in the area of concurrency control. We shall later discuss problems associated with multidatabase transaction processing in more detail.

Most of the currently implemented systems have adopted the relational data model as the global data model. In some cases, mappings from network and hierarchical data models are supported. Extensions of the relational data structure definition facilities have also been proposed to provide explicit support for data abstraction.

Since multidatabase systems utilize already existing databases, the possible interference of the global and local functions may be considered from two perspectives; the need to modify existing database design to support the multidatabase system needs, and degradation of the local application performance resulting from additional workload generated by a multidatabase system.

Modification of existing database systems to meet the specific requirements of the multidatabase systems users would seldom be permitted. This fact has determining influence on the multidatabase system design methodology.

The need to minimize the multidatabase system workload causes the processing cost, rather than response time, to be the principal goal of query optimization. Apart from the query optimization strategy, this fact has an important influence on implementation of the multidatabase management system user interfaces.

Typically, a multidatabase management system would support the following data abstraction architecture:

The Global Schema

The Global Schema constitutes the principal integration level. It comprises all logical data structure objects resulting from data abstraction and integration processes performed on the data objects defined in the local database schemata. This level provides an integrated view of the underlying databases, supporting, in most cases, distribution and data model transparency.

The User Schema

The User Schemata correspond to the concept of the External Schema of the ANSI/SPARC DBMS architecture. They comprise all of the user view and snapshot definitions.

The Local Schema

The Local Schemata, defined in the host DBMS data description language, represent the logical data structure of the underlying, local databases to be integrated into a multidatabase system.

The Auxiliary Schema

The Auxiliary Schema is stored, in some cases fragmented, in all sites of a computer network. It comprises data model mapping and other information related to the local resources, as well as information used to resolve incompatibilities of the local databases and the respective host DBMSes.

In the majority of multidatabase systems the Global Schema is replicated in all sites of the computer network. This fact and the need to access data residing in distinct, geographically distributed databases creates new problems in the area of database administration and design.

3. SELECTED MULTIDATABASE MANAGEMENT SYSTEMS

The state-of-the-art survey of the distributed database management technology has been given in [CER84, MOH84, STA85]. Both types of systems, namely the distributed database and the multidatabase systems, have been presented. We shall briefly summarize information regarding the most important developments in the area of multidatabase management systems. A representative sample of the current research and development work includes such systems as ADDS [BRE84, BRE86], MERMAID [BRI84, TEM83, TEM86a], MULTIBASE [SMI81, LAN82], NDMS [STA84a], SIRIUS-DELTA [LIT82], DQS [DQS88], and AIDA [TEM86b].

ADDS (A Distributed Database System)

ADDS has been developed at the Amoco Production Company, USA with the prime objective to link oil exploration databases installed in the computer network of IBM mainframes. The global schema is relational and the system integrates INQUIRE, IMS/VS, and SQL/DS databases. The global query facility provides for distribution and local data model transparency. Global queries are mapped into a collection of local queries, expressed in the host DBMS query language, and the local query functional incompatibilities are resolved by query postprocessing. Query optimization criteria are the communication cost and the local database processing cost. Distributed transaction processing is not supported in the system.

MERMAID

MERMAID is a tested system developed at the System Development Corporation, USA to integrate relational databases in a local network of VAX and SUN Workstation computers. The relational databases currently accessible through the system are INGRES, ORACLE, and the Britten-Litton IDM database machine. The global schema is relational and it also contains the auxiliary information. A single query processor accepts a number of relational data languages, namely SQL, QUEL and ARIEL. Query processing is based on an intermediate language DIL (Distributed Intermediate Language) designed to support distributed query processing. Query optimization is based on the communication cost and the query optimization algorithm is sensitive to data fragmentation and replication. Distributed transaction processing is not supported.

MULTIBASE

MULTIBASE has been developed at the Computer Corporation of America to support retrieval of data from heterogeneous, distributed databases. The global schema is based on the functional data model and DAPLEX is supported as the data language. A global query is decomposed into a set of local queries that are executed by the host DBMSes. Query postprocessing is used to resolve the incompatibilities of the local query languages. Fragmentation is not managed directly, however it is possible to use derivation mechanisms to generate global data from local data objects. Thus, when several local databases autonomously store local data objects, which can be regarded as portions of the same global data object, the global schema includes that global data object and does not make any reference to its components. In this sense the fragmentation transparency is supported. Query optimization is divided into two distinct steps, namely the global and local optimization phases. The global optimization algorithm minimizes the communication cost and attempts to utilize the potential of parallel processing. The local optimization algorithm minimizes the response time of the host DBMS data retrieval operations. Distributed transaction processing is not supported.

NDMS (Network Data Management System)

NDMS has been developed at CRAI, Italy to provide a facility for retrieval and distributed processing of data residing in distinct, preexisting databases and data files installed in a heterogeneous computer network. The current version of the system operates on a network of IBM and VAX computers integrating local databases supported by ADABAS, IMS/VS, IDMS DB/DC, RODAN and INGRES database management systems. The global schema, replicated in all sites of the computer network, is relational with SEQUEL supported as the data language. User schemata are supported by the view definition mechanism of SEQUEL and, both permanent as well as temporary, snapshots may be defined. The auxiliary schema comprises mapping definitions controlling materialization of global schema relations. Query processing is based on an intermediate data manipulation language designed to exploit the inherent parallel execution capabilities. A centralized query planner generates a query plan, defined as a sequence of the intermediate language operations, resulting from the query optimization algorithm. The goal function of the query optimization algorithm is minimization of the overall query processing cost, including the communication cost as well as the CPU and I/O operation costs. Query postprocessing is not necessary, since all local database accesses are performed via the host DBMS data access modules developed in the respective data manipulation languages. Distributed transaction processing is supported and the concurrency control approach is based on the two-phase commit protocol and the corresponding host DBMS facilities.

SIRIUS-DELTA

A part of the SIRIUS project, initiated in 1977 at INRIA, France, has been devoted to development and experimentation with multidatabase system architectures. Approach to the multidatabase system architecture, developed within the project, is characterized by a different treatment of the database integration problem. It is believed that the distribution transparency is not necessary in the multidatabase system and that the localization of data embodies important semantic

databases. Dependencies include semantic dependencies, integrity constraints and privacy dependencies. A collection of relational databases is considered to be a relational multidatabase. The multidatabase update operations are also supported by MALPHA.

DQS (Distributed Query System)

DQS has been developed at CRAI, Italy as a commercial system based on the research and development experience stemming from NDMS development work. It provides facilities for distributed query processing on an integrated collection of heterogeneous databases residing in a SNA network of IBM mainframes. The system is integrated with the CICS TP environment and currently supports integration of databases controlled by IMS/VS, IDMS DB/DC and ADABAS database management systems. Integration of operating system access method files (ie. VSAM) is also supported. The global schema is relational and SEQUEL has been implemented as the distributed query language. Distributed transaction processing is not supported.

AIDA (Architecture for Integrated Data Access)

AIDA has been developed at the System Development Corporation, USA. It allows the user of one or more existing databases stored under one or more relational DBMSes to access data using a common language, either ARIEL or SQL. AIDA includes two components that have been developed within the MERMAID project, the MERMAID testbed and the ARIEL language and translator. Similar approach has also been adopted with respect to query optimization. The important feature of the system is an attempt to integrate the distributed data, text, image, voice, and knowledge processing.

The above presentation indicates clearly that most of the systems do not match the principal user requirements formulated in section 1 of this paper. This is the result of the evolving multidatabase system technology and the varying objectives of the various organizations sponsoring the research and development work. Although the relational data model prevails as the global schema representation, many systems do not support heterogeneous data models of the underlying databases. They are still considered multidatabase management systems, because integrated access to databases managed by diverse DBMSes is supported.

In most cases, distributed transaction processing is not supported. This is due to unresolved problems of concurrency control in the multi-DBMS environment. The difficulty results from the fact that multidatabase management systems utilize the host DBMSes on the level of user interfaces, that is on the query language or the data manipulation language level. Hence, it is impossible to apply most of the concurrency control algorithms developed for the distributed database management systems.

The lack of cooperation between local and global locking mechanisms may, for example, lead to undetectable global deadlocks. A detailed presentation of concurrency control problems in multi-DBMS environment may be found in [GLI84a, GLI84b].

4. FUTURE DEVELOPMENT TRENDS

Although little production level application experience exists at this moment, the present commercial software product development projects in the area of MDBMS, supported by the various user requirement and marketing studies, are indicative of the emerging new data management software technology.

It seems that the commercial systems, presently under development, will reflect the present state-of-the-art of prototype systems presented in this paper. The major issue in this area is the increase of portability of the MDBMS software, both in the sense of new underlying DBMSes and the new computer system hardware/software environments.

Increased portability will be achieved through standardization of the MDBMS node software architecture, comprising both the interfaces and functionalities. It seems that about 80% of the MDBMS control software is independent of the specific computer system hardware/software environment and may be directly ported to other environments. The remaining internal system functions must be designed and implemented according to the technical characteristics of the specific computer system hardware/software environment.

Problems that are still outstanding, both in terms of research results and the development work, fall into the following areas; high-level man-machine interfaces, multidatabase system design and implementation methodologies and tools, and system control function algorithms, mainly in the area of query and transaction processing.

The high-level man-machine interface must provide facilities appropriate to the different levels of skill and experience of the variety of end-users working in the multidatabase system environment. It should also provide development staff with comprehensive set of application development tools, thus simplifying the task of producing new applications.

The growing number of multidatabase systems and the inherent complexity of data and function mappings will stimulate efforts in the area of design methodologies and tools. This will cover both the semantic and functional design areas and the performance optimization and tuning area.

Query processing requires further improvements in the area of distributed query optimization. The present architecture of the MDBMS node software, in particular the intensive use of intermediate storage, opens new possibilities in the area of query processing. One of examples may be dynamic construction of index structures supporting the relational

algebra operations.

Transaction processing may only be applied in production level systems, if satisfactory solutions in the areas of multidatabase updates and distributed concurrency control will be found and implemented. The principal problem of implementing multidatabase updates is that of providing means to enforce the global and local integrity constraints. The concurrency control algorithms must consider the functional incompatibility of concurrency control mechanisms of the underlying DBMSes.

