



INTERNATIONAL TELECOMMUNICATION UNION

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Z.600

(11/2000)

SERIES Z: LANGUAGES AND GENERAL SOFTWARE
ASPECTS FOR TELECOMMUNICATION SYSTEMS

Distributed processing environment architecture

CAUTION !

PREPUBLISHED RECOMMENDATION

This prepublication is an unedited version of a recently approved Recommendation. It will be replaced by the published version after editing. Therefore, there will be differences between this prepublication and the published version.



Documents

Document Sharing Hub

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications. The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this Recommendation, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this Recommendation may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the Recommendation development process.

As of the date of approval of this Recommendation, ITU [had/had not] received notice of intellectual property, protected by patents, which may be required to implement this Recommendation. However, implementors are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database.

© ITU 2001

All rights reserved. No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from ITU.

ITU-T RECOMMENDATION Z.600
DISTRIBUTED PROCESSING ENVIRONMENT ARCHITECTURE

CONTENTS

	Page
1 Summary	4
2 Scope.....	4
2.1 Purpose.....	5
2.2 Objective	6
2.3 Non-objectives	6
3 References.....	6
3.1 Identical Recommendation International Standards	6
3.2 Other Specifications.....	6
4 Definitions	7
4.1 Terms defined in the ODP Reference Model: Foundations.....	7
4.2 Terms defined in the ODP Reference Model: Architecture	7
4.3 Definitions for Distributed Processing Environment Architecture.....	7
5 Abbreviations.....	8
6 Introduction.....	9
6.1 DPE Architecture	10
6.2 DPE Reference Points.....	10
6.3 Extensions to CORBA	11
7 Engineering Modelling Concepts	13
7.1 Object Life Cycle.....	13
7.2 Communication in the DPE	14
7.2.1 Operational communication.....	14
7.2.2 Stream communication	14
7.3 Computational to engineering view mapping.....	15
7.4 Bindings	15
7.4.1 Operational vs. Stream Binding.....	15
7.4.2 Explicit vs. Implicit bindings.....	15
7.5 Channels.....	16

7.6	Elaboration on the bindings	17
7.6.1	Introduction.....	17
7.6.2	Programmable stream interfaces.....	18
7.6.3	Control and Management of Media Streams	19
8	Kernel Transport Network	20
8.1	Messaging Layer.....	20
8.2	Transport Layer.....	20
8.3	DPE requirements	21
9	DPE Interoperability	21
9.1	Interoperability Framework	22
9.2	DPE extensions to CORBA 2 interoperability	22
10	Engineering services provided by the DPE kernel	23
10.1	Introduction.....	24
10.2	Approach to real-time	24
10.3	Requirements on the execution model.....	25
10.4	Modularity requirements.....	25
10.4.1	Flexible DPE architecture	25
10.4.2	Multi-protocol support.....	26
10.4.3	Generic communication scheme	26
10.4.4	Support for flexible binding.....	27
10.5	Functional requirements	27
10.5.1	Support for stream interfaces	27
10.5.2	Multithreading support	27
10.5.3	A flexible event-to-thread mapping.....	28
10.5.4	Concurrency management	28
10.5.5	Generic scheduling scheme	29
10.5.6	Time service.....	29
10.6	Non-functional requirements	29
10.6.1	Object granularity	29
10.6.2	Small memory footprint.....	30
10.6.3	Documented time behaviour.....	30
10.7	Multiple Interfaces	30
11	Object Services	31
11.1	Introduction.....	31
12	Bibliography	32
	Annex A - Business modelling concepts	34

Annex B - Engineering modelling concepts	40
Appendix I - Object life cycle scenario	43
Appendix II - Flexible binding.....	44
Appendix III - Issues for the support of stream interfaces.....	45

1 Summary

This Recommendation describes the Distributed Processing Environment (DPE) Architecture, which represents the run-time environment for telecommunication and information services and applications.

The purpose of the DPE Architecture is to provide detailed technical requirements leading to specifications, both to help the DPE vendors to develop their products and the application developer to understand the infrastructure support that the DPE provides.

