

The CAMOUFLAGE Project - Introduction of TINA into Telecommunication Legacy Systems

E. Holz, O. Kath
Humboldt University Berlin
Axel-Springer-Str. 54a, 10099 Berlin, Germany
{holz|kath}@informatik.hu-berlin.de

M. Geipl, G. Lin, V. Vogel
Deutsche Telekom AG
Am Kavalleriesand 3, 64295 Darmstadt, Germany
{geipl|vogel}@fz-telekom.de

Abstract

The CAMOUFLAGE Project is a national German research initiative led by the Deutsche Telekom. It aims at the provision of a TINA conform distributed processing environment based on existing legacy telecommunication systems. The envisaged target architecture is a switched ATM network deploying B-ISDN. Using an integrative approach a mutual stimulation between the TINA technology and the B-ISDN technology is expected. The paper presents the overall design of the CAMOUFLAGE DPE, the CMA for the B-ISDN and gives a detailed description of the B-ISDN based transport network for operational interactions.

A general mechanism for the introduction of different interoperability protocols (Open Communication Interface) to CORBA is proposed. The B-ISDN protocol GFP has been selected as the foundation for the implementation for such an interoperability protocol. It allows not only a fast inter-ORB communication, but can also - due to its different modes - be tailored to the actual needs (speed, quality, amount of data).

Within the scope of the Transport Network a Connectivity Management (CyM) basing on the TINA NRA has been developed for a B-ISDN environment. Due to the fact that B-ISDN does contain already parts of the functionality required by the NRA, CyM is a subset of NRA, extended by additional objects to control the protocol stacks and the hardware.

Different scenarios for the introduction of CyM have been specified, requiring only simple or no extensions to the existing B-ISDN networks.

1. Introduction

In 1993 the TINA-Consortium has been founded with the mission to „... provide a consistent reference architecture for open telecommunications, encompassing services, operations/management architecture, and technology manage-

ment“ [11]. Today after 5 years of work most parts of the TINA framework are finished. The task is now to bring the work into practice by providing platforms supporting TINA and applications utilizing TINA. Most of the projects dedicated to these areas today are concentrating on the implementation of the SERVICE ARCHITECTURE (SA) [16] and on the development of a Distributed Processing Environment based on the OMG-CORBA ([21], [22]). One among these projects is TANGRAM [29].

While the projects do conform to the SA, many aspects of the CONNECTION MANAGEMENT ARCHITECTURE (CMA) [14] have been omitted. This is due to the following facts:

- The CMA did not take into account the architecture of existing telecommunication networks and technologies.
- CORBA 2.0/IIOP has been used by most of the projects as the baseline kernel transport network, as transport network for streams proprietary solutions have been applied.

As a result, the efficient utilization of legacy communication infrastructures is limited. The new NETWORK RESOURCE ARCHITECTURE (NRA) [13] is an advancement of the CMA, going beyond its scope and taking critics into account.

After an initial feasibility study [26] the CAMOUFLAGE project has been set up between Humboldt University Berlin, Technical University Ilmenau and Deutsche Telekom AG with the goal to:

- provide a TINA DPE [15] basing on a legacy communication infrastructure (B-ISDN),
- enable the re-use of existing software components (CORBA, TANGRAM-SA),
- develop strategies for a graceful and gradual introduction of TINA technology.

2. B-ISDN

The B-ISDN protocol suite ([1], [2]) has been developed with the goal to deploy a connection-oriented signalling protocol on top of ATM-networks. The B-ISDN signalling development enlarges the advantages of ATM (scalability and time transparency) with the concept of dynamic establishment, modification and deletion of calls and connections. Depending on the capability set, B-ISDN supports a wide range of connections (switched, permanent, semi-permanent, point-to-multipoint, point-to-point, multi-party, multi-connection) ([18], [19], [20]). In contrast to connectionless protocols as IP (Internet Protocol) B-ISDN guarantees connection characteristics for a connection from its reservation till its explicit release. Due to this guaranteed Quality-of-Service (QoS), the connection-oriented properties and the high bandwidth B-ISDN is exceptionally suited for the application in the area of multimedia.

3. CAMOUFLAGE architecture

The CAMOUFLAGE architecture reflects the integrative approach applied for the development of a B-ISDN based TINA-DPE. The whole design aims at a high degree of reuse of existing components. A CORBA-2 Object Request Broker will be used as the central building block for the DPE, and B-ISDN for the interoperability network work operational interactions and for stream interactions. The main work of the CAMOUFLAGE project concentrates on the connection of these two technologies by implementing the operational interaction transport network as a connection between CORBA islands and the implementation of a B-ISDN based CMA for the TN.

