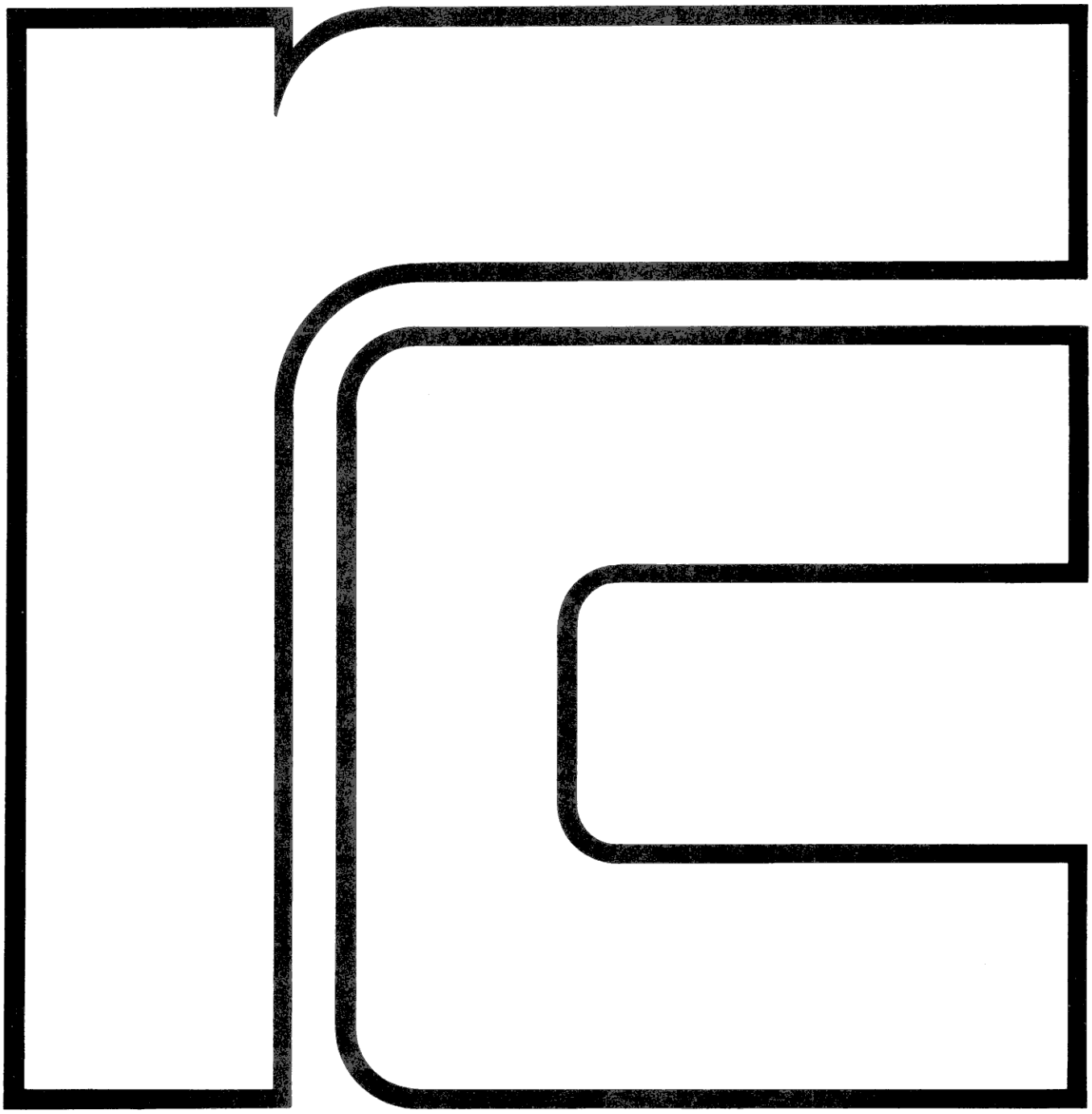
GENERALIZED TERMINAL

ISO-OSI Architecture



ONLINE 81

RCSL No: 43-GL 11021

Edition: January 1981

Author: Tom Jacobsen

Title:

The ISO-OSI Model Applied to Microprocessor and Minicomputer Systems. An ONLINE'81 contribution.

A/S REGNECENTRALEN
af 1979

Keywords:

Generalized Terminal, ISO-OSI, X.29, X.25, X.22, X.21, ECMA Transport Station, Teletex, s.h, s.f, s.d EURONET DEVT, DTE, RCNET, Virtual Machine, RC850, RC700, RC3600, Pascal 80, Compiler Architecture

Abstract:

This paper will be presented at the ONLINE'81 conference, held at Düsseldorf, 10-14th February 1981.

The generalized Terminal is presented. The ISO Reference Model for Open Systems Interconnection is utilized as a template for the internal software architecture. The ideas of a virtual machine and the derived requirements for a compiler architecture are dealt with.

Printed by A/S Regnecentralen af 1979, Copenhagen

RCSL 42-i1592

January 1981

Tom Jacobsen, RC Computer

THE ISO-OSI MODEL APPLIED TO MICROPROCESSOR AND MINICOMPUTER SYSTEMS

Abstract

RC COMPUTER A/S has defined its computer for the future. It is defined in terms of a virtual machine. This virtual machine is determined by a high level systems programming language, and by the interface to the kernel of the operating system.