The material herein is based on work done in the TINA Consortium by the TINA core team and several auxiliary projects in the member companies of TINA-C, supporting the core team.

The Recommendation contains:

- an explanation of the relationship between modelling concepts so far as such a relationship needs to be concerned in the computing architecture;
- a description of the Kernel Transport Network (KTN) which is the DPE analogue of the telecommunications signalling system;
- an interoperability framework for the DPE;
- requirements for the DPE kernel services.

A number of DPE engineering object services can be identified, that support the execution of telecommunication services. These DPE object services are identified and associated to the functions and transparencies of the Reference Model for Open Distributed Processing (RM-ODP). Detailed requirements on the DPE object services and their specifications are for further study.

The DPE, the DPE object services and the applications deployed on a DPE need to be managed. What are the requirements on management, and how can management of these entities be accomplished is also for further study.

Not all DPE kernel services identified in this Recommendation are required for all applications. DPE profiles that support different kinds of services and applications need to be defined. These profiles need to specify which kernel services are mandatory for a given profile. The definition of DPE profiles and their use are for further study. The DPE must support quality of service (QoS) as needed by the services and applications. How this is accomplished by the DPE is for further study.

2 Scope

This Recommendation specifies the Distributed Processing Environment (DPE) which supports the execution of distributed telecommunication applications. The DPE can be regarded as the infrastructure on which distributed telecommunications applications such as multimedia and real-time applications can execute.

This Recommendation provides the basic requirements and functionality for the DPE to support the execution of distributed telecommunications applications. Any distributed telecommunications application designed according to RM-ODP will benefit from using the DPE as a platform. The DPE itself is based on the RM-ODP concepts and principles ([2] and [3]).

RM-ODP specifies a **viewpoint** (on a system) as a form of abstraction achieved using a selected set of architectural concepts and structuring rules, in order to focus on particular concerns within a system. The RM-ODP viewpoints are:

- enterprise viewpoint;
- information viewpoint;

- computational viewpoint;
- engineering viewpoint;
- technology viewpoint.

This Recommendation focuses on the RM-ODP engineering viewpoint. It constitutes a specialization and refinement of the RM-ODP engineering language to take into account specific requirements from the telecommunication domain.

RM-ODP specifies **distribution transparency** as the property of hiding from a particular user the potential behaviour of some parts of a distributed system. The RM-ODP distribution transparencies are:

- access transparency;
- failure transparency;
- location transparency;
- migration transparency;
- relocation transparency;
- replication transparency;
- persistence transparency;
- transaction transparency.

RM-ODP specifies **functions** to support Open Distributed Processing. The RM-ODP functions are:

- management functions;
- coordination functions;
- repository functions;
- security functions.

The RM-ODP distribution transparencies and the ODP functions are associated with DPE kernel services and object services. This Recommendation uses the engineering functions as a basis and, where appropriate, refines them or specializes them for the DPE.

The DPE adopts OMG CORBA 2 as the prime technology base and thus the CORBA common object services are incorporated where appropriate in the DPE.

In particular, this Recommendation encourages the study of Real-Time CORBA [13], since this OMG specification satisfies already a number of requirements identified in this Recommendation.

2.1 Purpose

The purpose of the DPE Architecture is to provide detailed technical requirements leading to specifications, both to help the DPE vendors to develop their products, and the application developer to understand the infrastructure support that the DPE provides. The stated requirements are based on validated results from implementation projects.

In summary this Recommendation contains:

- detailed technical requirements;
- statements of required functionality.

2.2 Objective

The objective of this Recommendation is to provide an abstract description of the DPE. Implementations of the DPE have to support RM-ODP compliant applications as developed according to RM-ODP concepts and methodology. This abstract description of the DPE is implementation independent and should be used by DPE platform vendors as a reference for constructing DPE compliant platforms.

2.3 Non-objectives

This Recommendation describes services that are supposed to be present in many DPE implementations. It provides descriptions of these services without indicating how those services should be combined.