Aspects of the SA and DPE Services besides the Object Communication Support are investigated by CAMOUFLAGE at this time only marginally. For the future a combination with other projects, which address these aspects is foreseen (TANGRAM). Seen from a session oriented point of view CAMOUFLAGE can be clearly assigned to the Connectivity Session.

4. Support for operational object interaction

The principle of operational interactions in TINA is strongly related to the operation call principles defined by the OMG. Therefore the application of CORBA is favoured by CAMOUFLAGE for the operational interactions between objects within the same node and the work currently concentrates on mechanisms for the interaction between objects within different nodes. In other terms this means the provision of generic mechanisms for an inter-ORB communication.

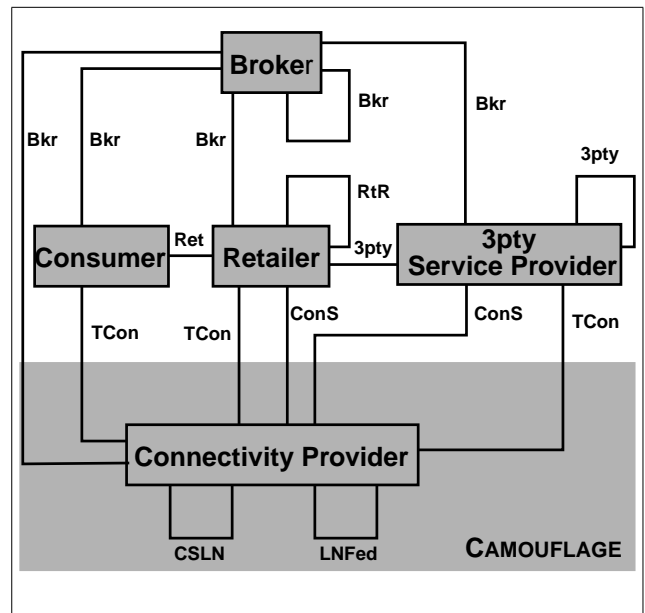


Figure 1 CAMOUFLAGE and the TINA business model

tion. For inter-ORB communication at the invocation transport level the OMG has defined the General Inter-ORB Protocol (GIOP) [21]. At this time, only the TCP/IP based realization IIOP of the GIOP is standardized.

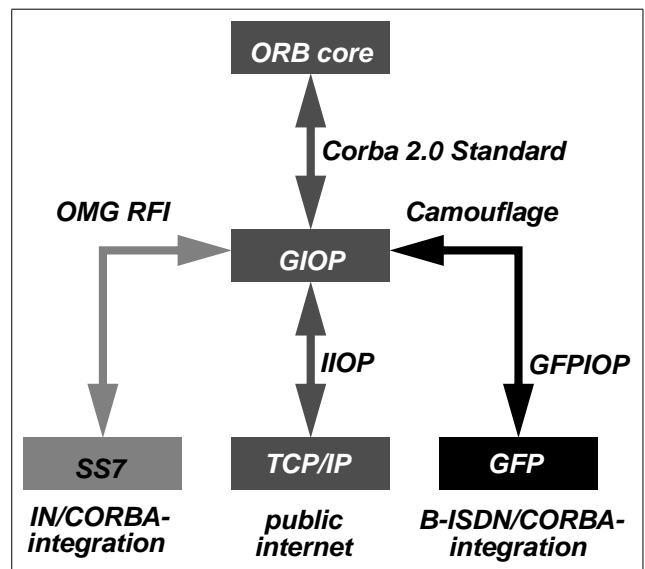


Figure 2 Inter-ORB protocols

With the OMG RFI "Issues for Intelligent Networking with CORBA" [23], which addresses CORBA/IN interworking aspects and the interconnection of CORBA islands by IN, the OMG shows one direction for the integration of CORBA and legacy telecommunication systems.

CAMOUFLAGE extends this development by widening the scope to the interconnection of CORBA islands via B-ISDN and aspects of B-ISDN/CORBA interworking. For that pur-

pose a new specialisation - GFPIOP - of the GIOP based on the B-ISDN Generic Functional Protocol (GFP) was defined (see Figure 2).