REFERENCES

- BRE84 Breitbart, Y.J., Tieman, L.R., ADDS - Heterogeneous Distributed Database System, Proc. of the 3rd Int. Seminar on Distributed Data Sharing Systems, F.A. Schreiber and W. Litwin (Eds), Parma, Italy, March, 1984, North Holland, 1985.
- BRE86 Breitbart, Y., Olson, P.L., Thompson, G.R., Database Integration in a Distributed Heterogeneous Database System, Proc. of the Int. Conference on Data Engineering, February 5-7, 1986, Los Angeles, USA, IEEE Computer Society, 1986.
- BRI84 Brill, D., Tempelton, M., Yu, C., Distributed Query Optimization in MERMAID: a Frontend to Data Management Systems, Proc. of the Int. Conference on Data Engineering, Los Angeles, April 1984.
- CER84 Ceri, S., Pelagatti, G., Distributed Databases - Principles and Systems, McGraw-Hill, 1984
- CODD71 Codd, E.F., A Data Base Sublanguage Founded on the Relational Calculus, Proc. of the 1971 SIGFIDET Workshop, San Diego, USA, November 11-12, 1971, ACM, 1971.
- DAY82 Dayal, U., Hwang, H.Y., View Definition and Generalization for Database Integration in Multibase: A System for Heterogeneous Distributed Databases, Proc. of the 6th Berkeley Workshop on Distributed Data Management and Computer Networks, May 1982.
- DQS88 Belcastro, V., Dutkowski, A., Kaminski, W., Kowalewski, M., Mallamaci, L., Mezyk, S., Mostardi, T., Scrocco, F.P., Staniszkis, W., Turco, G., An Overview of the Distributed Query System DQS, Proc. of the Int. Conference of Extending Database Technology - EDBT '88, Venice, Italy, March 14-18, Springer-Verlag.
- GLI84a Gligor, V.D., Luckenbaugh, G.L., Interconnecting Heterogeneous Database Management Systems, IEEE Computer, Volume 17, Number 1, 1984.
- GLI84b Gligor, V.D., Popescu-Zeletin, R., Concurrency Control Issues in Distributed Heterogeneous Database Management Systems, Proc. of the 3rd Int. Seminar on Distributed Data Sharing Systems, F.A., Schreiber, W. Litwin (Eds.), Parma, Italy, March 1984, North-Holland 1985.
- LAN82 Landers, T., Rosenberg, R.L., An Overview of Multibase, Proc. of the 2nd Int. Symposium on Distributed Data Bases, H.-J. Schreiber (Ed.), Berlin, September 1982, North-Holland, 1982.
- LIT82 Litwin, W., Boudenant, J., Esculier, C., Ferrier, A., Glorieux, A.M., La Chimia, J., Kabbaj, K., Molinoux, C., Rolin, P., Stangret, C., SIRIUS Systems for Distributed Data Management, Proc. of the 2nd Int. Symposium on Distributed Data Bases, H.-J. Schreiber (Ed.) Berlin, September 1982, North-Holland, 1982.
- MOH84 Mohan, C., Recent and Future Trends in Distributed Data Base Management, Proc. of the NYU Symposium on New Directions in Database Systems, May 1984.
- MOS87 Mostardi, T., Staniszkis, W., Multidatabase System Design Methodology, Proc. of the 10th Int. DBMS Seminar, Cedzyna, Poland, 1987.
- SMI81 Smith, J.M., Bernstein, P.A., Dayal, U., Goodman, N., Landers, T.A., Lin, W.-T.K., Wong, E., MULTIBASE - Integrating Heterogeneous Distributed Database Systems, Proc. AFIPS NCC, Volume 50, 1981.
- STA83 Staniszkis, W., Kowalewski, M., Turco, G., Krajewski, K., Saccone, M., Network Data Management System - General Architecture and Implementation Principles, Proc. of the Third International Conference on Engineering Software, Imperial College of Science and Technology, London, England, April, 1983.
- STA84 Staniszkis, W., Kaminski, W., Kowalewski, M., Krajewski, K., Mezyk, S., and Turco, G., Architecture of the Network Data Management System, Proc. of the 3rd Int. Seminar on Distributed Data Sharing Systems, F.A. Schreiber and W. Litwin (Eds.), Parma, Italy, March, 1984, North-Holland, 1985.
- STA85 Staniszkis, W., Distributed Database Management, State-of-the-Art and Future Trends, Proc. of the 8th Int. Seminar on Database Management Systems, October, 1985, Piestany, Czechoslovakia.
- TEM83 Tempelton, M., Brill, D., Hwang, A., Kameny, I., Lund, E., An Overview of the MERMAID System - A Frontend to Heterogeneous Databases, Proc. of the 16th Annual Electronics and Aerospace Conference, Washington D.C., September, 1983.
- TEM86a Tempelton, M., Brill, D., Chen, A., Lund, E., MERMAID - Experiences with Network Operation, Proc. of the Int. Conference on Data Engineering, February 5-7, 1986, Los Angeles, USA, IEEE Computer Society, 1986.
- TEM86b Tempelton, M., Brill, D., Chen, A., Dao, S., Lund, E., MacGregor, R., Ward, B., Introduction to AIDA - A Front-end to Heterogeneous Databases, to appear in a IEEE publication.

Janusz STOKŁOSA *

DES, model OSI,
standardy ISO

O STANDARDACH KRYPTOGRAFICZNYCH ISO DLA SIECI KOMPUTEROWYCH

W artykule scharakteryzowano właściwości algorytmu DES i zarysowano tok prac prowadzonych w ISO nad normą dla tego algorytmu. Zwrócono uwagę na aktywność ISO w zakresie ochrony kryptograficznej w modelu OSI, a także w sferze zarządzania kluczami i uwierzytelniania.

1. UWAGI WSTĘPNE

Od połowy lat siedemdziesiątych problematyka kryptografii komputerowej zajmuje informatyków ze względu na możliwości jej zastosowań do ochrony informacji przesyłanych pomiędzy organizacjami handlowymi, bankowymi, użytkownikami indywidualnymi itp. Jest znanym fakt, że banki i inne organizacje finansowe korzystają z metod kryptograficznych, a dzienne transakcje bankowe dokonywane za pomocą łączy telekomunikacyjnych szacuje się na ok. 10^{12} \$ [3]. W tej sytuacji korzystanie ze skutecznych algorytmów stało się sprawą o istotnym znaczeniu.

2. DES

W 1977 r. Narodowe Biuro Standardów USA opublikowało normę DES (Data Encryption Standard) szyfrowania danych [6]. Algorytm ten wybrano spośród wielu będących w użyciu. Został on opracowany w IBM. Także inna organizacja normalizacyjna w Stanach Zjednoczonych, ANSI, przyjęła DES jako standard w 1981 r. [1].

DES jest symetrycznym produktowym szyfrem blokowym [7]. Działa na 64 bitach tekstu jawnego jednocześnie. Jest szyfrem produktowym ponieważ składa się z sześciu nieregularnych transpozycji bitów wykorzystujących 16 wariantów klucza. Pomimo dużej złożoności algorytm DES jest tylko prostym szyfrem podstawieniowym. Podstawienia są wykonywane przez 32 bloki podstawień, z których każdy zawiera permutacje cyfr szesnastkowych O+P. Kompozycja bloków podstawień powoduje, że DES jest odporny na

* Ośrodek Informatyki Politechniki Poznańskiej, Poznań.

złamanie. Badając w 1978 r. algorytm DES Agencja Bezpieczeństwa Narodowego USA (NSA) stwierdziła, że jest on pozbawiony słabych punktów zarówno z matematycznego, jak i statystycznego punktu widzenia [8]. Od tamtego czasu poświęcono dużo uwagi analizie właściwości algorytmu DES [3,4,5,10,11]. Wykryto przy tym trzy następujące regularności:

- własność dopełnienia; dla każdego klucza k i wiadomości m zachodzi $DES_k(\bar{m}) = \overline{DES_k(m)}$, przy czym $\bar{x}$ oznacza negację słowa binarnego x ,
- istnienie kluczy słabych; istnieją co najmniej 4 klucze k takie, że $DES_k^2 = I$, przy czym I jest przekształceniem tożsamościowym,
- istnienie kluczy półsłabych; istnieje co najmniej 6 różnych par kluczy (k, k') , $k \neq k'$, takich że $DES_k \circ DES_{k'} = I$.

Od 1980 r. w Międzynarodowej Organizacji Standaryzacyjnej (ISO) trwały prace nad tekstem normy obejmującej algorytm DES (w ISO przyjęto dla niego akronim DEA1). W roku 1983 opublikowano jego wstępną wersję [9], którą poddano pod dyskusję. Po 6 latach od rozpoczęcia prac zdecydowano jednak nie prowadzić ich dalej. Głównym powodem było przeświadczenie, że przyjmując DES jako standard uzależniono by się zbyt od jednego algorytmu. Przerywając ukończone już prawie prace nad standardem kierowano się w ISO obawą, że istnienie jednej normy szyfrowania danych naraziłoby wielu użytkowników na ataki intruzów, którzy w takiej sytuacji mieliby ułatwione zadanie. W kwietniu 1987 r. przyjęto inne rozwiązanie polegające na utworzeniu rejestru (wykazu) algorytmów szyfrujących [12], który zawierałby algorytmy opublikowane i nieopublikowane. Taki rejestr dawałby potencjalnym użytkownikom możliwość wyboru. Umieszczenie algorytmu niepublikowanego w rejestrze zależałoby wyłącznie od decyzji jego odkrywców.

Należy w tym miejscu nadmienić, że zastosowanie superkomputerów umożliwia złamanie algorytmu DES - jak się szacuje [3] - w ciągu kilkunastu dni. W tej sytuacji NSA ogłosiła, że po roku 1988 nie będzie udzielać atestów na układy i programy wykorzystujące ten algorytm. Można jednak przewidywać, że w wielu zastosowaniach, przy uwzględnieniu odpowiednich okresów stosowania kluczy, DES będzie jeszcze długo używany w kryptografii komputerowej, m.in. ze względu na realizujące go szybkie układy scalone dostępne w handlu [13]. Jednocześnie są prowadzone prace zmierzające do zaprojektowania nowej, bezpieczniejszej odmiany układu realizującego algorytm DES.

3. STANDARDY KRYPTOGRAFICZNE DLA MODELU OSI

Jednym z efektów zaprzestania przez ISO prac nad normą szyfrowania danych jest opóźnienie momentu opublikowania dwóch innych standardów, które także były w zaawansowanym stanie przygotowania. Są to: algorytm o sposobach 64-bitowego szyfrowania blokowego oraz standard wzbogacają-

cy warstwę fizyczną modelu OSI w możliwości szyfrowania. Dokument o sposobach szyfrowania wymaga niewielkiej pracy wynikającej z nieopublikowania tekstu standardu DES. Został on zredagowany ogólnie, uwzględniając szyfrowanie blokowe 64-bitowe (niekoniecznie DES). Stąd jego publikacja jest oczekiwana wkrótce. Także przekazanie pod dyskusję wstępnej wersji tekstu dotyczącego szyfrowania w warstwie fizycznej ma nastąpić niebawem. Ponadto w ISO trwają prace nad przygotowaniem normy szyfrowania blokowego nie ograniczającej się do bloków 64-bitowych.

Wszystkie prace na temat zwiększania bezpieczeństwa w warstwie 2 (połączeń) modelu OSI zostały zawieszone, ponieważ zgodzono się, że w powstałej sytuacji standard nie mógłby być przyjęty. Potrzeba wprowadzenia ochrony w warstwie 3 (sieciowej), 4 (transportowej) i 6 (prezentacji) jest uznana, a prace dotyczące - jak to ujęto w roboczym opracowaniu ISO [12] - "warunków wykonywania stosownych operacji związanych z ochroną kryptograficzną" postępują naprzód. Problematykę bezpieczeństwa w modelu OSI omówiono w [2].