One of the products based on this combination, is the Generalized Terminal. It utilizes, as a template, the ISO Reference Model for Open Systems Interconnection. The ISO model is used as an architectural outline for the internal software structure. Furthermore it indentifies the interface between the session layer and the transport layer as being subject to standardization.

Systems Architecture

RC COMPUTER is currently developing a generalized terminal. The terminal is based on the ISO Reference Model for Open Systems Interconnection and utilizes the ISO model as a template for the software architecture. Furthermore, it identifies the interface situated between the transport layer and the session layer, as being subject to company standardization.

The ISO model for interconnection consists of seven layers, the uppermost layer being the application layer, the bottom layer being the physical layer and the in-between layers being used for presentation control, session control, transport control, network control, and link control. The transport control layer is situated right in the middle providing the upper layers with network independent transport service. It is itself based on network dependent service. Thus the transport control layer transforms the network dependent service, offered by the actual network interfaces, into network independent transport services, as required by the network independent applications.

To illustrate the importance of the functions of the transport control layer interface, the ISO model may be visualized as a double cone with the upper cone representing the different applications, and the lower cone describing the transport systems.

The transport systems may be based on many different systems, such as public networks, local networks, broadcast networks, microwaves, optical links and satellite links.

The applications may be simple terminals, refined terminals, compound systems for office automation, host computers and distributed systems.

In principle, all kinds of applications and all kinds of transport systems may be hosted by the model. Any application and any transport system may be identified as a path through their respective cones.

When the generalized terminal makes its debut, the network interfaces X.25 and X.21 will be utilized. For this purpose proper paths for communication have to be identified. Because the services offered by the X.25 and the X.21 are very different, the transport protocols for the transport control layer will either be different or at least they will be based on different functions for the same transport protocol. These functions will be known as varying sets of network dependent elements of procedures.

The transport interface will not only be dependent on the kind of service to be provided, but will, to an even larger extent, be shaped by the actual systems programming language.

RC COMPUTER has defined Pascal 80 as the high level systems programming language for future products. The utilization of a high level systems programming language plus the kernel interface, is the essence of a virtual machine. Consequently, the software, including the system software can be developed and maintained independently of the actual hardware targeting that virtual machine.

As a result of the above analysis, the following subjects must be examined further in order to describe the generalized terminal and the utilization of the ISO model:

- Network Interfaces
- Transport Protocols
- Presentation/Session Protocols

- Pascal 80
- Compiler Architecture
- Hardware Implementations

Network Interfaces

CCITT has specified two network dependent interfaces which reflect two major techniques for network construction. These interfaces are X.25 for packet switching, and X.21 for circuit switching. Furthermore, another interface for circuit switching, X.22, will soon be introduced.

The X.25 interface may be characterized as a statistical multiplexer which allows for individual flow control, while X.21 is a single server without preempting and without waiting line. X.22 is a time multiplexer. Neither X.21 nor X.22 offer flow control or link control.

The above mentioned CCITT specified interfaces interconnect a stand-alone terminal or a cluster or a local network to a public network. They may also be integrated with the network by implementing the network node as part of the layer-three network connection. Thus the generalized terminal may be integrated with RCNET, which is a datagram-based-network, constructed and manufactured by RC COMPUTER.

The CCITT X.25 line-server is non-preemptive and in most cases a single server. Fragmentation may allow for small waiting times for small amounts of data, and can facilitate parallel services. Thus the total network time will decrease. Fragmentation causes large quantities of data to be transmitted as a kind of pseudo background activity. The costs of fragmentation are seen in increased header

overhead and in an increased number of packets to be routed and handled by the network. Non-fragmentation may result in monopolization of line-servers and an average low utilization of the buffer resources.

No priority facilities are provided by X.25, but due to the statistical multiplexing some pseudo-priority classes may be introduced.

The CCITT X.21 and X.22 interfaces do not provide any priority facilities. Being just single servers without preempting and without waiting line, X.21/X.22 provide a rather primitive service which requires that all functions, other than network addressing, must be implemented at a higher level.

Transport Protocols

ECMA, the European Computer Manufactures Association, has forwarded a final draft for a standard ECMA Transport Protocol. The protocol has been developed in parallel with the CCITT Network Independent Basic Transport Service for Teletex, CCITT Draft Recommendation S.h. During the last stage of the development of the ECMA protocol, the CCITT Draft Recommendation was included as part of the ECMA protocol. This inclusion was possible because the ECMA proposal defines different classes of service, ranging from the most primitive, CCITT Teletex Class, to a sophisticated class, enabling concurrent utilization of different network services. The sophisticated ECMA class has much in common with the one originally proposed by IFIP (INWG No. 96.1).

The Teletex class is named Class Zero. Another class of interest is the ECMA Class Two. This class allows individual flow control, but

is without facilities for error detection and error correction.

The Generalized Terminal will at least utilize the CCITT specified interfaces X.25 and X.21.

Since no relevant ISO standards exist, and in order to ensure that minimal re-mapping will be necessary when the ISO standards become available, the most appropriate candidate for the transport protocol is one or more of the above mentioned ECMA classes. Class zero and class two are of special interest, primarily because class zero offers the most basic service and class two offers facilities for multiplexing.

Presentation/Session Protocols

Even though a great variety of protocols for virtual terminals have been proposed, no ISO standard or candidate for standardization exists at the present time. However, the most interesting possibilities are EURONET DEVT, CCITT X.29, and CCITT Teletex.