Furthermore this Recommendation does not restrict the actual implementation of the DPE in any way.

3 Normative References

The following Recommendations and International Standards contain provisions, which through reference in this text constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and Standards are subject to revision, and parties to agreements based on this Recommendation are encouraged to investigate the possibility of applying the most recent edition of the Recommendations and Standards listed below. IEC and ISO maintain registers of currently valid International Standards. The Telecommunication Standardization Bureau of ITU maintains a list of currently valid ITU-T Recommendations.

3.1 Identical Recommendations | International Standards

- [1] ITU-T Recommendation X.901 (1997) | ISO/IEC 10746-1:1998, Information technology - Open Distributed Processing - Reference Model: Overview.
- [2] ITU-T Recommendation X.902 (1995) | ISO/IEC 10746-2:1996, Information technology - Open Distributed Processing - Reference Model: Foundations.
- [3] ITU-T Recommendation X.903 (1995) | ISO/IEC 10746-3:1996, Information technology - Open Distributed Processing - Reference Model: Architecture.
- [4] ITU-T Recommendation Z.130 (02/99) - ITU object definition language. .
- [5] ITU-T Recommendation X.931 (06/99) - Information technology - Open distributed processing - Protocol support for computational interactions.

3.2 Other Specifications

The edition of CORBA 2 indicated below was valid at the time of publication of this Recommendation. CORBA 2 is subject to revision, and parties to agreements based on this Recommendation are encouraged to investigate the possibility of applying later editions of CORBA 2 when they become available.

- [6] OMG, The Common Object Request Broker: Architecture and Specification, Revision 2.4: October 2000.

4 Definitions

For the purposes of this Recommendation, the following definitions apply.

4.1 Terms defined in the ODP Reference Model: Foundations

This Recommendation makes use of the following terms defined in ITU-T Recommendation X.902:

- a) architecture (of a system);
- b) domain;
- c) interface;
- d) object;
- e) template;
- f) thread.

4.2 Terms defined in the ODP Reference Model: Architecture

This Recommendation makes use of the following terms defined in ITU-T Recommendation X.903:

- a) binder;
- b) capsule;
- c) channel;
- d) checkpoint;
- e) cluster;
- f) kernel (nucleus);
- g) node;
- h) stub.

4.3 Definitions for Distributed Processing Environment Architecture

This Recommendation makes use of the following terms:

Business Administrative Domain: A business administrative domain is defined by the requirements of one or more business roles and is governed by a single business objective.

Business Role: The expected function performed by a stakeholder in a telecommunications business environment.

Communication API: The application programming interface through which the communication services are accessible

DPE (Distributed Processing Environment): The DPE is the infrastructure that provides the execution environment, including distribution transparencies, for distributed applications in a system. The DPE provides a distribution transparent view to its users. The users of the DPE are the application and service designers and developers. The DPE consists of a collection of DPE nodes that are interconnected.

DPE API: The application programming interface through which the DPE services are accessible.

DPE reference point: The engineering view of a reference point.

Engineering Computational Object (eCO): An eCO is the engineering representation of a computational object (one-to-one correspondence), which encapsulates state/data and processing.

Engineering service: Engineering services support the DPE distribution transparencies. There are a number of engineering services that are required by a wide range of applications (e.g. repository, object adaptor) and are fundamental for the construction of applications that execute on the DPE.

Generic Inter-ORB Protocol (GIOP): The GIOP is a messaging protocol used for object communication between different DPE nodes. The GIOP is specified in [6].

Internet Inter-ORB Protocol (IIOP): The IIOP is the mapping of the GIOP to TCP/IP. The IIOP is specified in [6].

Inter-domain Reference Point: Same as Reference Point. Unless otherwise stated in this Recommendation, an inter-domain reference point is implied.

Intra-domain Reference Point: A reference point not visible outside a business administrative domain. An intra-domain reference point is mainly used for procurement.

