Implementation Experience with the OMG IN/CORBA Interworking Specification

C. McArdle, R. Brennan, N. Jones, J. Vasic and T. Curran
Teltec Ireland
Contact: Conor McArdle, mcardlec@teltec.dcu.ie, Teltec, phone +353 1 7045569, fax +353 1 7045092
Teltec Ireland, Dublin City University, Glasnevin, Dublin 9, Ireland

ABSTRACT

The Object Management Group has recently adopted a specification to standardise interworking between Signaling System No.7 systems and CORBA-based systems. This specification defines two types of interworking mechanism: 1. Interworking between CORBA-based TC-User Application Entities (such as CORBA-based Service Control Points) and legacy TC-User Application Entities (such as Service Switching Points), where communication between the Signalling System No.7 and CORBA domains is through a gateway mechanism that provides a CORBA view of a legacy target and a legacy view of a CORBA target. This is specified as the gateway approach; 2. Interworking between islands of CORBA-based systems using the existing Signalling System No.7 infrastructure as a transport network for CORBA messages between Application Entities. This is specified as the SCCP Inter-ORB Protocol. This paper reviews the current specification for the gateway approach and presents some enhancements that are based on practical implementation experience. An implementation of the gateway specification is outlined and performance characteristics are presented.

1. INTRODUCTION

In recent years, the deregulation of public telecommunications networks has presented network operators and service providers with a rapidly changing environment. In this environment, service creation strategies have been driven by the need for fast time to market for new services and by the requirements for a high degree of service customisation. However, as the number of deployed services increases, the effort involved in interworking new and existing services has dominated the actual effort devoted to development of new services. Furthermore, network intelligence has been viewed and implemented as tightly coupled, vertically integrated applications, with dialogue capabilities in the form of standardised capability sets. The disadvantages of this approach are little software re-use, reduced scalability and performance, cumbersome service management and limited interconnection capabilities with external resources such as the Internet and private databases.

For these reasons, many actors in the telecom arena are currently considering solutions from the mainstream computer industry to help solve key issues regarding the future evolution of their systems. In particular, the need to differentiate the offering of value-added services is pushing network operators to consider the possibility of introducing top-end software technologies for the control of network intelligence. Equipment vendors are increasingly employing object oriented design principles and distributed processing platforms for the implementation of new generation IN systems. Current SCP platforms from many vendors are already structured following a client/server model in which UNIX-based servers are connected through high-speed data networks, with back-end distributed computing. New technologies in real-time database management systems already provide effective solutions to data replication issues, while distributed processing environment standards, such as the Object Management Group's (OMG) Common Object Request Broker Architecture (CORBA), are already in use for service management applications.

Recent initiatives have also considered the application of CORBA to real-time applications within IN and wireless network elements with a view towards standardising an open software creation and computing environment. Middleware technologies such as CORBA are increasingly seen as the appropriate infrastructure in a value-added telecom network. The reasons for this move towards the adoption of CORBA technology include: the ability to provide flexible system scalability, the ability to leverage commercial off-the-shelf IT technologies, the advantages of an open standards process, the ease of system integration with existing working systems, the ability to leverage new technologies as they emerge, and the avoidance of technology and vendor lock-in. Much of the investigation into the application of CORBA to IN systems has been initiated by the Eurescom P508 project [5], the goal of which was to determine the options for evolving from legacy systems towards TINA. A major result of the project was that the gradual introduction of a TINA DPE (i.e. CORBA technology enhanced with real-time capabilities) into the existing IN environment represents a fundamental prerequisite for such an evolution.

During the course of the P508 project, White Papers were produced [1], [2] and submitted to the OMG in order to support the then emerging activities on IN/CORBA interworking. These White Papers were targeted at providers of information technology solutions and had the purpose of stimulating their interest towards telecommunication operator specific needs. They analyse a specific element of the problem area – the introduction of CORBA into the Intelligent Network. The central idea put forward is to adopt the OMG CORBA standard, enhancing it to make it suitable for telecommunications systems, particularly IN. Subsequent to the White Papers, the OMG issued a Request for Proposals [3] that sought proposals for interworking between CORBA and Intelligent Networks systems. Subsequently, the work was continued within the Telecommunications Domain Task Force of the OMG, which has recently produced a standard [4], which focuses on the interworking of CORBA-based systems with TC-User applications, such as traditional IN and mobile systems. This standard is a joint submission from AT&T, GMD FOKUS, Nortel, IONA Technologies and Teltec Ireland in collaboration with Alcatel, Deutsche Telekom, Ericsson Telecommunications, Humboldt University, Object Oriented Concepts Inc. and Telenor. The standard is currently being implemented and evaluated.