The EURONET DEVT and the CCITT X.29 protocols have in fact integrated the presentation/session layers with the transport layer, and they are on a large scale related to the X.25. Therefore, the most appropriate protocol for the generalized terminal is the CCITT Teletext. In combination with ECMA class zero, the CCITT Teletex presentation/session protocol will constitute a Teletex terminal which can be applied to X.25 and X.21 network interfaces.

Pascal 80

The software will be written in the high level systems programming language, Pascal 80. Since the virtual machine is defined by its association to the high level systems programming language, the actual hardware implementations may be changed. Consequently new computers may be introduced without affecting the software. Thus the software will become portable between existing and future elements of the new computer family of RC COMPUTER.

Furthermore the present project will also implement an extension for the operating system kernel. This will enable access from a high level language to modules implementing communication in accordance with the ISO-model of OSI.

The high level systems programming language can be described as an extension to sequential Pascal. It will offer features for dynamic creation and deletion of process incarnations and facilities for inter-process communication.

Concurrent process incarnations communicate by means of objects. These objects can be thought of as existing in an environment common to all process incarnations. A means of communication between two process incarnations can be established by giving them access, one at a time, to the same object.

Two basic types are provided for obtaining and exchanging access to objects. The types are called reference and queued-semaphore. The value of type reference is either a reference to an object or to a nil.

Process incarnations can exchange access to an object. This is achieved by means of queued-semaphore type variables. Queued-

semaphores were introduced by Søren Lauesen. They combine the synchronizing effect of Dijkstra's semaphores with the exchange of data. One can visualize them as "mailboxes" with unlimited capacity, through which all communication between process incarnations must be channelled.

A queued-semaphore is composed of a sequence (FIFO-queue) of objects and a set of temporarily stopped process incarnations. At any given time, either the sequence of objects or the set of process incarnations or both will be empty. These three states of a queued-semaphore variable are named: open, locked, passive. In the open state, the sequence of objects is not empty. In the locked state, the set of process incarnations is not empty. In the passive state the sequence of objects, as well as the set of process incarnations, are empty.

The objects may be one simple object or it may be a stack of simple objects. Objects are composed of a header and a data part. The stack may be a stack of headers with reference to the same data part, or it may be a stack the elements of which is a pair consisting of a header and its associated data part. In the first case, stack operations allow for header hiding and header substitution.

To facilitate inter-process communication between process incarnations residing in loosely coupled processors, the above mentioned queued-semaphores will be extended. This paves the way for mailbox based communication in a dynamic environment and introduces an implicit queued-semaphore reference. The queued semaphore extension combined with standardized elements of procedures constitutes the transport interface.

Compiler Architecture

The virtual machine will be applied to a variety of different hardware implementations. This will put some constraints on compiler development.

Each new compiler can of course, be developed from scratch and this may result in a highly optimized compiler. However, the development requires a large amount of human resources and also a lot of verification. Each executable program has to result in the same set of equivalent actions, independently of the actual compiler and the actual hardware implementation.

Another possibility is to divide the three distinct tasks. The first will be dependent upon the language only. The last will be dependent upon the targeting hardware only, and the inbetween will be dependent upon some language features and upon the class of targeting hardware.

It will be typical that the first part consists of a parser and the semantic analyzer, while the second part will select the run-time model and in particular perform the addressing compilations. The last part will generate the specific code to be executed or interpretated.

Intercommunication between the above mentioned parts is attained by utilizing intermediate languages. Therefore, the existence of intermediate languages is foreseen in the future. One of the intermediate languages will be dependent on the virtual machine only. The others will resemble the high level systems programming language, as well as the class of targeting hardware.

Hardware Implementations

In the case of Pascal 80, RC COMPUTER has adopted the concept of the above mentioned compiler architecture. So far two different targeting machines have been constructed. The first one is based on the Zilog Z80 which is an eight-bit microcomputer used in RC700 and RC850. The second is a minicomputer currently under construction, which is based on bit sliced components. This minicomputer has a specially developed interface which facilitates firm-ware execution of a Pascal 80 related intermediate code.

RC700 is an intelligent micro-based autonomous system offering COMAL and PASCAL computer power for the professional user. The minicomputer is developed especially for the construction of network nodes, terminal concentrators, gateways and protocol converters, and front-ends.

RC850 is being developed as an intelligent screen based computer. It facilitates network connection and applications in the area of Teletex and office automation.

For our existing RC3600 minicomputer, a π-level software interpreter has been developed. This interpreter functions as a normal task and is controlled by the original operating system. Thus Pascal 80 processes may communicate with non-Pascal 80 processes.

Epilogue

The architecture presented here aims particularly towards construction of communication systems.

For these systems, the ISO Reference Model of Open Systems Interconnection is used as an architectural template. This is done in order to ease the adaptation of new standards and in order to define an internal interface for transport service. This interface reflects the service offered by a network independent transport interface, as well as the characteristics of the virtual machine.

The virtual machine is defined by the high level systems programming language PASCAL 80. In order to facilitate implementation on differing hardware, a compiler structure consisting of three distinct tasks is being developed.

The high level systems programming language PASCAL 80 has been chosen because of its simplicity, and because of RC COMPUTER's already gained experience in the use of this special language. Quite likely, five to ten years from now, other languages for system programming will be standardized. In that case some reprogramming will be needed. Nevertheless this period is very long compared with the rapid change of hardware components.