Kernel transport network (KTN): The kernel transport network is the network which transports invocations and responses between different DPE nodes.

NCCE (Native Computing and Communication Environment): The NCCE is a computer which generally consists of a set of resources (hardware and operating system) that together provide a distinct platform. The NCCE is outside the scope of the DPE. Typically each DPE node runs on a NCCE and uses its resources and services. For example a DPE node would use a TCP/IP stack provided by the NCCE for communication with other DPE nodes.

Object Request Broker (ORB): The ORB provides mechanisms for transparently communicating distributed client object requests to server object implementations, and for returning any server object responses to the requesting client object. The ORB is specified in [6].

Object service: Object services support the execution of distributed applications.

Reference Point: A reference point marks a boundary between business administrative domains. A Reference Point is a set of several viewpoint related specifications within a context, defining constraints for one or more reference points to operate under (see Annex A).

5 Abbreviations

AAL	ATM Adaptation Layer
API	Application Programming Interface
ACID	Atomicity, Consistency, Isolation, Durability
CORBA	Common Object Request Broker Architecture
CMIP/CMIS	Common Management Information Protocol/Common Management Information Services
DCE-CIOP	Distributed Computing Environment - Common Inter-ORB Protocol
DPE	Distributed Processing Environment
eCO	engineering Computational Object
ESIOP	Environment Specific Inter-ORB Protocol
GIF	Generic Interworking Framework (X.931)
GIOP	General Inter-ORB Protocol
IDL	Interface Definition Language

IIOP	Internet Inter-ORB Protocol
IOR	Interface Object Reference
IPC	Interprocess Communications
KTN	Kernel Transport Network
LAN	Local Area Network
MPEG	Moving Picture Experts Group
NCCE	Native Computing and Communication Environment
ODL	Object Definition Language
OMG	Object Management Group
ORB	Object Request Broker
POSIX	Portable Operating System Interface
QoS	Quality of Service
RM-ODP	Reference Model for Open Distributed Processing
SCCP	Signalling Connection Control Part
SNMP	Simple Network Management Protocol
SS7	Signalling System 7
TCAP	Transaction Capabilities Application Part
TCP/IP	Transmission Control Protocol/Internet Protocol
TINA	Telecommunications Information Networking Architecture
UDP	User Datagram Protocol

6 Introduction

This Recommendation provides the architecture and the requirements for building large distributed processing environments, supporting the execution of telecommunication services and applications, including support for real-time services and applications.

Middleware can be defined as a set of software components that supports the interconnection of distributed systems and services. Middleware are software products that can be positioned between or underneath applications and on top of system software, computing systems and communication networks and services. In most cases, middleware is hiding the heterogeneity of the underlying systems, e.g. programming languages, operating systems, computing systems and network protocols. From the system developer's point of view, middleware provides a homogeneous view of a well structured, heterogeneous distributed environment, where applications and services inter-operate through standardized open interfaces. The DPE is an instance of middleware.

The DPE is the infrastructure that provides the execution environment, including distribution transparencies, for distributed applications in a system. The DPE provides a distribution transparent view to its users. The users of the DPE are the application and service designers and developers. The DPE consists of a collection of DPE nodes that are interconnected.

This Recommendation provides an abstract description of the DPE. Implementations of the DPE have to support RM-ODP compliant applications as developed according to RM-ODP concepts and

methodology. This abstract description of the DPE is implementation independent and should be used by DPE platform vendors as a reference for constructing DPE compliant platforms.

The DPE includes a DPE kernel, which provides support for object life cycle control and inter-object communication. Object life cycle control includes capabilities to create and delete objects during run-time. Inter-object communication provides the mechanisms to support the invocation of operations provided by operational interfaces of remote objects. The DPE kernel provides basic, and technology independent functions that represent the capabilities of most operating systems, i.e. the ability to execute applications and the ability to support the communication of applications with each other.