Section 2 of this paper briefly outlines IN-CORBA Interworking as proposed in the standard. Section 3 discusses some enhancements, which are to be submitted to the OMG Finalisation Task Force for the specification. These enhancements are based upon practical implementation experience with the specification. In Section 4, an implementation of a practical SS7/TC-User to CORBA Gateway, which is conformant to the specification, is examined in terms of performance issues. Section 5 concludes the paper.

2. OVERVIEW OF IN/CORBA INTERWORKING

The OMG specification for interworking between CORBA-based and SS.7/TC Systems [4] defines two types of interworking mechanism:

Interworking between CORBA-based TC-User Application Entities (e.g. CORBA-based Service Control Points (SCP), and legacy TC-User Application Entities (e.g. Service Switching Points (SSP)) through a gateway mechanism that provides a CORBA view of a legacy target and a legacy view of a CORBA target. This is specified as the gateway approach.

Interworking between islands of CORBA-based systems using the existing Signalling System No.7 (SS7) infrastructure as a transport network for GIOP messages. This is specified as the SCCP Inter-ORB Protocol (SIOP).

These interworking mechanisms correspond to the EURESCOM P508 conclusion that there is a need to standardise both a gateway and a kernel transport network [5]. An overview of the gateway interworking standard is provided here as background to the work discussed in this paper. For a more complete introduction to the standard, see [8].

2.1 The Gateway Approach

The gateway approach is similar to earlier work on using CORBA for telecommunications management, which was jointly carried out by the NMF and X/Open [6]. In Figure 1, the CORBA-based TC-User Application Entity has IDL interfaces that are created through *Specification Translation* of the ASN.1 specifications of the TC-User protocol. This IDL-based specification provides a uniform interface that may be used when implementing either native CORBA-based Application Entities (AEs) or AE proxy CORBA objects at a TC-CORBA gateway. This uniformity is essential to ensure location transparency and eliminate the need for proxy objects for native CORBA to CORBA interactions. The translation algorithm is an extension of the previous NMF/The Open Group Joint Inter-Domain Management Task Force (JIDM) work on ASN.1 to IDL specification translation [6].

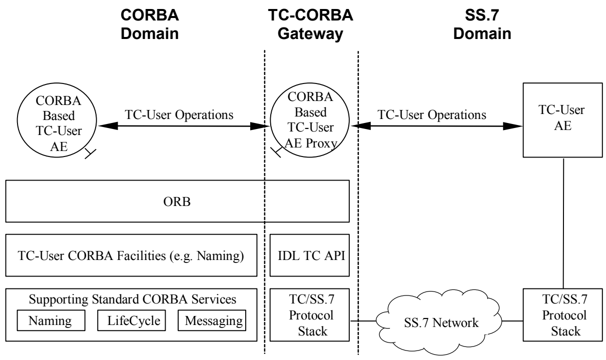

Figure 1: The Interworking Gateway

In order to support TC-User interaction semantics (naming, dialogues, etc.) in the CORBA domain, CORBA TC-User Facilities have been defined that reuse some of the CORBA Object Services (this is the *Interaction Translation* part of the specification). This allows maximum re-use of the CORBA infrastructure when using it as an environment for developing TC-User applications. It also means that building an TC-User application is simplified, as most of the TC-specific functionality has been encapsulated by specialising the CORBA Object Services and by provision CORBA TC-User Facilities by the standard.

In addition to the interaction and specification translations defined in the standard, a CORBA-IDL API for access to the TC service of a TC/SS.7 stack is defined. These *TC PDU-oriented* interfaces are designed to standardize access by TC-aware CORBA objects (such as proxy objects at a gateway) to a TC/SS7 protocol stack. This allows implementations of gateways that are independent of a particular SS7 stack vendor. At this time, most stack vendors offer proprietary APIs to their TC/SS7 stack. It is not, of course, necessary to use the TC PDU-oriented interfaces to implement a TC/CORBA gateway, as the custom mapping onto a particular TC/SS7 stack may be a part of the implementation of the proxy interface generated during Specification Translation. However, these interfaces can be useful if there is a need to build a distributed gateway which is not too closely coupled with the stack platform and hardware. These interfaces represent a low-level mapping that requires users to be aware of the TC service primitive interface defined in ITU-T Rec. Q.771 [7]. They also requires users to encode/decode ASN.1/BER data based on ITU-T Rec. Q.773 [7]. These interfaces can also be used to build TC-aware CORBA applications that do not rely on the ASN.1 to IDL translation algorithms specified but instead use some proprietary mechanism.