Today RC850 is based on an eight-bit microcomputer, Z80, but in order to host a full generalized terminal, a sixteen-bit microcomputer will be incorporated. The virtual machine, however, remains the same.

Connection to at least two widespread network services, X.25 and X.21, is planned and others such as X.22 and RCNET may be implemented as well.

List of Figures

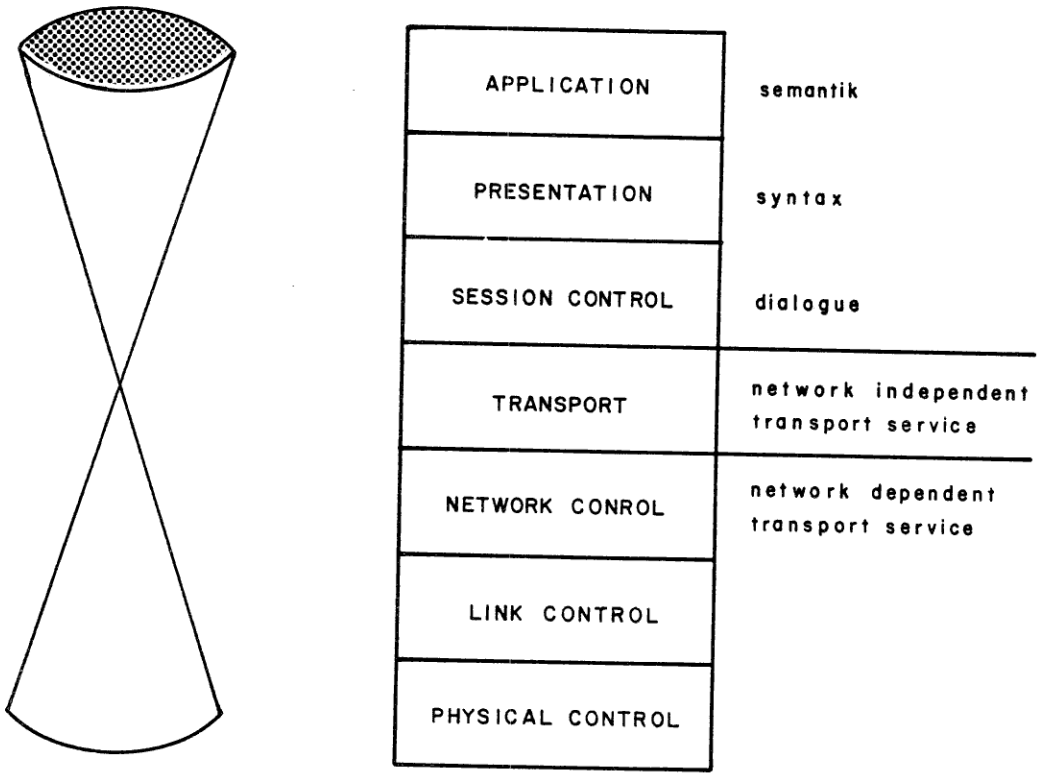
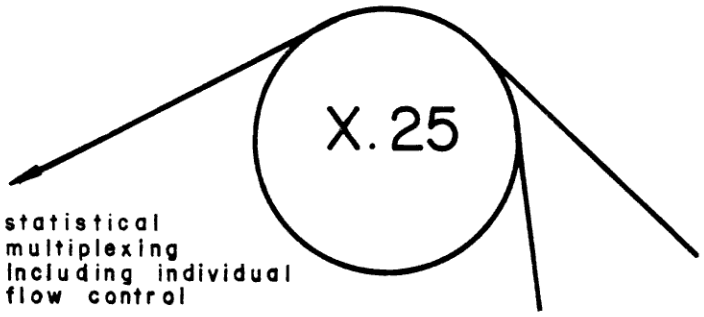
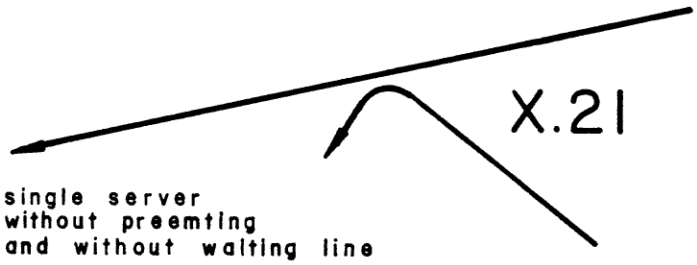


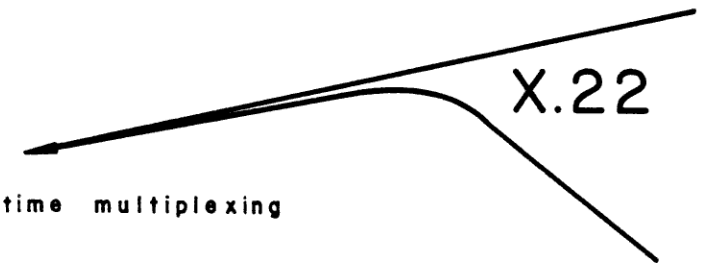
Figure 1 -
The ISO Reference Model of
Open Systems Interconnection