6.1 DPE Architecture

The DPE architecture as illustrated in Figure 1/Z.600 is the unifying entity that encompasses all the required functionality described in this Recommendation, into a standardized implementation. It does this by presenting to applications a homogeneous view of a well structured, heterogeneous distributed environment, where applications and services interact through standardized open interfaces. This architecture hides the heterogeneity of the underlying systems, e.g. programming languages, operating systems, computing systems and network protocols.

The DPE architecture provides the middleware for interaction between engineering computational objects on remote DPE nodes. It also provides tools for diagnosis and configuration for use by network administrators.

The components of the DPE Architecture are categorized in:

- DPE Kernel;
- DPE Object Services;
- DPE Support Tools.

A DPE node is controlled by one DPE kernel (see also Annex B).

There can be a DPE reference point between any two DPE nodes.

6.2 DPE Reference Points

The concepts **Business Administrative Domain** and **Reference Point** are used as explained in Annex A.

A Reference Point, which is defined as a set of related specifications, is positioned between two business administrative domains. The DPE reference point is the engineering view of a reference point and supports the service related reference point as illustrated in Figure 2/Z.600. The DPE API is used by the services and applications to access the functionality of the DPE.

A reference point marks a boundary between business administrative domains, thus a DPE reference point marks a domain boundary between DPE nodes operating under distinct business administrative domains.

Communication between distinct DPE nodes takes place over a DPE reference point.

Communication between services or between DPE nodes can take place in two different ways:

- **Client-server:** A common way to communicate between objects, in which software is split between server tasks and client tasks. A client sends requests to a server, according to some protocol, asking for information or action, and the server responds. There may be either one centralized server or several ones. This model allows clients and servers to be placed

independently on nodes in a network, possibly on different hardware and operating systems appropriate to their function, e.g. fast server/cheap client.

- **Message passing:** A technique for communicating between parallel processes. A service running on one node may send a message to a service running on the same or another node. The execution environment handles the actual transmission of the message. A message passing environment provides primitives for sending and receiving messages. These primitives may be either synchronous or asynchronous or both. A synchronous send will not complete (will not allow the sender to proceed) until the receiving process has received the message. This allows the sender to know whether the message was received successfully or not. An asynchronous send simply queues the message for transmission without waiting for it to be received. A synchronous receive primitive will wait until there is a message to read, whereas an asynchronous receive will return immediately, either with a message or to say that no message has arrived. Messages may be sent to a named service or to a named mailbox, which may be readable, by one or many services. Messages may be typed or non-typed. They may have a priority, allowing the receiver to read the highest priority messages first.

6.3 Extensions to CORBA

OMG CORBA 2 [6] is a mapping of the DPE Architecture into a specific technology. Although the DPE architecture is based to a great extent on CORBA 2, it does not repeat CORBA 2 functionality and describes the additional requirements that arise in the telecommunications domain.

The DPE kernel requires additional functionality other than what is provided by OMG CORBA 2. Examples of additional required functionality are:

- support for an enhanced object model, e.g. support for multiple interfaces and stream interfaces;
- support for and integration of media streams;
- integration of quality of service aspects;
- support for real-time services and applications;
- support for different networks for inter-DPE communications.

At the level of object services, many OMG common object services as specified in OMG CORBA Services [7] can be reused to build a DPE.