3. SPECIFICATION ENHANCEMENTS BASED ON IMPLEMENTATION EXPERIENCE

In this section we discuss some enhancements made to the current specification of the TC PDU-oriented interfaces. These enhancements are based upon practical implementation experience and are to be submitted to the OMG Finalization Task Force for the IN/CORBA Interworking specification. One major improvement to the current architecture and one additional clarification are presented here.

3.1 The Enhanced TC-PDU-Oriented Interfaces

The current TC PDU-oriented architecture defines three interfaces, two must be supported by the SS.7/TC stack (a factory interface for TC sessions and a TC session interface), one (a TC session call-back interface) must be supported by the stack's client application. Typical operation consists of a client application starting a session by requesting the factory to create a TC session interface and calling methods on the session object equivalent to TC dialog handling and component handling primitives to start and control TC dialogs. This is illustrated in the Figure 2 below.

This part of the specification works well, i.e. for dialogues originating on the CORBA side of the gateway. The solution is scaleable as any CORBA application supporting the call back interface may create new TC session (TcPduProvider) objects. However there is a problem for calls originating on the SS.7 side, in this case a TcPduUser object must register with the TcPduProviderFactory object. Only one instance of a TcPduUser may register for a given SS.7 Global Title (GT) and Application Context (AC) pair. This does not exactly match the architecture given for the interaction translation of TC-User AEs. In the interaction translation, the SS.7 address (GT and AC pair) maps to a CORBA factory object for the AC type. This means that there can be many instances of a CORBA Application Context object type for a single SS.7 Global Title. Interaction/specification translation proxy objects at a TC/CORBA gateway that use the TC PDU-oriented interfaces for communication to the SS.7 stack are thus limited to one instance of the TcPduUser interface to handle all call-backs from the stack. This has several disadvantages, it is not as scaleable as allowing a one to one mapping between call back interfaces and proxies, it

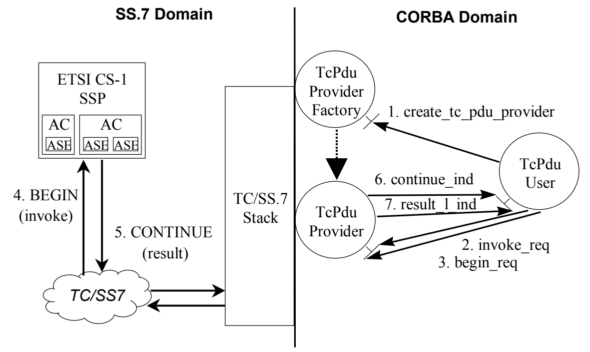

Figure 2: The TC PDU-oriented interfaces

hinders distribution of proxy objects over multiple CORBA network nodes and it increases the complexity of implementation of a gateway which dynamically supports a variety of proxy objects. The solution suggested here is to modify the current IDL so that instead of registering a TcPduUser object for a GT and AC pair, a new interface TcPduUserFactory is registered instead. All of the disadvantages currently encountered are no longer present in this solution. Of course a small extra cost is incurred in dialog setup as an additional method call must now be made (create_tc_pdu_user). This cost can be negated in practical implementations by pre-creating a desired number of call-back objects for each GT/AC pair supported. In addition to changing the registration operation and adding a TcUserFactory interface, it is necessary to modify the get_dialog_id operation in the TcPduProvider interface. This is because the TcPduProvider may now deal with multiple TcPduUser objects and it must be able to associate incoming TC service primitive requests with the correct call-back interface. The modified IDL to support this behaviour is listed below.

```
module TcSignaling {
// skip all unchanged definitions
interface TcPduProvider {
// skip unchanged definitions
// replace the current get_dialog_id operation
with the following:
DialogId get_dialog_id(TcPduUser user)
raises (NoMoreDialogs);
}; //end TcPduProvider
interface TcPduProviderFactory{
// skip unchanged definitions
```

```
// replace the current register operation with
the following:
void register (in TcSignaling::TcAddress dest,
in ApplicationContext a_c,
in TcPduUserFactory user_factory)
raises(AlreadyBound);
}; //end TcPduProviderFactory
interface TcPduUserFactory{
TcPduUser create_tc_pdu_user
(in ApplicationContext application_context)
raises(NoMoreDialogs);
}; // end TcPduUserFactory
}; // end TcSignaling
```