Packet Switched Interface



Circuit Switched Interface



Circuit Switched Interface

Figure 2 -
CCITT Network Interfaces

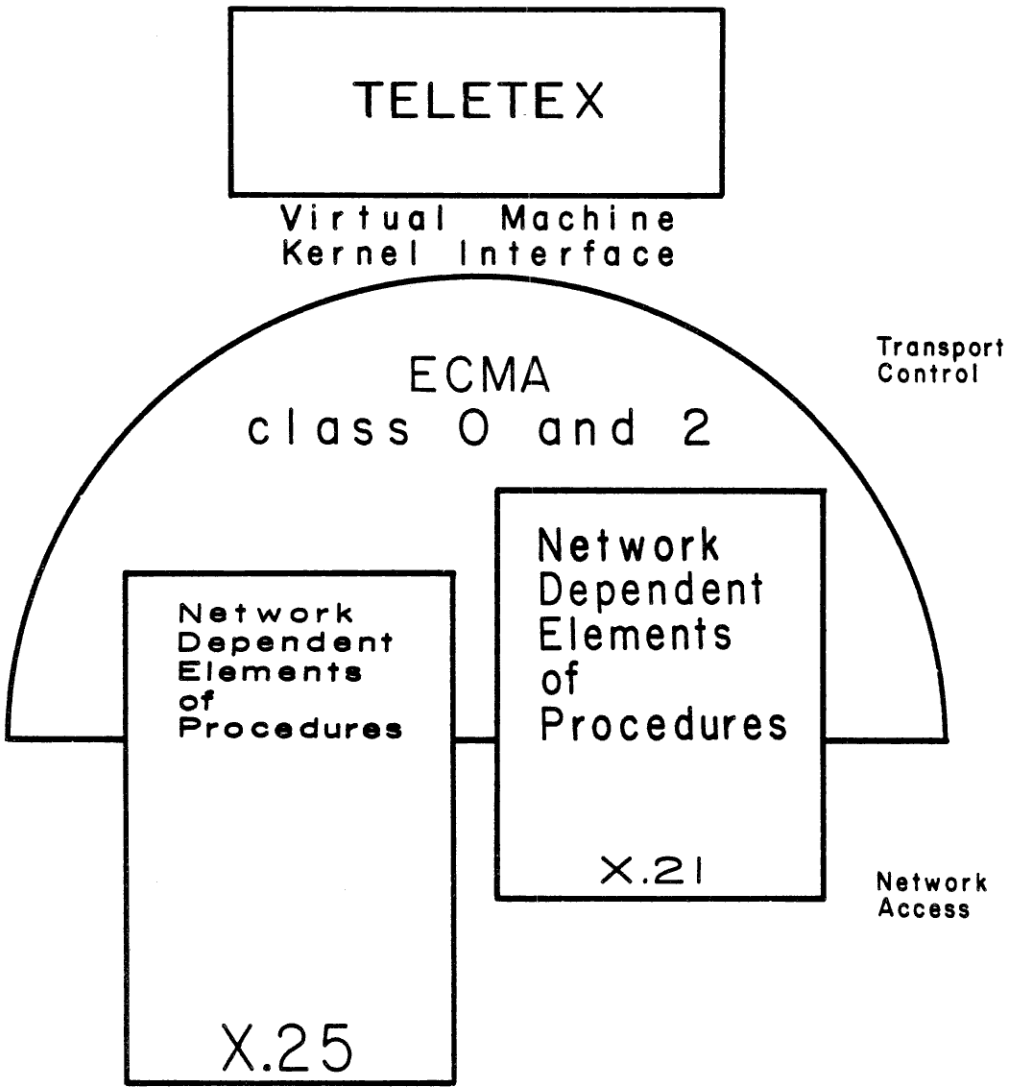


Figure 3 -
Virtual Terminal Architecture

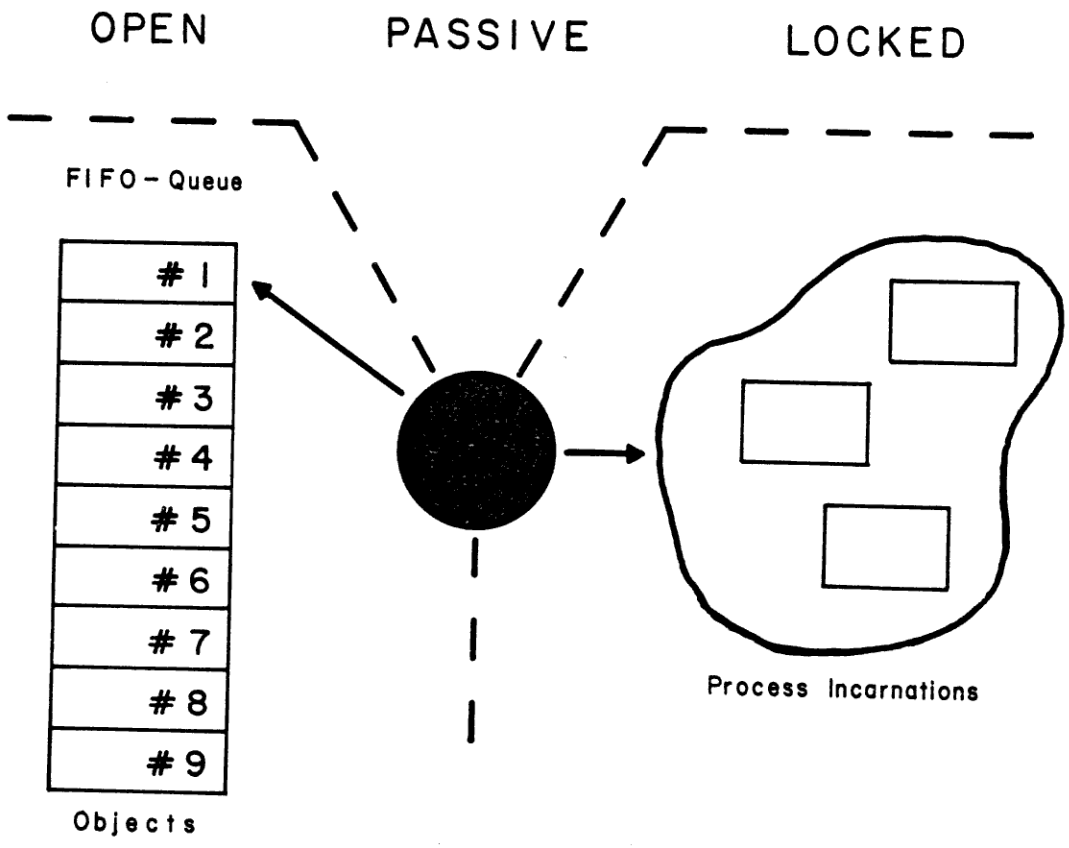


Figure 4 -
Pascal 80 - Queued Semaphores

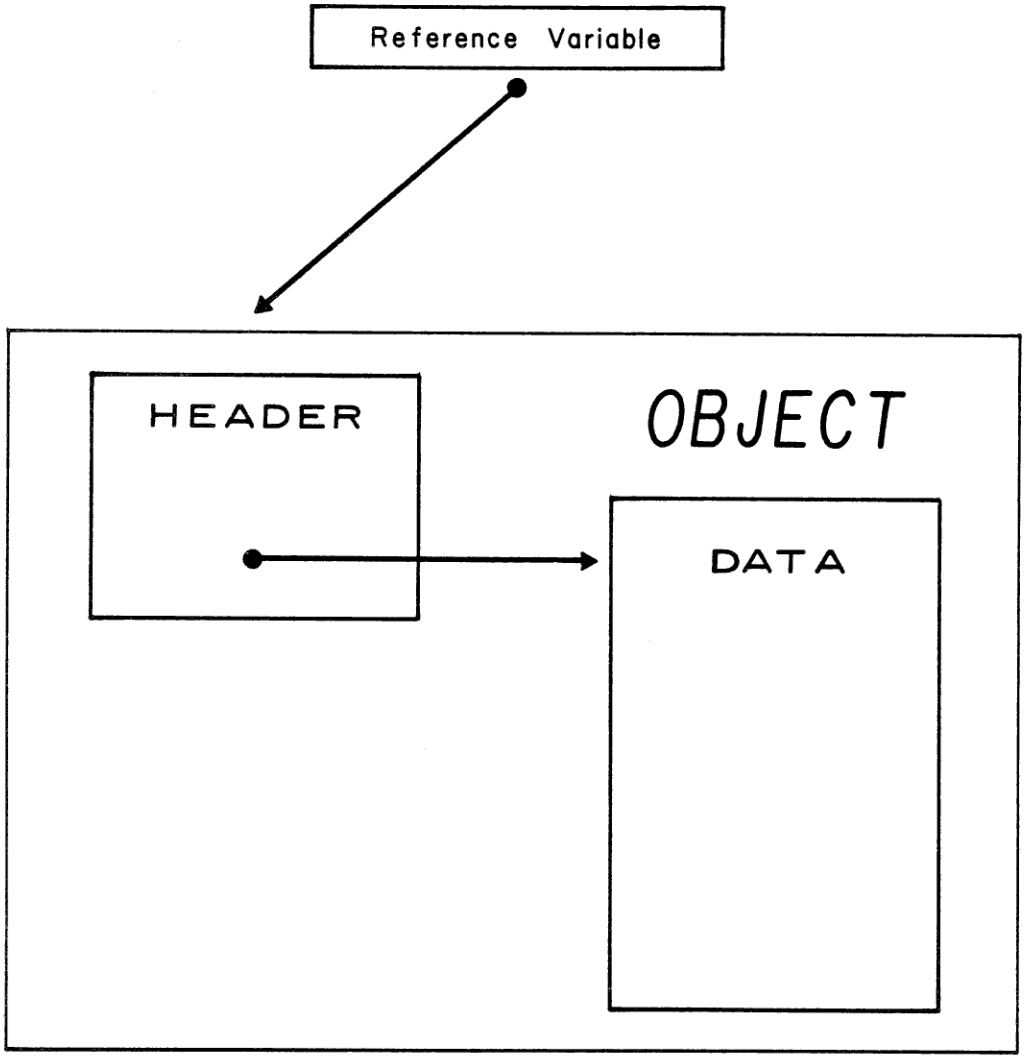


Figure 5 -
Pascal 80 - Object Structure

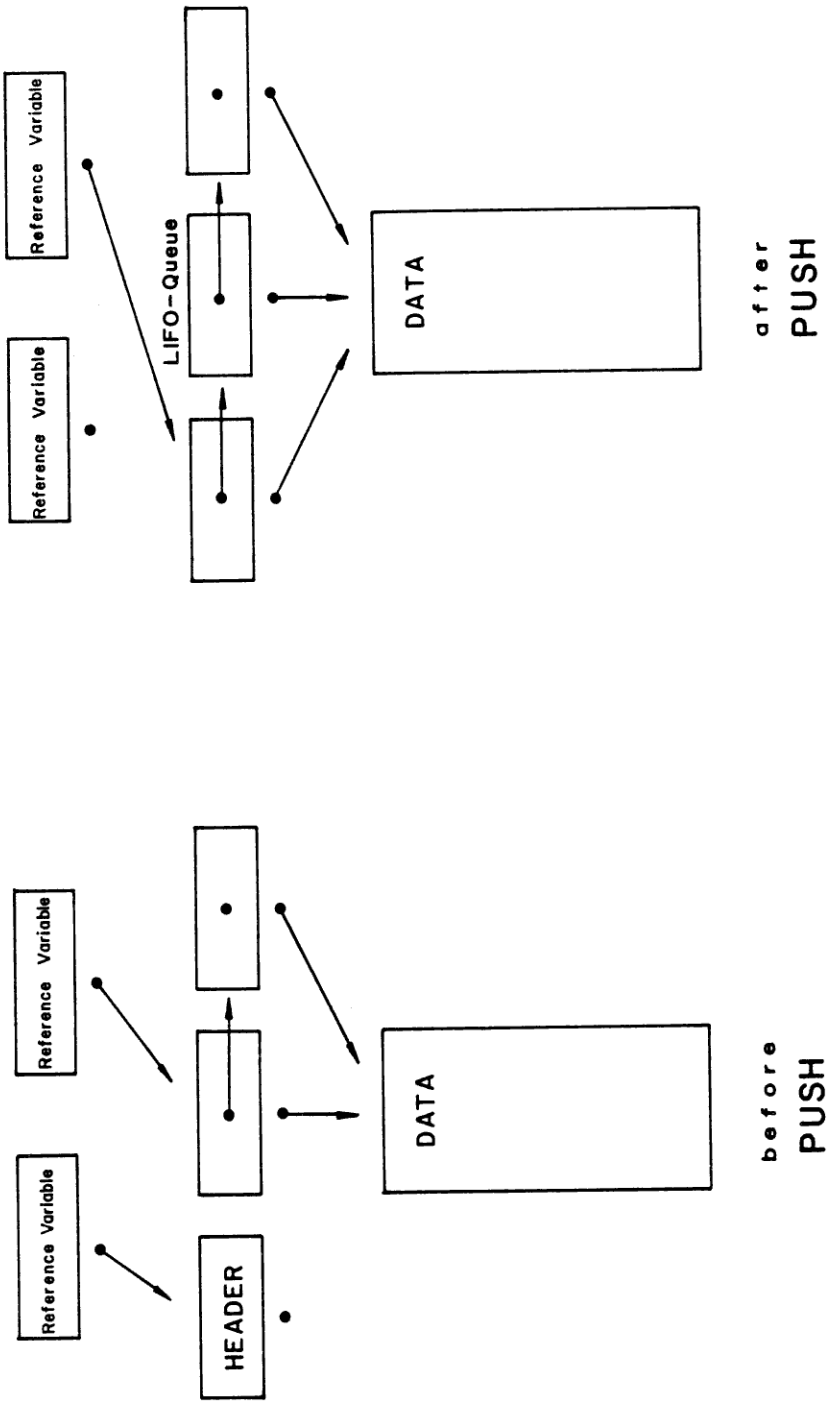


Figure 6 -
Pascal 80 - Stack Operations

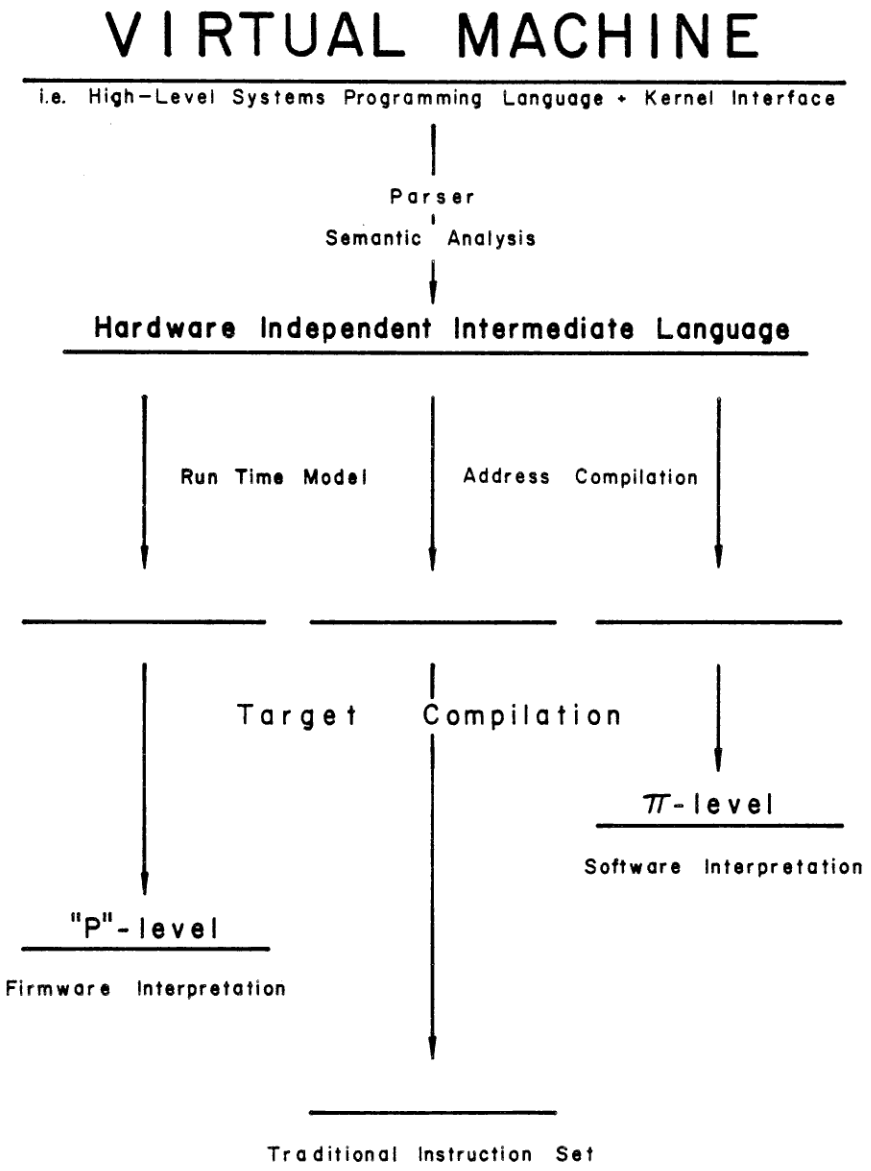


Figure 7 -
Compiler Architecture

Bibliography

Draft Revised CCITT Recommendation X.21
Interface between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for synchronous operation on Public Data Networks.
SG VII/TD 94, February 1980
(The recommendation is reviewed at the CCITT plenary, Geneva, November 1980).

Draft Revised Recommendation X.25
Interface between Data Terminal Equipment (DTE) and Data Circuit-terminating Equipment (DCE) for Terminals operating in the Packet Mode on Public Data Networks.