4. ZARZĄDZANIE KLUCZAMI I UWIERZYTELNIANIE

Co do zarządzania kluczami, to program prac prowadzonych w ISO jest bardziej precyzyjny. Dotyczą one zarządzania kluczami dla algorytmów symetrycznych z zastosowaniem technik kluczy tajnych, zarządzania kluczami dla algorytmów symetrycznych z zastosowaniem technik kluczy publicznych, zarządzania kluczami dla algorytmów niesymetrycznych z zastosowaniem technik kluczy publicznych.

W tych pracach jest także reprezentowana problematyka uwierzytelniania. Są przygotowywane opracowania dotyczące podpisów cyfrowych "ukrytych" (shadow) i "wdrukowanych" (imprint). Rzecz dotyczy algorytmów, odpowiednio, bezpośrednio realizujących podpis i metod podpisywania z wykorzystaniem funkcji haszującej (do tych prac są też włączone sposoby generowania funkcji haszujących).

Szczególne uwagę w pracach ISO poświęca się procedurom bankowym. Opracowano serię norm ukierunkowanych na autentyczność i zarządzanie kluczami. Trwają prace nad standardem uwierzytelniania wiadomości dotyczących handlu hurtowego i nad standardem specyfikującym algorytmy, które mogą być stosowane do uwierzytelniania wiadomości. Jest także w przygotowaniu norma dotycząca zarządzania kluczami w procesach finansowych związanych z handlem; punktem wyjścia jest w tym przypadku norma ANSI X9.17 [12].

Warto odnotować, że w pracach tych jak dotąd nie uwzględnia się zastosowań kryptografii publicznej do celów bankowych; wszystkie one dotyczą symetrycznych systemów kryptograficznych.

5. ZAKOŃCZENIE

Prace w zakresie przetwarzania informacji są prowadzone w ISO w Komitecie Technicznym 97 (TC 97). Podkomitet 20 (SC 20) opracowuje zagadnienia dotyczące technik kryptograficznych. TC 97/SC 6 prowadzi prace dotyczące warstw 1+4 modelu OSI, natomiast TC 97/SC 21 zajmuje się warstwami 5+7. Normy ochrony danych w bankowości są przedmiotem prac komitetu TC 68.

LITERATURA

1. ANSI X3.92-1981, Data Encryption Algorithm, American National Standards Institute, New York, 1980.
2. Buffenoir T., Security in the OSI model. Standards & Interfaces 7, 1/2(1988), 145-150.
3. Carrol J.M., Strategies for extending the useful lifetime of DES. Computers & Security 6, 4(1987), 300-313.
4. Chaum D., Everste J.-H., Cryptoanalysis of DES with reduced number of rounds sequences of linear factors in block ciphers. Williams H. C. (ed.), Advances in Cryptology - CRYPTO '85, LNCS 218, Springer, Berlin, 1985, 192-211.
5. Desmedt Y., Quisquater J.-J., Davio M., Dependence of output on input in DES: small avalanche characteristics. Blakley G.R., Chaum D. (eds.), Advances in Cryptology: Proceedings of Crypto 84, LNCS 196, Springer, Berlin, 1984, 359-376.
6. FIPS Publication #46, Data Encryption Standard, Federal Information Processing Standard, National Bureau of Standards, U.S. Department of Commerce, Washington, DC, 1977.
7. Hoffman L.J., Poufność w systemach informatycznych. WNT, Warszawa, 1982.
8. Invalvment of NSA in the development of the Data Encryption Standard, Staff Report of the Senate Select Committee on Intelligence, United States Senate, 1978.
9. ISO/DP 8227, Data encipherment, specification of algorithm DEA1, 1983.
10. Kaliski B.S. jr., Rivest R.L., Sherman A.T., Is the Data Encryption Standard a group? (Results on cycling experiments on DES). Journal of Cryptology 1, 1(1988), 3-36.
11. Moore J.H., Simmons G.J., Cycle structure of the DES for keys having palindromic (or antipalindromic) sequences of round keys. IEEE Trans. Software Engineering SE-13, 2(1987), 262-273.
12. Price W.L., Standards for data security - a change of direction. Pomerance C. (ed.), Advances in Cryptology - CRYPTO '87, LNCS 293, Springer, Berlin, 1987, 3-8.
13. Stokłosa J., LSI and VLSI realizations of cryptographic algorithms for microprocessor systems. IV Międzynar. Konf. "Relcomex '87", vol.2, 1987, 363-370.

ON THE ISO CRYPTOGRAPHIC STANDARDS FOR COMPUTER NETWORKS

In the paper the activity of the ISO dealing with the DES algorithm is presented. Cryptographic standards for the OSI model and for the banking are discussed.

electronic mail,
X. 400

Ewan Sutherland †

X.400 Developments in the United Kingdom

UK industry and commerce are adopting X.400 electronic mail, initially through the use of gateways and later full products based on the 1984 version of the standard. Much of the interest is in the use of X.400 as a vehicle for Electronic Data Interchange (EDI). Academic users are taking advantage of their well-established position to go direct to the 1988 version. Suppliers are writing their own software or are acquiring X.400 shells which they can modify to fit their requirements. Overall there is much interest in X.400 with many real developments in use and the prospect of many more.

Introduction

Developments in the UK are well underway in the adoption and use of both the 1984 [1] and 1988 [2] versions of X.400, although the latter is only very recently available. It is slightly disturbing to consider that 1990 and 1992 may bring new versions; though the effect of these will depend on the significance of the changes they contain. The UK has been an active participant in the standardisation process for many years.

One indication of the growth in the facsimile market, the main rival to X.400, is that over 200,000 units were sold in the UK in 1988. A machine is now treated as a conventional desk top item for many managers. Recent growth in the facsimile market and, in particular, the rise in management awareness has clearly detracted from the potential growth of electronic mail.

From business and commerce there is growing interest in X.400 to link the very many private systems in firms and in its application to electronic data interchange (EDI) [3]. A useful measure of this has been the large number of conferences, reports and seminars on X.400 in 1988 and 1989 [4].

Profiles

The vast array of possible implementations of X.400 has given rise to a number of profiles of which the most important are the European Norms [5] of the Joint European Standards Institute (CEN, CENELEC and CEPT). British Telecom has its own OSI profile, known as Open Network Architecture (ONA) [6], which includes a profile for message handling systems based on the European Norm. The Central Computer and Telecommunication Agency (CCTA) acts as an internal consultancy for the UK Government. It has issued an X.400 profile as part of its general Government OSI Profile (GOSIP) [7]. These profiles are now widely supported by suppliers and are being used in procurement by the public sector. The CCTA publishes a review of OSI conformance of products by leading British and foreign firms [8].

British Telecom

British Telecommunications plc (BT) the main, and formerly only, UK supplier of telecommunication services has multiple interests in electronic mail as telecommunications provider, service provider and as a vendor of software.

Initially BT licensed electronic mail software from Dialcom Inc, a software company in the USA. It bought the company in 1986. The service provided with the Dialcom software is marketed in the UK as *Telecom Gold* and has grown from 200 users in 1982 to over 105,000 mailboxes. However, it is thought that as many as 40,000 of these may belong to BT staff and that only 30,000 others are regular users. Dialcom software is licensed in 21 countries having grown from 14 countries amounting to another 175,000 users.

Telecom Gold provides access to a user directory, a gateway to telex and many online databases (e.g., Petroleum Monitor and the Official Airlines Guide). It will also arrange for a BT Radiopager to "bleep" when new mail arrives so that a user can be informed of this almost anywhere in the UK. Mail can be sent to subscribers of the international licensees of Dialcom. The software is being converted to conform with X.400 (1984), a process which is expected to be completed during the course of 1989.

† The University, Stirling, FK9 4LA, Scotland.

The initial growth of Telecom Gold has been substantially based on the gateway to telex which provided the users of electronic mail with a large existing community to communicate with. An optional direct dial-in (DDI) service for telex was recently introduced, while a one-way gateway operates to group 3 facsimile.

BT has operated as an X.400 administrative management domain (ADMD) since June 1988 providing a service known as *Gold 400*. A number of X.400 private management domains (PRMD's) are connected to this, e.g., Digital Equipment Corporation, IBM, ICL, ITL, IDEM and JANET. So far the only connection to another ADMD is to France Telecom's *Atlas 400*. BT is a member of the International Administration Domain Operators' Group (IAOG). A one-way gateway has recently been opened to Group 3 facsimile as has a two-way gateway to telex.

BT has abandoned its teletex [9] service because of the lack of demand. In France and West Germany the PTT's persist, though in the face of very low demand. There are probably less than 100,000 users worldwide.

BT operates an analogue voice mail messaging service known as *VoiceBank*. This uses touch tone keypads and can be linked to a radiopager.

BT markets a range of X.400 software, for personal computers, file servers, the DEC VAX range and Unix. Development work is being carried out on Office Document Architecture (ODA). [10]

Commercial Mail Services

In addition to British Telecom, there are a number of other providers of electronic mail services in the UK; Mercury (part of Cable and Wireless plc) [11], Telemail (part of Telenet Inc of the USA) [12], ISTELE [13] and One-to-One [14]. These companies offer electronic mail services plus gateways to telex, physical delivery of mail and some additional services such as access to databases.

Telemail Inc of the USA offers an X.400 service which at present links the UK with Japan, Australia and the USA, mainly to Telemail users. Mercury has indicated rather reluctant interest in the area, while ISTELE is participating in the ITUSA pilot (see below).

The British Standards Institute (BSI) is currently working on a standard for ADMD's to cover the interconnection of ADMD's and the uniqueness of PRMD and Organisation names within and between ADMD's.

Commerce and Industry

The business community has a considerable potential for improved communications as demonstrated by the demand for telex, facsimile, mobile telephones, etc. Many firms have large internal electronic mail systems, often extending internationally, but are constrained to operate within the firm. Interconnection has only been possible by use of telex and facsimile with the inefficiencies they imply.

The Information Technology Users Standards Association (ITUSA) [15] conducted a study into X.400 in 1986 which concluded that the absence of a gateway to IBM systems was an insuperable barrier [16]. It is currently conducting a Pilot Study into the use of X.400 in conjunction with Sydney Ltd. Initially, it runs X.400 over a proprietary asynchronous protocol to provide access on dial-up telephone lines, but it is being extended to X.25. The absence of an asynchronous protocol has been identified as a major omission from X.400. The participants in the study are mainly large UK commercial firms. Sydney markets X.400 products for VMS, Unix, Xenix and MS-DOS operating system [17].

Most of the better known X.400 products are available in the UK, e.g., DEC's Mailbus with its X.400 gateway. Lucas Industries are currently undertaking a β -test on the gateway from IBM's DISOSS to X.400 with connections to Gold 400 and to British Aerospace [18].

The Department of Trade and Industry sponsors an implementors' group on message handling systems, comprising firms implementing or intending to implement X.400 software. It is an opportunity for firms to discuss non-commercial issues such as difficulties and ambiguities in standards and profiles and to feed comments to the British Standards Institute (BSI), ISO, CCITT, etc. It works closely with CCTA on the GOSIP profile. There are also implementors' groups dealing with the Directory Service and Office Document Architecture.