A further suggested enhancement to the specification involves a modification to the TcPduUser and TcPduProvider primitive handling operations. Currently, TC dialog and component handling operations on these interfaces are distinct operations as shown in Figure 2. A sample from the modified IDL shown below allows TC dialog and component handling to be encapsulated in one operation. This is expected to give a considerable performance improvement.

```
void begin_ind (in GwTcPduHandler sender,
                in AeAddress orig,
                in AeAddress dest,
                in DialogQos qos,
                in DialogPortion d_p,
                in DialogId d_id,
                in ComponentList c_list)
```

3.2 Extended Support for SS.7 Addressing

In the current IN/CORBA Interworking specification, there is a limitation on the types of SS.7 addresses that can be used. In fact, only Global Title addressing is supported. For some applications and network environments this is not adequate where all three SS.7 addresses, (namely the Global Title, the Point Code and the Sub-System Number) are required.

The specification currently allows addressing of the form:

```
typedef Istring TcAddress;
```

which was originally intended only to carry a Global Title. This normally consists of dialed digits or the Mobile Identification Number (MIN) of a mobile user. In some cases, the Global Title is used to route messages only up to a certain point in the network and is then translated to provide more detailed addressing in the form of Sub-System Numbers and Point Codes, which complete the routing to the final location. The Point Code may identify a particular SCP and the Sub-System Number may identify a particular database associated with that SCP, for example. In order to accommodate all possible SS.7 addressing schemes, the `TcAddress` is expanded to include the Sub-System number and the Point Code as listed in the modified IDL listed below:

```
struct TcAddress {
   Istring            global_title;
   Istring            point_code;
   unsigned short     sub_system_number;
   // note that sub_system_number is
   // constrained to the values 0-254
};
```

4 INITIAL PERFORMANCE TESTING

In the interest of evaluating the suitability of the TC-CORBA Gateway approach for provisioning of CORBA-based telecom applications, this section presents some initial performance results for a TC-CORBA Gateway implementation based on the TC-User interfaces described in Section 2 and the enhancements described in Section 3. The following sections describe the test environment for the Gateway and define suitable performance metrics for its evaluation.

4.1 The Gateway Test Environment

The execution environment for the initial performance testing of the Gateway is shown in Figure 3. A comercial ORB and SS.7 implementaion have been used for testing. The TC-CORBA Gateway, SS.7 stack, test application and the traffic generator are deployed across three physical nodes, as shown. The traffic generator provides a source of test messages to be passed to the Gateway node over SS.7 and also acts as a sink for messages from the Gateway. The SS.7 messages are based on a simple IS41-MAP application, with the application data being carried as ASN.1 BER encoded MAP over TCAP. The generated messages are passed to the SS.7 stack on the Gateway node via a V35 link.

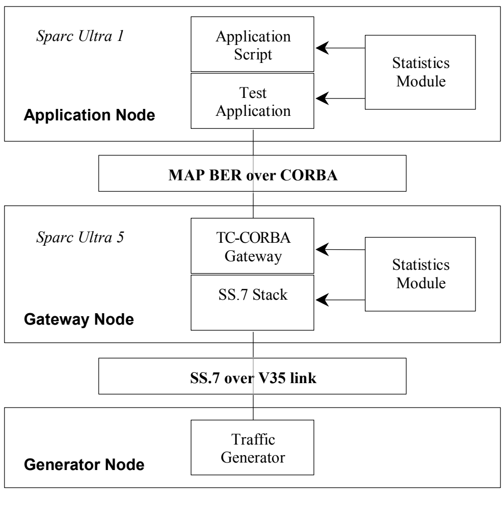

Figure 3: The Gateway Test Environment

The TC-CORBA Gateway module resides on the Gateway node and interfaces with the SS.7 Stack module at the TCAP level. TCAP messages arriving from the stack are translated to their IDL equivalent by the Gateway module and forwarded to the test application via invocations on a standard `TcSignaling::TcPduUser` CORBA object on the Application Node. The TC-CORBA Gateway module exposes a standard `TcSignaling::TcPduProvider` CORBA interface through which the Test Application may reply to invocations from the Gateway. The Gateway module performs the extraction of the MAP application data, which is carried in the received IDL, and passes it back through the SS.7 stack. CORBA invocations between the Gateway and Test nodes are carried over a standard 10 Mb Ethernet link via TCP/IP. A Statistics module at the Gateway node allows recording of timing data related to the stack and Gateway module that may be stored for use in off-line analysis.