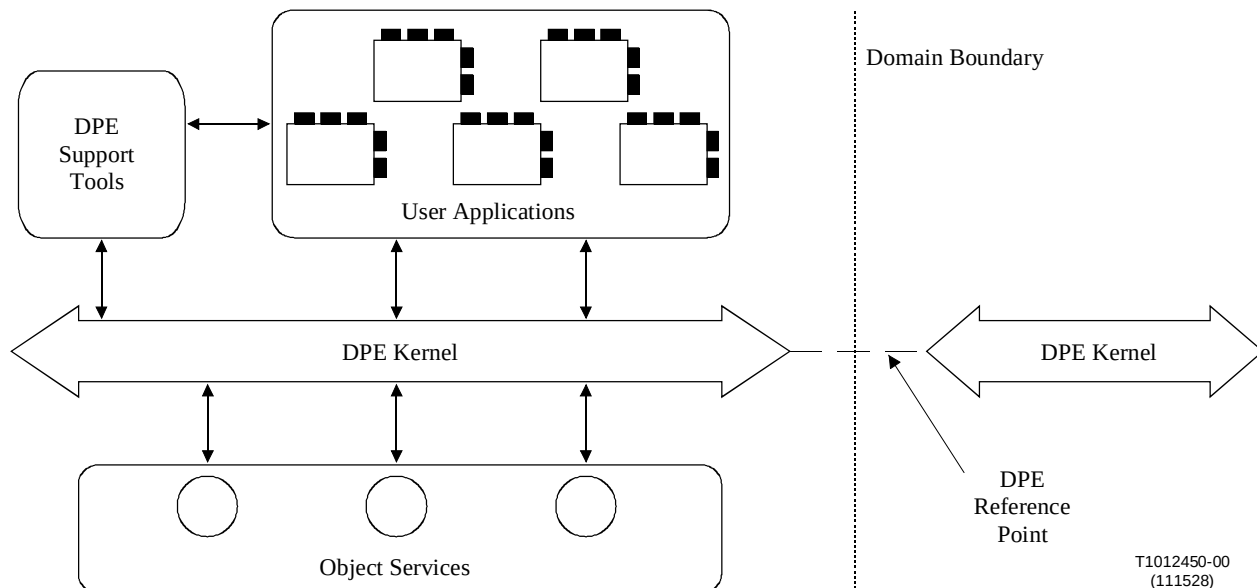


FIGURE 1/Z.600

Architecture

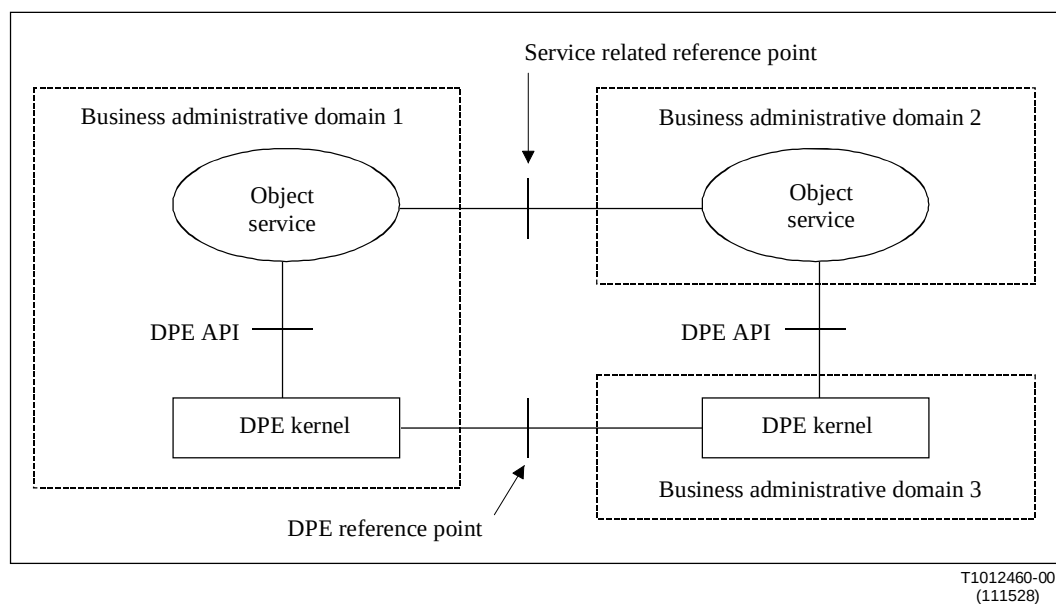


FIGURE 2/Z.600

Reference points and APIs

7 Engineering Modelling Concepts

This Recommendation uses the basic engineering modelling concepts as defined in [3] and described in Annex B. In particular Annex B elaborates on the notions of:

- DPE node;