A number of UK firms sell X.400 products. Logica plc, a leading software house, is selling an X.400 (1984) shell. Scicon-SD sells a number of products, including one based on P3 [19]. Case Ltd (part of the Dowty Group) has products to complement its well established range of message and telex switches [20]. ICL is now offering X.400 conformance as part of its OfficePower office information system running on Unix and intends to provide X.400 on its VME machines [21]. Net-Tel Ltd have a PC product called Route400 [22]. ITL plc markets a product on fault tolerant computers which includes secure mailboxes [23]. Xionics provides X.400 conformance in its office system [24].

Clearly this level of product development is not based on original implementations of X.400. Many firms are buying X.400 shells, e.g., from Logica or Retix, and modifying them for their own use. This process is simplified by the provision of abstract syntax notation (ASN.1) compilers. The failure of P3-based products has resulted in a vacuum in the market, it is not clear whether PC products will be based on the new P7 protocol with message store or whether they will use co-resident user agent and message transfer agent.

Electronic Data Interchange

The market for electronic data interchange (EDI) is growing rapidly in the UK, in most economic sectors. In part it is related to the push for reduced costs and to the enthusiasm for Just-In-Time manufacturing. [25]

There are significant technical differences between Electronic Data Interchange (EDI) and X.400. The store and forward nature of X.400 restricts its applicability in EDI applications. The different encoding techniques, EDI does not use ASN.1, also present problems, in particular the absence of an EDI body part or content type for EDI. Therefore, there will require to be mutual changes and refinements to the standards.

X.400 offers, though does not yet provide, universal connectivity for computers, networks and value added data services (VADS). It is in the shared use of an infrastructure of networks and message transfer agents that the benefits lie, though probably with different user agents. The market success of one will support the growth of the other.

X.400 is good platform for other applications. At present work is being funded by the CEC on the use of X.400 and/or EDI for inter-library lending.

Government

The Directives of the Commission of the European Communities (CEC) require all public authorities, including most of the education sector, to put computer and networking purchases out to open tender and require that networking conform to OSI, except where a more adventurous approach is being taken [26].

CCTA runs Inter-Departmental Electronic Mail (IDEM), a service for central Government providing an electronic mailboxes and also connects together Government departmental mail systems using X.400 (1984) [27]. To date, there has been a very slow take-up rate of X.400 in municipal government.

Academic Community

The UK Academic Community (UK.AC) comprising the universities, polytechnics and research establishments is a well established user of electronic mail with the Grey Book protocol [28] running over the Blue Book file transfer protocol [29] and X.25 (1980) on its Joint Academic Network (JANET) [30]. Despite the benefits obtained from the use of Grey Book its days are clearly numbered. Current implementation efforts are being directed into work on X.400 software rather than further versions of Grey Book. Grey Book software has been written in the universities though it has been "adopted" by some hardware suppliers.

The Joint Network Team (JNT) and the Computer Board for the Universities have a well-established migration policy for the network intending to achieve X.400 (1988) conformance as part of a general move to OSI [31]. The aim is to move to OSI as rapidly as practicable given available software, standards and funding.

X.400 (1984) was rejected because of the limited benefits it offered over Grey Book and the recognition of the likely length of the transition period. It avoids the double transition from Grey Book to 1984 then from 1984 to 1988 and makes it possible to go directly to use of Reliable Transfer Service (RTS) Normal Mode.

The aim is to have a basic IA5 (plain ASCII) text service in the first instance, with transparent transfer of other body parts. It will take some years before significant numbers of systems are available to receive or convert more complex body parts.

Having made the decision in favour of X.400 (1988) it will be important for the community to watch future developments in the standard. More importantly, it will be necessary to make clear to suppliers the type of software required, with high levels of functionality and a high quality user interface. The latter is especially important given the growing population of non-technical users. The major benefit of the adoption of X.400 should be the availability of commercial software of high quality and high functionality together with full support.

Given the large number and diversity of machines currently using Grey Book software migration will be slow. For older machines it would be purposeless to create new software, it will be simpler to wait for them to go out of service. For communications within an institution protocol conversions between Grey Book and X.400 are expected to be performed locally. Alternatively the organisation is expected to have a single protocol switch-over date. Where two sites are running different protocols they will make use of protocol converters on JANET, these are expected to be necessary until the mid-1990's.

That the choice between X.400 (1984) and X.400 (1988) should have been made for individual institutions is regrettable. The benefits of X.400 (1984) over Grey Book are modest if any. Unlike the commercial world, the academic community already has a functioning interconnection standard which operates well at national level and is complemented by international gateways. However, the software and the user interfaces are seldom excellent.

Efforts are being made to produce a Unix version of X.400 (1984) software by second quarter 1989 and X.400 (1988) by the third quarter, both from University College London (UCL) [32]. Now that P7 is fully defined, work is being commissioned on mail systems for Unix machines, IBM PC's and Apple Macintoshes.

An experimental gateway based on the EAN software from the University of British Columbia [33] is operated for X.400 (1984), this will shortly be replaced with the UCL software. It reaches a few UK organisations, e.g., Hewlett Packard's Research Laboratory at Bristol and the Government's IDEM together with the French and German research networks. Direct traffic with UK PRMD's will be initiated where sufficient volume warrants it. Similarly, direct traffic between MTA's would be established when judged worthwhile. These decisions will largely be a matter of traffic volume and tariffs. A link has been established to BT's Gold 400 service but is not generally available. A 64Kbps link to the National Science Foundation Network (NSFNET) in the USA is being established over the new trans-Atlantic optical fibre cable TAT-8 which will carry X.400 mail amongst other traffic.

Using software from an ESPRIT project based on an early version of X.500 [34], a trial Directory Service has been established running on Unix machines at University College (London), Rutherford Laboratory, Brunel University, etc [35]. Queen Mary College are conducting Office Document Interchange Format (ODIF) tests [36].

Conclusion

Progress in the commercial and industrial world is rapid and substantial at all levels, from ADMD down to individual users. For the future the key issues which will affect developments are

- interconnection of the large user groups of proprietary office information systems
- availability of X.500 software integrated into electronic mail software
- availability of software supporting ODA, ODIF and Hypertext
- lower tariffs and competition

The academic community seems to be well ahead of commerce and industry, though this has always been the case in electronic mail. Once fully conformant X.400 software is on the market this is likely to cease. With the greater resources available to commercial enterprises they will soon overtake academic users.

Facsimile is on, possibly, its last burst of growth, driven by the recent postal strike (late summer 1988). There are few, if any, signs of demand for Group 4 facsimile. However, this will depend on ISDN tariffs and, if these prove favourable, will require the growth of a mass market to allow the decline in unit costs.

References

- 1 CCITT "X.400 Message Handling Systems - System Model Service Elements" (CCITT, Geneva, 1984).
- 2 ISO/IEC "Message Oriented Text Interchange Standard" 10021 (ISO/IEC, Geneva, 1989).
- 3 Department of Trade and Industry "Vanguard: EDI and X.400 Study" (HMSO, London, 1988).
- 4 e.g. "Electronic Message Systems 88" (Blenheim Online, Pinner, Middlesex, 1988), "X.400 Markets: the users decide" (Ovum Ltd, Rathbone Street, London, W1P 1AF, 1989) and Carl Edgar Law "X.400 and OSI: electronic messaging in the 1990's" (IBC Technical Services, Weybridge, Surrey, 1988).
- 5 "Private Message Handling Systems; user agent and Message Transfer-agent; private management domain to private management domain" ENV 41 201 and ENV 41 202 (CEN/CENELEC, Brussels, 1988).
- 6 "The Open Network Architecture Tutorial Guide and Compendium of Profiles" (British Telecom, Ipswich, 1988).
- 7 UK Government OSI Profile - Message Handling Systems (CCTA, London, 1988).
- 8 CCTA "Open Systems Interconnection - a second review of suppliers implementations" (HMSO, London, 1988).
- 9 CCITT "T.73: Document Interchange Protocol for the Telematic Services" (CCITT, Geneva, 1984).
- 10 Dialcom Ltd, 60-68 St Thomas St, London SE1 3QU, UK.
- 11 Mercury Link 7500, Mercury House, Great West Road, Brentford, TW8 9DS, UK.
- 12 Telemail Ltd, 15 Whitehall, London, SW1A 4DD, UK.
- 13 ISTE Ltd, PO Box 5, Redditch, B97 4DQ, UK.
- 14 One-to-One Ltd, Scorpio House, 102 Sydney Street, Chelsea, London, SW3 6BR, UK.
- 15 ITUSA, CentrePoint, New Oxford Street, London, WC1A 1DU, UK.
- 16 National Computer Centre "ITUSA X.400 Study Second Report" (ITUSA, London, 1986).
- 17 Sydney Ltd, Victoria House, Aldershot, Hampshire, GU11 1EJ, UK.
- 18 Lucas Ltd, PO Box 52, Shirley, Solihull, West Midlands, B90 4JJ, UK.
- 19 SD-Scicon, 2 Bessborough Gardens, Vauxhall Bridge Road, London, SW1V 2JE, UK.
- 20 Don Cochrane "Easing the Migration into the Future of Communications Technology" *Computer Networks and ISDN Systems* 14 pp 323-329 (1987).
- 21 ICL Ltd, Bridge House, Putney Bridge, London, SW6, UK.
- 22 Net-Tel Ltd, 4 Place Farm, Wheathampstead, AL4 8SB, UK.
- 23 Information Technology plc, Maylands Avenue, Hemel Hempstead, HP5 7DF, UK.
- 24 Xionics Ltd, 4 Dollis Park, London, N3, UK.
- 25 "The EDI Handbook; trading in the 1990's" (Blenheim Online, Pinner, Middlesex, 1988).
- 26 Commission of the European Communities *Official Journal* L 36 pp 31-35 (1987).
- 27 CCTA "Inter-Departmental Electronic Mail - stage two" (HMSO, London, 1986).
- 28 Steve Kille (editor) "The JNT Mail Protocol" (Joint Network Team, Rutherford Appleton Laboratory, Didcot, Oxfordshire, 1984).
- 29 File Transfer Protocol Implementors Group "A Network Independent File Transfer Protocol" (Joint Network Team, Rutherford Appleton Laboratory, Didcot, Oxfordshire, 1981).
- 30 Robert Cooper "The JANET Internet" *Computer Bulletin* 4 (2) pp 10-11 (1988).
- 31 Academic Community OSI Transition Group "Transition to OSI Standards" (Joint Network Team, Rutherford Appleton Laboratory, Didcot, Oxfordshire, 1987).
- 32 Steve E Kille "PP - a message transfer agent" in *Proceedings of the IFIP WG 6.5 Conference on Message Handling Systems, Irvine, California, October, 1988* (North-Holland, Amsterdam, 1988).
- 33 Gerald Neufeld *et al* "EAN: an X.400 message system" pp 3-15 in *Computer Message Systems '85* Edited by R Uhlig (North-Holland, Amsterdam, 1986).
- 34 CCITT "X.500 The Directory - overview of concepts, models and services" (CCITT, Geneva, 1988) (see also ISO 9594)
- 35 Department of Computer Science, University College, Gower Street, London, UK.
- 36 QMC Interactive Systems Ltd, Queen Mary College, London, E1 4NS, UK.

distributed data base,
data replication,
relational algebra

Petr Šíroky*

TRANSFORMATION OF UPDATES IN DISTRIBUTED DATA BASES WITH REPLICATED DATA

Update on a copy in distributed database systems with replicated data requires sending appropriate updates to other copies of the same data. This paper deals with the problem of finding such updates in an environment with complex replication strategy. A simple model of data replication is introduced. General operators for the description of both data replication and update operations are presented. A solution to the problem based on relational algebra is briefly described. Results are presented in SQL-notation.