The Application node provides a platform for running the CORBA-based MAP application. Invocations arriving from the Gateway node are processed by the Test Application according to a pre-programmed message sequence, stored in an Application Script. The Test Application then generates appropriate MAP messages in response to messages received from the Gateway and passes them back to the Gateway via the `TcSignaling::TcPduProvider` CORBA interface. A Statistics module on the Application node logs timing data specific to the Test Application and the CORBA communication to and from the Gateway node. The timing measures recorded in the test environment are discussed in Section 4.3.

Thus, the Gateway Test Environment allows simulated test applications to run between the Application node and the Traffic Generator via the Gateway and provides a method for collection of performance statistics.

4.2 The Test Application

The test application, running on the platform described above, consists of one MAP operation being passed from the Generator to the Test Application through the SS7 Stack and the Gateway node and a result message being passed back from the Test Application to the Generator via the reverse route. Thus, the Generator initiates a simple TCAP dialogue with the application, which responds by sending a result back to the Generator thus ending the dialogue. The message sequence for the application is shown in Figure 4.

The `begin_ind` operation is invoked on the `TcPduUser` CORBA interface object that resides on the application node. This operation carries the invoke which contains the BER encoded MAP application data as an IDL octet sequence. Similarly, in the reverse direction, the `end_req` operation is invoked on the `TcPduProvider` CORBA interface object that resides at the Gateway. This operation carries the result which contains the BER encoded application data that is the result of the invoked operation. Interactions with the `TcSignaling::TcPduProviderFacotory` object, which is prescribed by the specification, are not included in the test application as it is assumed that a single `TcPduProvider` object may handle many dialogue sessions at the same time. The `TcPduProviderFactory` is used to initialise the Gateway with a number of `TcPduProvider` objects but is not contacted during normal running of the Gateway. References to these `TcPduProvider` objects are cached by the test application during initialisation and used as required thereafter. Note that the specification does not mandate the use of the Factory for every TCAP dialogue session.

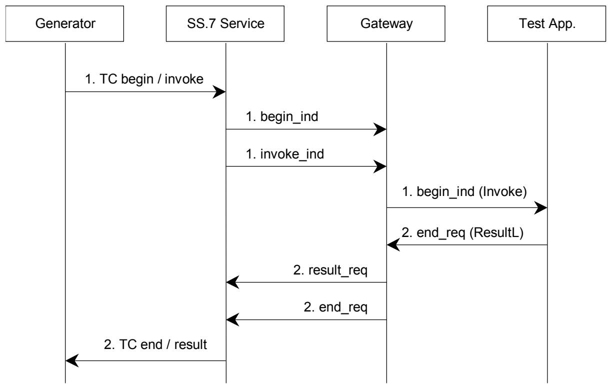

Figure 4: Message Sequence for Test Application

The size of the MAP application data was chosen to reflect a typical application. Each message initiated at the Generator is timed at various points during its processing. Performance measures derived from this timing data are discussed in the following section.

4.3 Performance Measures

In order to obtain initial performance metrics for the IN/CORBA Gateway, various timing measures for processing in the test application were collected from the test environment. The complete transaction time for the application is decomposed into a number of relevant time points at the Gateway node and Test node as shown in Figure 5. The processing times indicated are:

$t_2 - t_1$: the processing time associated with the SS.7 stack on the Gateway node for message 1

$t_3 - t_2$: the processing time in the Gateway module for message 1. This includes processing in the TCAP layer at the Gateway/SS.7 stack interface

$t_4 - t_3$: the processing time in the ORB for message 1. This includes the marshalling (protocol encoding) time for the `begin_ind` CORBA invocation at the Gateway node, the transport time over the network and the demarshalling (protocol decoding) time at the application node

$t_6 - t_5$: the processing time in the ORB for message 2. This includes the marshalling (protocol encoding) time for the `end_req` CORBA invocation at the Application node, the transport time over the network and the demarshalling (protocol decoding) time at the Gateway node

t_7 - t_6 : the processing time in the Gateway for Message 2. This includes processing in the TCAP layer at the Gatway/SS7 stack interface

t_8 - t_7 : the processing time associated with the SS.7 stack on the Gateway node for message 2