ACM Computer Communication Review 10.1 & 2 (1980) 56-129
(The recommendation is reviewed at the CCITT plenary, Geneva, November 1980).

CCITT Provisional Recommendation X.29 (1978)
Procedures for the exchange of control information and user data between a packet mode DTE and a packet assembly/disassembly facility.
CCITT Grey Book, Geneva 1978
(The recommendation is reviewed at the CCITT plenary, Geneva, November 1980).

ISO TC97/SC16 N 537
Basic Specifications of the Reference Model of Open Systems Interconnection
November 1980

ISO TC97/SC16 N 568
Annex Describing the Reference Model for Open Systems Interconnection
November 1980

ECMA TC23/80/46
Notes on using the ISO Reference Model of Open Systems Interconnection
March 1980

ECMA TC24/80/67
Final Draft - Transport Protocol
July 1980

CCITT Revised Draft Recommendation S.d.
Control Procedures for the Teletex Service
SG VIII / TD 4, Montreal, June 1980

CCITT Revised Draft Recommendation S.f.
Character Repertoire and Coded Character Sets
for the International Telex Service
SG VIII / Contribution 183, March 1980

P.R.D. Scott
Introducing Data Communications Standards
The National Computing Centre, 1979

T. Jacobsen and P. Thisted
CCITT Recommendation X.25 as part of the Reference Model of Open Systems Interconnection
ACM Computer Communication Review 10.1 & 2 (1980) 48-55

T. Jacobsen
The ISO Reference Model of Open Systems Interconnection
Online Conferences Ltd.
Networks 80, London, June 1980

David Alex Lamb et.al.
The Charrette Ada Compiler
Department of Computer Science
Carnegie, Mellon University
CMU-CS-80-148, October 1980

Søren Lauesen
Program Control of Operating Systems
BIT 13 (1973), 323-337

Biography

Tom Jacobsen obtained his master of science degree in physics and chemistry in 1974 at the University of Copenhagen. He was then appointed scientific associate at the Regional EDP-center at the Technical University of Denmark.

In 1978 the author joined RC Computer A/S where he now works as an international advisor in the department for export and communication networks. He has planned networks in Denmark and Czechoslovakia.