1. INTRODUCTION

There are two basic approaches to data distribution and allocation in distributed data bases: partitioning and replication. In partitioning, the whole data base is split into parts, each of which is located at exactly one node in the network. Replication means that there are several copies of data distributed over the nodes.

Motives for data replication are: faster access to local data, lower communication costs, improved system performance because of increased parallelism, availability of data in case of network or node failures.

The main problem in replicated data bases is keeping various

* General Computing Center, Czechoslovak Academy of Sciences, Praha

copies of data mutually consistent - every update must be realized at all nodes that store a copy of updated data. Transformation of the original update into subtransactions sent to other nodes and global and local synchronisation of update subtransactions at various nodes (bringing about a large concurrency control overhead) are the two most important difficulties.

This paper deals with the first aspect of the problem. It describes a strategy for creating transformed subtransactions for various update operations and complex replication strategies. Replication conditions, update operations and resulting subtransactions are handled in terms of relational algebra.

The paper is organized as follows: the second section introduces a model describing data replication in a distributed data base. The third section describes transformation of various update operations into subtransactions and is based on the model from the second section.

2. A MODEL OF DATA REPLICATION

We can view a replicated data base as a system of nodes with their local data bases, parts of which are copied to other nodes in the network.

Let $A, B, C, \dots$ be the local data bases and F_{IJ} , $I, J = A, B, C, \dots$ a part (a fragment) common for local data bases I and J (a copy of this fragment is located in both data bases) (Fig.1).

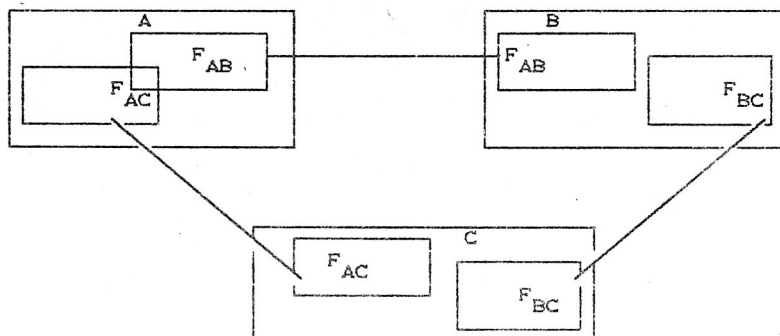


Fig.1

We will see data in a fragment as a logical unit rather than as physically stored data, which means that we can write $F_{IJ} = F_{JI}$. It is to be stressed that this paper is not concerned with conceptual aspects of replication, i.e. whether the fragment F_{IJ} is really a part of the same entity at both nodes or whether the information from the node I is used at J in other context.

We may introduce an operator W_{IJ} mapping a local data base I into a fragment F_{IJ} : $W_{IJ}(I) = F_{IJ}$; $W_{JI}(J) = F_{IJ}$. The operator W_{IJ} may be interpreted as a definition of a fragment F_{IJ} in the local data base I. Let us examine this operator in terms of the relational model of data commonly used in distributed database systems, both in theory and in practice.

We can view a local data base as a system of relations and the operator W_{IJ} can be implemented as a sequence of relational algebra operators. But not all of the relational algebra operators are appropriate for the definition of fragments. E.g. it is obvious that creating a fragment using join could mean vast physical storage overhead. We would like to restrict considerations in this paper to two basic relational operators: select and project [3].

We can formulate an algebraic expression defining a fragment. In SQL-notation it would be written as follows:

```
SELECT set of attributes for project
FROM relation
WHERE select condition
```

Note that the fragment F_{IJ} need not necessarily be an isolated relation at node J. It may be just a part of a bigger relation s : $F_{IJ} = W_{JI}(s)$.

Fig.2 An example of fragment definition [2]

relation r:	<u>ENAME</u>	<u>SAL</u>	<u>AGE</u>	<u>DEPT</u>
	Gauss	50	25	Math
	Laplace	45	30	Math
	Newton	50	35	Physics
	Einstein	75	50	Physics
	Russell	99	50	Philosophy
	Galileo	40	75	Astronomy
	Aiken	55	20	CS

```

WIJ:  SELECT ENAME, AGE, DEPT
      FROM r
      WHERE (AGE < 40) ^ ((DEPT = Math) v (DEPT = Physics))

```

```

fragment FIJ:
      ENAME      AGE      DEPT
      Gauss      25       Math
      Laplace    30       Math
      Newton     35       Physics

```

3. TRANSFORMATION OF UPDATE OPERATIONS

Let us introduce an operator Z representing an update operation on the relation r : $r' = Z(r)$.

For a relation in the fragment:

$$r'_{F_{IJ}} = W_{IJ}(Z(r)) \quad (1)$$

$$r'_{F_{IJ}} = Z^1(W_{IJ}(r)) = Z^1(r_{F_{IJ}}) \quad (2)$$

The expression (1) describes a replication of an updated relation, whereas (2) describes update of a replicated relation. The two expressions above yield the equation:

$$W_{IJ}(Z(r)) = Z^1(W_{IJ}(r)) \quad (3)$$

where Z^1 is the transformed update operation on the fragment F_{IJ} .

The update operator Z can be expressed (with some difficulties in the operation "change") as a sequence of relational operators similarly to W_{IJ} : "add" as union with a constant relation, "delete" as difference of tuples satisfying a certain condition and "change" as difference of tuples satisfying a certain condition and union with a relation representing new values.

Our goal is to find a sequence of relational operators representing Z^1 if Z and W_{IJ} are known. Using the equation (3) we can derive general expressions for every update operation (add, delete, change). Neither the derivations nor the resulting expressions can be displayed here, because the introduction of relational algebra formalism would mean exceeding the size of this paper.

Fig.3 shows some examples of updates to the original relation (Z) and corresponding update subtransactions (Z^1) using relations

from Fig.2 and simple SQL-notation.

Fig.3

```

1) Z: DELETE r
      WHERE (DEPT=Math) ∨ (DEPT=Astronomy)
Z1: DELETE r'
      WHERE (DEPT=Math) ∧ AGE < 40

2) Z: UPDATE r
      SET AGE=AGE+6
      WHERE (DEPT=Physics) ∨ (DEPT=Philosophy)
Z1: DELETE r'
      WHERE (AGE >= 34) ∧ AGE < 40 ∧ (DEPT=Physics)
      UPDATE r'
      SET AGE=AGE+6
      WHERE (DEPT=Physics) ∧ AGE < 40

```

4. CONCLUSIONS

Data replication and update operations have been described using general operators. A possible solution to the problem of finding appropriate transformed update subtransactions on fragments has been proposed using relational algebra.

REFERENCES

- [1] Ceri, S., Pernici, B., Wiederhold, G.: Distributed Database Design Methodologies
Proceedings of the IEEE, Vol.75, No.5, May 1987,
pp.533-546
- [2] Dayal, U., Bernstein, P.A.: On the Correct Translation of Update Operations on Relational Views
ACM TODS, Vol.8, No.3, September 1982, pp.381-416
- [3] Maier, D.: The Theory of Relational Databases
Computer Science Press, Rockville, 1983

fault-tolerance.

Omega network

G.G. VESELOVSKY *
A.Yu. ZALOZHNEV *
A.A. FUKHALSKY *

PERFORMANCE AND FAULT-TOLERANCE ANALYSIS FOR MODIFIED OMEGA NETWORK

Performance and fault-tolerance of multipath modified Omega network are investigated under a random distribution of request destinations over network outputs. Investigations are carried out by methods of analytical modelling and simulation. It is shown that performance of such a network is rather high and occurrence of $R-1$ arbitrary faults, where R is a number of paths for every input-output pair, only slightly affects it. Analysis and simulation were realized for the case of a two-stage 4-path 256 x 256 modified Omega network using 32 x 32 switches.

At present the problem of multistage interconnection networks utilization in computer networks and parallel computer systems is of utmost significance. To the rank of prospective topologies one can refer the highly fault-tolerant modified Omega network. Due to excessive interconnecting abilities of switching elements in this network availability of more than one disjoint paths is provided for each input-output pair [1].

Fig. 1 shows a 4-path 16 x 16 modified Omega network. In [1] the features of the modified Omega network are considered with respect to its application in synchronous parallel systems of the SIMD-type, i.e. for the case with a connection set known beforehand. In other words, input-output permuta-

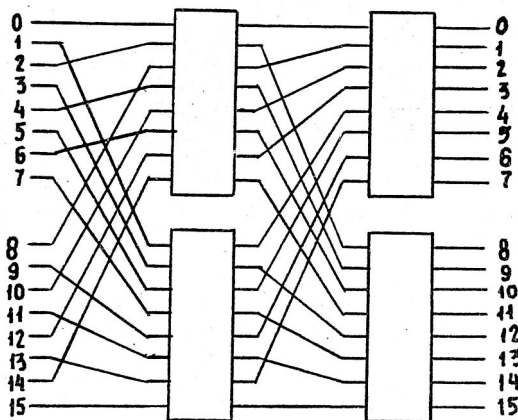


Fig. 1. A 4-path 16 x 16 modified Omega network.

*Institute of Control Sciences, Moscow, USSR

tion families realized by the ordinary 1-path Omega network. However, fault-tolerance is achieved at the cost of a multiple performance reduction i.e. in the presence of faults in the network an assigned permutation is realized in several passes through the network. A pass is a set of input-output connections in the network which do not conflict with one another.

As opposed to [1] in this paper the modified Omega network is being investigated under a random distribution of request destinations over network outputs, which is characteristic for computer networks and multiprocessor systems of the MIMD type. The following indices can be used to estimate the network performance: U - the number of active (accepted) requests from the total number of received requests; t_w - average wait time of a request till the beginning of transmission; G - a bandwidth i.e. an average number of requests, accepted per one unit of time. It is not difficult to prove that the above characteristics may be expressed through one another. Investigations were carried out by methods of analytical modelling and simulation. In both cases it has been assumed that the addresses of requests destination are independent and uniformly distributed over network outputs. Network functioning at each input was considered as repetition of a sequence, consisting of three steps: wait time (t_w), time of transmission (t_{tr}) and the time of a next request generation (t_{gen}). Analysis was realized in a discrete time with t_{tr} assumed to be constant and equal to t_0 i.e. to the cycle of network operation while t_{gen} had geometrical distribution with the parameters t_0 , P_{in} , where P_{in} is a probability of request generation at an input at the beginning of a cycle. In this case $U = P_A$, where P_A is a probability of request acceptance within the network cycle, bandwidth is $G = U$ and a wait time is $t_w = (1/P_A - 1)t_0$.