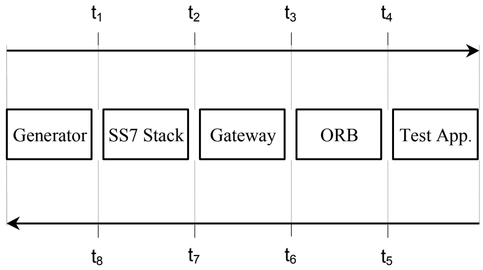

Figure 5: Timing Points for Test Application

4.4 Results and Analysis

The experimental results for the timing measures identified above are given in Table 1. Of primary interest to investigation of performance issues are the metrics defined as follows:

τ_g The total processing time, per message, at the Gateway node. This is composed of the processing related to the SS.7 stack, processing in the Gateway module and processing related to encoding or decoding of CORBA invocations. It is assumed that marshalling and demarshalling times are equal and that, compared to these times, the transport times in the network are negligible. The total time may be calculated as follows:

message 1 : $\tau_g = (t_2-t_1) + (t_3-t_2) + [(t_4-t_3)/2]$

message 2 : $\tau_g = [(t_6-t_5)/2] + (t_7-t_6) + (t_8-t_7)$

τ_T The total time for a message to pass through the entire system. This is composed of processing in the SS7 Stack, processing in the Gateway module and encoding and decoding of CORBA invocations at the gateway, and encoding and decoding of CORBA requests at the Application node. This time may be calculated as follows:

message 1 : $\tau_T = (t_2-t_1) + (t_3-t_2) + (t_4-t_3)$

message 2 : $\tau_T = (t_6-t_5) + (t_7-t_6) + (t_8-t_7)$

The total processing time at the Gateway, τ_g, gives an indication of the obtainable throughput for the Gateway node. It is independent of any particular implementation of the application node. If measured when the Gateway is very lightly loaded, i.e. there are negligible queueing delays in the SS.7 and Gateway layers, then $1/\tau_g$ gives the theoretical service rate for the gateway node.

The total time for a message to pass through the entire system, τ_T, gives an indication of overall performance of a typical CORBA-based system designed in accordance with the specification.

t_2-t_1 (ms)	t_3-t_2 (ms)	t_4-t_3 (ms)	t_6-t_5 (ms)	t_7-t_6 (ms)	t_8-t_7 (ms)
0.42	0.38	0.85	0.75	0.47	0.61

(ms)	τ_g	τ_T
message 1	1.23	1.65
message 2	1.46	1.83
average	1.35	1.74

Table 1. Experimental Results

5 CONCLUSIONS

There are many motivating factors, in terms of technical advantages and business drivers, for the migration of traditional INs toward object-oriented, distributed computing middleware platforms such as CORBA. A standardised interworking gateway is a key element in this migration path, allowing CORBA-based systems to coexist with legacy systems in an open environment. This paper has presented a practical view of the OMG standard for IN/CORBA Interworking based on the authors' experiences of implementing the specification. Implementation work and performance analysis will continue to be feedback into the review process in order to maintain a practical and workable specification.

REFERENCES

[1] Object Management Group, "Intelligent Networking with CORBA," OMG DTC Document: telecom/96-12-02, December, 1996

[2] Object Management Group, "White Paper on CORBA as an Enabling Factor for Migration from IN to TINA: A P508 Perspective," OMG DTC Document: telecom/97-01-01, January, 1997

[3] Object Management Group, "Interworking Between CORBA and Intelligent Network Systems," Request for Proposal, OMG DTC Document: 97-12-06, December, 1997

[4] Object Management Group, "Interworking between CORBA and TC Systems," OMG document telecom/98-10-03, August, 1998

[5] EURESCOM Project P508, "Introduction of Distributed Computing Middleware in Intelligent Networks White Paper," OMG DTC Document: 97-09-01, September, 1997

[6] The Open Group, Preliminary Specification Inter-Domain Management: Specification Translation, X/Open Document: P509, ISBN: 1-85912-150-0

[7] ITU-T, Rec. Q.772, "Signaling System No.7 - Transaction Capabilities information element definitions," White Book, 1993

[8] Nilo Mitra, Rob Brennan, "Design of the CORBA/TC Inter-working Gateway," Proceedings 6th International Conference on Intelligence in Services and Networks, Han Zuidweg et. al. (eds.), ISBN: 3-540-65895-5 Springer-Verlag, April, 1999