In addition Tom Jacobsen is chairman of the Danish Member Body of the ISO committee for Open System Interconnection. Furthermore, he is involved in development of principles for structured programming. He is also a member of the Simula Development Group and active in the area of new systems programming languages.

FINLAND
RC SCANIPS OY
Espoo, 0 51 35 22

FRANCE
RC COMPUTER S.A.R.L.
Paris, 12 33 53 63

HOLLAND
REGNECENTRALEN (NEDERLAND) B.V.
Gouda 1820-29455

KUWAIT
KUWAITI DANISH COMPUTER CO. S.A.K.
Safat, 83 01 60

NORWAY
A/S RC DATA
Jessheim 29 70 220

SWEDEN
SCANIPS DATA AB
Stockholm, 8 34 91 55

SWITZERLAND
RC COMPUTER AG
Basel, 61 22 90 71

UNITED KINGDOM
REGNECENTRALEN (UK) LTD.
London, 1 606 3252

UNITED STATES
LOCKHEED ELECTRONICS COMPANY, Inc.
New Jersey, 201 757 1600

WEST GERMANY
RC COMPUTER G.m.b.H.
Hannover, 511 63 99 51

RC COMPUTER
A/S REGNECENTRALEN af 1979

HEADQUARTERS: LAUTRUPBJERG 1 - DK 2750 BALLERUP - DENMARK
Phone: + 45 2 65 80 00 - Cables: rcbalrc - Telex: 35 214 rcbaldk