Simulation was performed in continuous time with t_{tr} and t_{gen} distributed according to the truncated normal distribution.

Methods of determining P_A for multistage networks with a single path for each input-output pair and for the modes with blocked requests ignored are presented in [2].

It has been previously mentioned that this paper deals with a multipath network and in addition, it contains an analysis of the network operation involving the rejected requests resubmitting during the next cycle. Therefore, determination of P_A could not have been realized with the help of the known means.

In this paper the following general approach to calculating the networks with resubmitting of blocked requests is proposed. It is

based on reduction to the ignoring of blocked requests. This approach is applicable to a network with an arbitrary structure.

We denote: $P_A^W(p)$ - probability of a request acceptance at the network input, in a given cycle of the network operating in the mode with resubmitting of blocked requests and with a probability of a new request p ; $P_A^L(p)$ - probability of request acceptance in the mode of blocked requests ignoring; $P_E^W(p)$ - probability of request availability at the input in the beginning of the network cycle for the mode with resubmitting of blocked requests.

Then, considering the events, containing the requests at various inputs, independent we have

$$P_E^W(p) = P_E^W(p) P_A^L(P_E^W(p)) p + P_E^W(p) (1 - P_A^L(P_E^W(p))) + (1 - P_E^W(p)) p,$$

whence

$$p = \frac{P_A^L(P_E^W(p))}{1 - P_E^W(p)(1 - P_A^L(P_E^W(p)))} P_E^W(p),$$

i.e. $p = f(P_E^W(p))$, where the function f , which, as it can be easily seen is monotonic and determined by analysing the mode with ignoring the blocked requests. Denoting the inverse function f^{-1} we obtain

$$P_A^W(p) = P_A^L(f^{-1}(p)),$$

i.e. probability of request acceptance in the mode of blocked request resubmitting is expressed by the characteristics of the mode of blocked requests ignoring.

Computational method for multipath Omega network in the mode of blocked requests ignoring depends on the fact whether the number of stages is equal to two or more than two. For the two-stage network alternate paths (the total number of them B^2/N) can be united in one "pipe", so that movement of requests along different "pipes" goes on independently. It allows us to obtain the following computational formulae for probability of a request acceptance:

$$P_A^L(p) = (1 - (\sum_{i=0}^{B^2/N} P(i)(1 - 1/B)^i)^{N/B})/p,$$

where

$$P(i) = \sum_{j=1}^B \binom{B}{j} (1-p)^{B-j} p^j (B/N)^i (1-B/N)^{j-i}, \quad 0 \leq i \leq B^2/N - 1, \quad (1)$$

$$P(B^2/N) = 1 - \sum_{i=0}^{(B^2/N)-1} P(i)$$

For a network with the number of stages greater than two we propose the following approximate procedure. Consider an arbitrary group of R alternative outputs of a first stage switch. The probability $p(i)$, $0 \leq i \leq R$, of finding i requests at these R outputs is given in (1) with B^2/N replaced by R . Our approximation consists in regarding these outputs as independent with a probability p_1 of a request availability. This probability is chosen so as to preserve the mean number of requests at the group of R outputs:

$$p_1 = \frac{1}{R} \sum_{i=1}^R ip(i).$$

So we are given the probability of request appearance at an input of a second stage switch under the assumption that these events are independent for different inputs. Now the probability of request appearance at an output of the network is calculated in the same manner as for a single-path network considering the second stage as an initial one.

Computations, involving the application of the developed analytical model and simulation were realized for the case of a two-stage 256×256 modified Omega network, using 32×32 switches. Such a choice was caused by a sufficiently great dimension of the network and a modular 32×32 switch is, at present technologically realizable with its acceptable cost and sizes. Availability of 4 paths for each input-output pair allows us to estimate entirely the particularities of the multipath network in the considered operation mode. For comparison implemented under the same conditions, a crossbar and 1-path Omega network of the same dimension were investigated. Fig. 2 shows performance U versus input mean request rate ρ for interconnection networks mentioned above. First of all we should note that the difference between the results of analysis (solid lines) and the

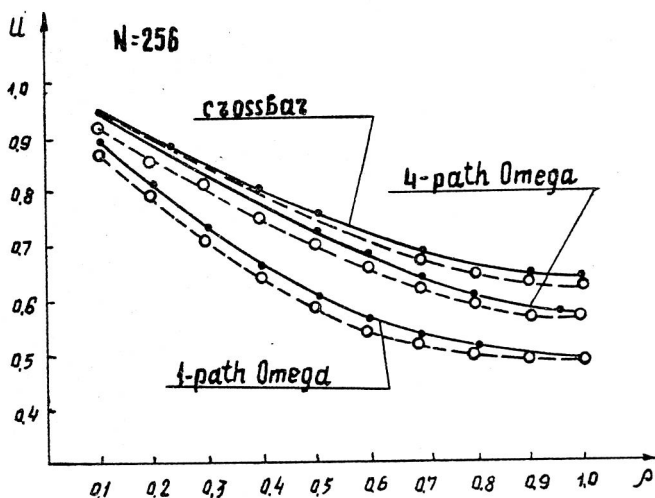


Fig. 2. Performance versus input mean request rate.

results of simulation (dashed lines) is very small. From the graphs in Fig. 2 it becomes clear, that performance of the 4-path network is noticeably higher than the one of the 1-path network and it approaches a crossbar performance; it is closely linked with the decreasing of number of conflicts in interconnections due to redundant paths. Computations also proved that consideration of rejected requests resubmitting during the next cycle affects the obtained results within the limits not exceeding seven percent.

Influence of various numbers of failed paths on the modified Omega network performance has been as well investigated by means of simulation (in such a network the paths failures can be caused by partial faults of large modular switches and by faulty links).

It has been found that occurrence of up to three arbitrary faults in the two-stage 4-path 256×256 modified Omega network does not practically affect its performance (see Table 1). This effect is available particularly due to the fact that because of the conflicts connected with destination addresses, some paths in the network turn to be vacant and probability is high, that these vacant paths are alternative with respect to those where faults occur.

TABLE 1

Dependence of performance U of the 4-path 256×256 modified Omega network on the number of faults r

r	0	1	2	3
U	0,576	0,572	0,568	0,562

Thus, in this particular case we deal with fault-tolerance in the real sense of this word, i.e. with no significant reduction of performance.

References

1. Padmanabhan K. and Lawrie D.H. A class of redundant path multi-stage interconnection networks. - IEEE Trans. Comput., 1983, vol.C-23, No.12, p.1099-1108.
2. Patel J.H. Performance of processor-memory interconnections for multiprocessors. - IEEE Trans. Comput., 1981, vol.C-30, No.10, p.771-780.

Józef WOŹNIAK *

protokoły komunikacyjne
radiowe sieci teleinformatyczne
transmisja wieloetapowa

PROTOKOŁY DOSTĘPU DO KANAŁU W RADIOWYCH SIECIACH TELEINFORMATYCZNYCH Z TRANSMISJĄ WIELOETAPOWĄ

W pracy przedstawiono kierunki prac badawczych dotyczących protokołów dostępu do kanału w rozległych radiowych sieciach teleinformatycznych. Zaprezentowano przykłady protokołów znajdujących zastosowanie w sieciach naziemnych z transmisją wieloetapową oraz sieciach satelitarnych. Wskazano na duże zainteresowanie protokołami rywalizacji w implementacjach sieciowych o różnym przeznaczeniu.

1. WSTĘP

Sieci radiowe wykazują specyficzne cechy wynikające głównie z własności stosowanego kanału. Stanowią one alternatywę dla istniejących już sieci przewodowych typu punkt-punkt. W wielu zastosowaniach nie jest bowiem możliwe lub ekonomicznie uzasadnione tworzenie sieci z wykorzystaniem tradycyjnych mediów transmisyjnych. Rozsiewczy charakter transmisji w kanale radiowym sprawia, że radiowe sieci teleinformatyczne wykazują szereg pozytywnych własności jak: łatwość rozbudowy, duża niezawodność, stosunkowo niski koszt tworzenia sieci, czy też możliwości komunikacji z użytkownikami ruchomymi. Wśród ich zasadniczych wad należy wymienić: stosunkowo duże rozpraszanie energii, wysoki poziom zakłóceń zewnętrznych, łatwość dostępu nieautoryzowanego oraz możliwość celowego zakłócania transmisji. W sytuacji gdy pewne zasoby są użytkowane przez dużą liczbę niezależnych użytkowników pojawia się każdorazowo potrzeba zwielokrotniania dostępu. Historycznie najstarszym sposobem wykorzystania pasma kanału jest oczywiście jego

* Politechnika Gdańska, Instytut Telekomunikacji, Gdańsk

podział na podkanały częstotliwościowe/ FDMA/. Mankamenty tej metody uwidaczniają się jednakże szczególnie jaskrawo, gdy konieczne staje się zapewnienie wzajemnej łączności pomiędzy dużą liczbą użytkowników.

Dualność i wymiennność czasu i częstotliwości sprawia, że zamiast podziału dostępnego pasma, ekwiwalentnym staje się podział czasu dostępu do całego pasma /TDMA/. Oferowana przez TDMA elastyczność i różnorodność metod dostępu do kanału sprawia, że w dalszym ciągu obserwuje się dużą koncentrację prac badawczych wokół algorytmów pracy systemów bazujących na podziale dostępu.

2. KIERUNKI ROZWOJU SIECI RADIOWYCH Z TRANSMISJĄ WIELOETAPOWĄ

Mówiąc o kierunkach rozwoju radiowych sieci teleinformatycznych mamy na uwadze zarówno tendencje ujawniające się w pracach badawczych jak też konkretnych ich implementacjach. Zagadnieniom klasyfikacji i analizy efektywności reguł dostępu do kanału, stosowanych w sieciach radiowych poświęconych zostało szereg prac przeglądowych/ np.[1],[2],[3],[4]/. Rozważając w dalszej części artykułu tendencje w rozwoju sieci radiowych będziemy prezentowali wybrane protokoły dostępu do kanału przydatne w sieciach rozległych. Dokonamy przy tym podziału tych sieci na 2 klasy: (i)-sieci naziemne z transmisją wieloetapową /typu MAN oraz WAN/; (ii)-sieci satelitarne.

2.1. Sieci naziemne z transmisją wieloetapową

Prowadzone aktualnie prace badawcze stanowią w znacznym stopniu rozwój idei zainicjowanych na początku lat siedemdziesiątych, poprzez uruchomienie sieci ALOHA [5]. Dotyczą one rozległych radiowych sieci teleinformatycznych z transmisją wieloetapową /Packet Radio Networks/. W sieciach tych pakiety przebywają trasę pomiędzy węzłem źródłowym a docelowym w kilku etapach. Są one podobnie jak w tradycyjnych, tj. przewodowych sieciach komputerowych /np. ARPA/ buforowane w węzłach pośrednich. Z uwagi jednakże na rozświeczony charakter transmisji w kanale radiowym rośnie skala trudności przy analizie pracy sieci. Szczególnej wagi nabierają wówczas procedury wyboru trasy oraz sterowanie przepływem pakietów. Uwidaczniają się też różnice pomiędzy protokołami dostępu do kanału stosowanymi w sieciach z transmisją jedno- i wieloetapową.

W przypadku sieci scentralizowanych o strukturach drzewiastych w analizach teoretycznych dominują protokoły typu ALOHA /głównie protokół synchroniczny tzw. S-ALOHA/ oraz CSMA[6],[7],[8].

Znacznie większą, różnorodność klas reguł dostępu obserwujemy w sieciach o strukturach rozproszonych. Również i w tych sieciach badana jest efektywność protokołu S-ALOHA [9], [10], świadcząca o dużej jego atrakcyjności podyktowanej prostotą funkcjonowania i łatwością implementacji. Pojawiają się również protokoły hybrydowe, jak przykładowo koncepcja połączenia S-ALOHA i CSMA [11]. W przypadku sieci z transmisją wieloetapową z uwagi na możliwość istnienia dużych populacji ukrytych stacji przydatność czystego protokołu typu CSMA staje się ograniczona. Pojawiają się też zupełnie nowe koncepcje protokołów z rezerwacją dostępu do kanału. Przykładem tego typu reguły dostępu jest protokół SUPERBRAM [12]. Zgodnie z nim w sieci przesyłane są dwa typy pakietów: pakiety rezerwacyjne oraz informacyjne. Pakiety pierwszego typu rywalizują o rezerwację prawa dostępu do kanału w tzw. okresie transmisji. Protokół zapewnia poszczególnym stacjom sprawiedliwy dostęp do kanału, a proces synchronizacji pracy stacji jest całkowicie zdecentralizowany /brak jest również ramek czasowych/.

W sieci z transmisją wieloetapową wybór określonego protokołu jest uzależniony zarówno od topologii sieci jak też od struktury generowanego ruchu. W przypadku wyboru typu TDMA ze sztywnym przydziałem ramek zasada przyporządkowania poszczególnych ramek stacjom musi uwzględniać postulat bezkolizyjnej transmisji. Znanych jest szereg algorytmów pozwalających na optymalizację tego procesu /np. [13], [14]/.

Wśród naziemnych sieci rozległych można również wydzielić sieci łączności dla obiektów ruchomych. Zasadnicza różnica w stosunku do sieci stacjonarnych wiąże się z mobilnością stacji i koniecznością dynamicznego uwzględniania zmian ich położenia w algorytmach dostępu do kanału. W przypadku sieci dla obiektów ruchomych można wykorzystać modele typowe dla radiotelefonicznych sieci komórkowych [15], względnie realizować transmisję pakietów w sposób zbliżony do stosowanego w sieciach teleinformatycznych.

2.2 Sieci satelitarne

Inny kierunek rozwoju sieci radiowych wiąże się z wykorzystaniem transponderów satelitarnych /nisko- lub wysokoorbitowych tzn. geostacjonarnych/. Ich użycie pozwala na tworzenie sieci o charakterze globalnym - jak to ma miejsce w systemie INTELSAT - oraz realizację szerokiego zestawu usług [16]. Wykorzystanie do transmisji pasm częstotliwości 4/6 GHz /tzw. pasmo C/ oraz 12/14 GHz /pasmo Ku/ i wyższych umożliwia tworzenie kanałów o bardzo dużych szerokościach /rzędu po 500 MHz dla częstotliwości 12/14GHz/.

Kanały satelitarne mogą być wykorzystane zgodnie z techniką FDMA lub TDMA. Możliwe jest też stosowanie różnorodnych protokołów dostępu w zależności od charakterystyk generowanego ruchu i wymagań stawianych przez użytkowników. Zastosowania wymagające małych opóźnień wiążą się z użyciem technik rywalizacji; przy jednoczesnym założeniu niewielkich obciążeń kanału. Duża efektywność wykorzystania kanału wiąże się z koniecznością stosowania reguł przydziału pasma kanału na żądanie. Uwzględniając znaczne opóźnienia w łączności z satelitami geostacjonarnymi /rzędu 0.25 s/ pociąga to za sobą duże opóźnienia w transmisji pakietów. Z uwagi na ten fakt korzystne rozwiązanie mogą stanowić techniki mieszane, tj. z rezerwacją pasma kanału w przypadku wystąpienia kolizji /np. protokół SRUC [17], [18]/.

Rozwijane koncepcje satelitarnych sieci teleinformatycznych wiążą się zwykle z wykorzystaniem pojedynczego transpondera. W pracy [19] prezentowana jest idea dynamicznego przełączania anteny nadawczej transpondera w celu uzyskania większej koncentracji wysyłanej wiązki. Kanał do satelity wykorzystywany jest w tej propozycji zgodnie z algorytmem S-ALOHA, podczas gdy kanał z satelity jest bezkolizyjny.

W literaturze przedstawiane są również propozycje sieci wykorzystujących zarówno transpondery geostacjonarne jak i niskoorbitowe. Pozwala to na realizację transmisji wieloetapowych bez wykorzystania retransmisyjnych stacji naziemnych. Taka dwustopniowa struktura proponowana jest w pracy [20]. Dostęp stacji naziemnych do kanału transpondera niskoorbitowego realizowany jest przy zastosowaniu kodowych metod zwielokrotniania w połączeniu z protokołem ALOHA. Protokół ALOHA stosowany jest także do rezerwacji kanału pomiędzy transponderem niskoorbitowym a geostacjonarnym.

Gwiazdzysta struktura sieci satelitarnych /z pojedynczym transponderem/ sprzyja rozwojowi tzw. sieci dedykowanych, w których szereg małych stacji komunikuje się ze stacją centralną. W ramach wyposażenia końcowego stacji instalowane są anteny o wymiarach 1-1.8 m /tzw. very small aperture terminals/. Stacja główna wykorzystuje zwykle antenę o średnicy ok. 5m. Wśród protokołów stosowanych w funkcjonujących sieciach satelitarnych dominują techniki sztywnego lub dynamicznego przydziału pasma kanału zgodnie z TDMA /np. INTELSAT/ względnie dostępu niekontrolowanego typu ALOHA. Asynchroniczny protokół ALOHA znajduje zastosowanie w jednym z kanałów sieci MARISAT [5]. Jest on również wykorzystany w systemie EUTELSAT do transmisji danych z platform wieżniczych [21]. Inny przykład zastosowania tego protokołu stanowi sieć PACNET [5].

3. PODSUMOWANIE

W artykule dokonano krótkiego przeglądu reguł dostępu znajdujących zastosowanie w rozległych teleinformatycznych sieciach radiowych. Przedstawiono wybrane wyniki prac badawczych dotyczących protokołów dostępu, jak też przykłady funkcjonujących sieci. Należy przy tym zwrócić uwagę na fakt dużej przydatności prostych protokołów rywalizacji typu ALOHA. Znajdują one zastosowanie we wszystkich strukturach sieciowych i aplikacjach. Jednocześnie należy podkreślić, że protokoły te zapewniają niską sprawność wykorzystania kanałów. Prowadzone prace badawcze mają na celu opracowanie algorytmów nowych i efektywnych, a jednocześnie stosunkowo łatwych w implementacji.

4. LITERATURA

- [1] Tobagi F.: Multiaccess Protocols in Packet Communication Systems. IEEE Trans. Commun. Vol.28, 1980, s.468-488.
- [2] Seidler J.: Analiza i Synteza Sieci Teleinformatycznych. PWN 1979.
- [3] Kozłowski A., Wozniak J.: Radiowe Sieci Teleinformatyczne. W: Problemy Teleinformatyki. red. Sobczak W., WKiŁ 1984.
- [4] Pach A.: Wielodostępowy Kanał we Współczesnych Sieciach Teleinformatycznych. Przegląd Telekom., Nr6, 1981, s.182-184.
- [5] Abramson N.: Development of the ALOHANET. IEEE Trans. Inform. Theory, Vol.31, 1985, s.119-123.
- [6] Tobagi F.: Analysis of a Two-Hop Centralized Packet Radio Network- Part I: Slotted ALOHA. IEEE Trans. Commun. Vol.28, 1980, s.196-207.
- [7] Tobagi F.: Analysis of a Two-Hop Centralized Packet Radio Network Part II: CSMA. IEEE Trans. Commun. Vol.28, s.208-216.
- [8] Fratta L., Wozniak J.: Performance Analysis of a Two-Hop Star Network with S-ALOHA Multiple Access Scheme. Proc. ITC12, 1988, s.5.3B.6.1-9.
- [9] Silvester J., Kleinrock L.: On the Capacity of Multihop Slotted ALOHA Networks with Regular Structure. IEEE Trans. Commun. Vol.31, 1983, s.974-982.
- [10] Takagi H., Kleinrock L.: Throughput-Delay Characteristics of Some Slotted ALOHA Multihop Packet Radio Networks. IEEE Trans. Commun. Vol.33, 1985, s.1200-1207.
- [11] Nelson R., Kleinrock L.: Rude CSMA: A Multihop Channel Access Protocol. IEEE Trans. Commun. Vol.33, 1985, s.785-791.
- [12] Chlamtac I., Franta W.: A Multiaccess Protocol for Multihop Radio Networks. IEEE Trans. Commun. Vol.33, 1985, s.1067-1075.
- [13] Cidon I., Sidi M.: Distributed Assignment Algorithms for Multi-Hop Packet Radio Networks. Proc. IEEE INFOCOM 1988, s.1110-1118.
- [14] Ephremides A., Truong T.: Distributed Algorithm for Efficient and Interference-Free Broadcasting in Radio Networks. Proc. IEEE INFOCOM 1988, s.1119-1124.
- [15] Ben D.: Radiotelefoniczne Systemy Komórkowe. KST 1987, Tom A, s.37-63.
- [16] Colino R.: An Inside Look at the New INTELSAT. Telephony, Nr22, 1986, s.50-52.
- [17] Tanaka S., Ishida K.: The SPUC Protocol for Sattelite Packet Communication -A Performance Analysis. IEEE Trans. Commun. Vol.34, 1986, s.937-945.

- [18] Borgonovo F., Fratta L.: SRUC - A Technique for Packet Transmission on Multiple Access Channels. Proc. ICC 1978, s.601-607.
- [19] Wu Czeng-Shong, Li V.: Random Access for a Multibeam Satellite with Dynamic Transponder Switching. Proc. IEEE INFOCOM 1988, s.1022-1031.
- [20] Sharifi M., Arozullah M.: A Multiple Access Technique for Centralized Multiple Satellite Networking with On-Board Processing in the Central Node. Proc. IEEE INFOCOM 1988, s.1012-1021.
- [21] Crespo E., Nelson A.: Digital Satellite Services and Applications via the EUTELSAT Satellite Multiservices System. Proc. ICDSC-7, 1986, s.759-764.

CHANNEL ACCESS PROTOCOLS FOR MULTIHOP PACKET RADIO NETWORKS

In the paper the main research activities on channel access schemes for multihop packet radio networks are discussed. Examples of channel access protocols used in satellite and terrestrial networks are presented. In addition the usefulness of contention protocols in different network applications and environments is shown.

Протоколы доступа к каналам в дистанционных
радиосетях телеобработки

В работе сделано короткий обзор протоколов доступа к каналам в радиосетях телеобработки. Рассмотрены главные направления научно-исследовательских работ. Даны примеры действующих сетей. Обращено внимание на большой интерес протоколами случайного доступа.

Spis rzeczy

Profesor Jerzy Bromirski (1915-1989)	3
J.van Arkel,	
Sieci komputerowe a systemy ewidencji ludności	5
J.Barczyński,	
Modularna weryfikacja protokołów	10
J.Barczyński,	
Analiza semantyczna specyfikacji EXAMPLE	15
E.Bilski,	
Architektura wyższych warstw modelu odniesienia OSI/ISO	20
D.Borowska, W.Borowski, R.Filipek, A.Gospodarowicz, Z.Łękawa, S.Zając,	
Ewidencja i rozliczanie użytkowników systemu SKOT w sieci KASK	33
A.Brodnik, M.Spiegel, B.Korousic,	
Tania przemysłowa sieć komputerowa do zastosowań w czasie rzeczywistym	37
U.Buhrandt, J.Hinkelmann,	
Projekt systemu testowania zgodności "TEKOS"	46
O.Buzek,	
Uzyskiwanie współdziałania przez konwersję protokołów	51
J.G.Cabański, M.Lewandowski, S.Marach, Z.Oczki,	
ASTER - Autonomiczny sterownik transmisji szeregowych dla PC XT/AT	55
J.G.Cabański, M.Lewandowski, S.Marach, Z.Oczki,	
MasterNET - Gwiazdzista sieć lokalna PC XT/AT	58
W.Cellary,	
Kilka uwag o badaniach prowadzonych w dziedzinie rozproszonych baz danych	62
A.Dubielewicz, J.Ratajczak,	
Sterowanie wykonaniem kompletu testów w systemie testowania zgodności protokołów w sieci KASK	71
I.Dubielewicz, W.Komorowski,	
Realizacja systemu testowania zgodności w sieci komputerowej KASK	76
I.Dubielewicz, K.Koleśnik,	
Język TTCN-MP w systemie testowania zgodności sieci KASK	81
J.Dudziak, M.Wnuk,	
Mikroprocesorowe sterowniki transmisji danych dla m.c. Odra 1300	86

A.Fierla

Język logiki modalnej dla systemów komunikujących się	90
M.Flasiński, L.Kotulski,	
O językowym podejściu do sterowania alokacją oprogramowania rozproszonego	97
J.Gruber,	
Sieć DECnet komputerów VAX firmy Digital Equipment Corp.	102
J.Gruber, L.Z.Misiura,	
Badania przydatności lokalnej sieci mikrokomputerowej 10-NET w systemie rozproszonego przetwarzania danych KOMPAN	110
E.Hudyma, E.Kosmulska-Bochenek,	
Zarządzanie warstwą komunikacyjną sieci komputerowej	117
Z.Hulicki,	
Krakowska Akademicka Sieć Komputerowa - koncepcja i założenia projektowe	122
A.Huzar, A.Kaliś,	
Problemy synchronizacji w stacji transferu zbiorów sieci komputerowej KASK	126
Z.Huzar, Z.Sprąwski,	
Tworzenie oprogramowania sieciowego - problemy metodologiczne i teoretyczne	132
W.Irzeński,	
Warstwa sieciowa rozproszonego systemu dydaktycznego	149
R.Kaszuba, J.Kwiatkowski,	
Szeregowanie komunikatów do transferu	154
P.Kociatkiewicz,	
Stan obecny i rozwój systemów teleprzetwarzania danych w Instytucie Komputerowych Systemów Automatyki i Pomiarów	159
K.Koczur, L.Kotulski,	
Optymalizacja dystrybucji oprogramowania w lokalnych sieciach komputerowych	174
J.Konorski,	
Przepustowość sieci pakietowej z działającym mechanizmem unikania zakleszczeń	178
H.König, O. Krohmann,	
Pewne podejście do automatycznej implementacji protokołów	182
P.Kremienowski,	
Funkcje zarządzania w sieciach komputerowych typu SNA	189
A.Kusiuk,	
Transmisja sygnałów komputerowych w cyfrowych kanałach telefonicznych systemu PCM	194

M.Kwiatkowski,	
Usługi lokalnej sieci komputerowej HADES	198
N.Linge,	
Połączenia lokalnych sieci komputerowych	203
Y.Lyamin, V.Yalovetsky,	
Narzędzia rozproszonego przetwarzania danych w sieciach komputerowych	215
K.Odolak, A.Pękalski,	
Kierunki rozwoju technicznego publicznej sieci teleinformatycznej w Polsce do roku 1995	219
M.Siebert, K.Nolte, U.Hostmann,	
Empiryczne modele predykcji systemu terminalowego REG	224
W.Staniszkis,	
Przegląd technik zarządzania bazami danych	232
J.Stokłosa,	
O standardach kryptograficznych ISO dla sieci komputerowych	237
E.Sutherland,	
Rozwój systemu X.400 w Zjednoczonym Królestwie	241
P.Siroky,	
Transformacja aktualizacji w rozproszonych bazach danych z powielonymi danymi	245
G.G.Veselovski, A.J.Zaloznev, A.A.Puchalski,	
Analiza zachowania zmodyfikowanej sieci Omega	250
J.Woźniak,	
Protokoły dostępu do kanału w radiowych sieciach teleinformatycznych z transmisją wieloetapową	255

Contents

Prof. Jerzy Bromirski (1919-1989)	3
J. van Arkel, Networks and population information systems	5
J. Barczyński, Modular verification of protocols	10
J. Barczyński, Semantical analysis of EXAMPLE specification	15
E. Bilski, Upper layers architecture of the RM OSI	20
D. Borowska, W. Borowski, R. Filipek, A. Gospodarowicz, Z. Łęka, S. Zając, Registry and account for users of the SKOT system from KASK network	33
A. Brodник, M. Spegel, B. Korousic, A low-cost industrial network for real-time applications	37
U. Buhrandt, J. Hinkelmann, The design of a conformance test system "TEKOS"	46
O. Buzek, Achieving interoperability via protocol conversion	51
J. G. Cabański, M. Lewandowski, S. Marach, Z. Oczki, ASTER - Autonomous serial transmission controller for PC XT/AT	55
J. G. Cabański, M. Lewandowski, S. Marach, Z. Oczki, MasterNET - Local network for IBM PC XT/AT	58
W. Cellary, Some thoughts on distributed database research	62
A. Dubielewicz, J. Ratajczak, The test suite control in the KASK conformance testing system	71
I. Dubielewicz, W. Komorowski, An implementation of the conformance testing system	76
I. Dubielewicz, K. Kolesnik, TTCN implementation in the conformance testing system for the KASK network	81
J. Dudziak, M. Wnuk, Microprocessor based data transmission controllers for the Qdra computer	86

A.Fierla

Modal logic language for communicating systems	90
M.Flasiński, L.Kotulski,	
On the linguistic approach to controlling a distributed software allocation	97
J.Gruber,	
VAX networking and communication software - general overview	102
J.Gruber, L.Z.Misiura,	
Studies of functional properties of the 10-NET microcomputer local area network in KOMPAN system	110
E.Hudyma, E.Kosmulska-Bochenek,	
Layer management of a computer network	117
Z.Hulicki,	
Data communication network for the universities in Cracow	122
A.Huzar, A.Kaliś,	
Some synchronization problems in file transfer station of computer network KASK	126
Z.Huzar, Z.Spławski,	
Network software development - methodological and theoretical problems	132
W.Irzeński,	
Network layer of the distributed education system	149
R.Kaszuba, J.Kwiatkowski,	
Message transfer scheduling	154
P.Kociatkiewicz,	
Actual state and development trends of teleprocessing system designed in the Institute of Computer Systems for Automation and Measurement	159
K.Koczur, L.Kotulski,	
Optimization of the software distribution in local computer networks	174
J.Konorski,	
Network throughput under deadlock prevention schemes	178
H.König, O.Krohmann,	
An approach to automated protocol implementation	182
P.Kremienowski,	
Management functions in the SNA type computer networks	189
A.Kusiuk,	
Transmission computer signals in the digital telephone channels	194

M.Kwiatkowski,	
User services of HADES local area network	198
N.Linge,	
The interconnection of local area networks	203
Y.Lyamin, V.Valovetsky,	
A tools for distributed data processing for computer nets	215
K.Odolak, A.Pękaliski,	
Technology trends in public data network area in Poland up to 1995	219
M.Siebert, K.Nolte, U.Hostmann,	
Empiric prediction models of the REG-system	224
W.Staniszkis,	
An overview of the multidatabase management technology	232
J.Stokłosa,	
On the ISO cryptographic standards for computer networks	237
E.Sutherland,	
X.400 developments in the United Kingdom	241
F.Široky,	
Transformation of updates in distributed data bases with replicated data	245
G.G.Veselovsky, A.Yu.Zalozhnev, A.A.Pukhalsky,	
Performance and fault-tolerance analysis for modified Omega network	250
J.Woźniak,	
Radio channel access protocols	255

PRACE NAUKOWE CENTRUM OBLICZENIOWEGO
(wydane w latach 1980—1989)

Nr 1, Studia i Materiały nr 1, <i>Zagadnienia oprogramowania systemów komputerowego wspomagania dydaktyki</i> , Wrocław 1980	20,—
Nr 2, Studia i Materiały nr 2, <i>Zagadnienia oprogramowania węzła pod-sieci komunikacyjnej sieci komputerowej MSK</i> , Wrocław 1983	180,—
Nr 3, Konferencje nr 1, <i>Sieci komputerowe. Usługi, protokoły, modele. Część 1</i> , Wrocław 1985	160,—
Nr 4, Konferencje nr 2, <i>Sieci komputerowe. Usługi, protokoły, modele. Część 2</i> , Wrocław 1986	250,—
Nr 5, Konferencje nr 3, <i>Sieci komputerowe. Teoria, technika, zastosowania</i> , Wrocław 1987	335,—
Nr 8, Monografie nr 1, Huzar Z., <i>Programowanie procesów komunikujących się w czasie rzeczywistym</i> , Wrocław 1989	300,—



Cena zł 1000,—

Subscription should be sent (at any time of the year) to:

„Ars Polona”

Krakowskie Przedmieście 7, 00-068 Warszawa

Bank account number: PBK XIII Oddz. W-wa 370044-1195-139-11

Zamówienia na prenumeratę można składać:

OR PAN, PKiN, 00-901 Warszawa

Nr konta bankowego: PBK IX Oddz. W-wa, 370031-4792

Wydawnictwa Politechniki Wrocławskiej
ma stale na składzie Księgarnia Wr 49
Wybrzeże Wyspiańskiego 27, 50-370 Wrocław
oraz Wojewódzka Księgarnia Techniczna
ul. Świdnicka 8, 50-067 Wrocław

ISSN 0860-1615