Within an ORB domain, the default ORB transport layer based on IIOP is applied (see Figure 3). For Inter-ORB

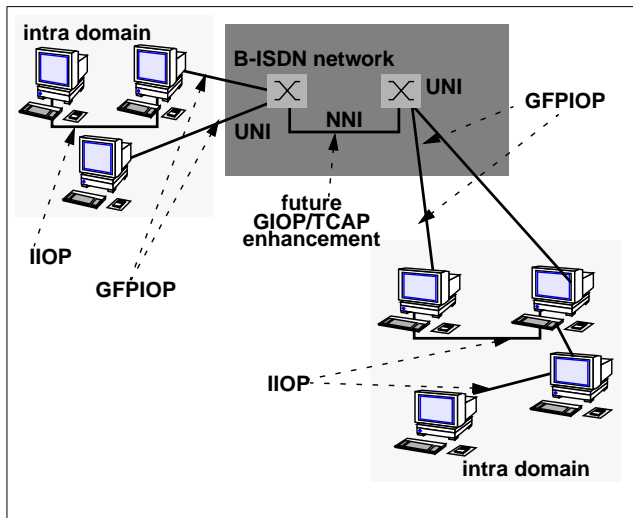


Figure 3 Usage of GFPIOP

communication via B-ISDN UNI signalling protocols, the defined GFPIOP protocol is used.

4.1. The Open Communication Interface

A direct inclusion of alternative mappings into the ORB would require for each protocol new changes to the ORB core. To be as open as possible and to enable the use of different transport protocols by the same ORB, a well defined but small interface between the ORB and the transport protocols for GIOP messages would decouple the ORB from the underlying interoperability transport network. Such an approach will also allow the dynamic addition/removal of transport protocols.

The Open Communication Interface (OCI) is defined here with the main goal is to provide a generic approach for the interconnection of DPE nodes using existing telecommunication networks and protocols. The OCI is located between GIOP and the transport protocol for GIOP messages. The functionalities of the OCI are supplied by entities within the CORBA runtime environment (interfaces `I_CorbaGiopLayer` and `I_GiopMessageQueue`) and within the transport service (interfaces `I_Transport` and `I_GiopMessageQueue`) respectively.

The `I_Transport` interface provides methods for the registration, unregistration and modification of objects, for the creation and modification of address information parts in object references (Profile Bodies) and for the establishment and release of connections representing a binding to a particular object.

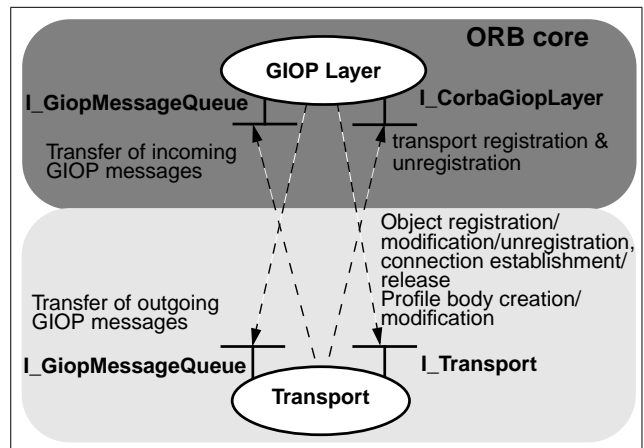


Figure 4 Interfaces and functional parts of OCI

The `I_CorbaGiopLayer` interface of the CORBA runtime environment contains operations for the registration and unregistration of a transport service. Registration in this context denotes the information of the CORBA runtime environment, that the binding of interfaces by a particular network technology is available. The `I_GiopMessageQueue` interface of the transport service stores GIOP messages to be sent to other DPE nodes and is filled up by the GIOP Layer. The `I_GiopMessageQueue` of the GIOP Layer on the other side stores incoming GIOP messages and is filled up by the transport service. The `I_GiopMessageQueue` provides additional operations to manage and store GIOP messages within a fifo data structure.

In the following sections these parts will be described in more detail, including their specifications in IDL.

a) Registration of transport services

The `registerTransport` method provided by the GIOP layer at the `I_CorbaGiopLayer` (Figure 5) allows a transport service to register itself as an entity, capable to transfer GIOP messages via connections within a particular network and to provide therefore physical interface binding representations through this network.

The transport service invokes `registerTransport` with its own reference and an indication of the supported network technology (e.g. `TAG_INTERNET_IOP` for TCP/IP) as in-parameters and receives an identification, under which this transport is registered within the CORBA runtime environment.

The unregistration of a particular transport services results in the cancellation of all interface bindings provided by the requesting transport service. The `unregisterTransport` method should only be called by the transport service in case of major network problems.

```

interface I_CorbaGiopLayer
{
// exception definitions omitted
readonly attribute
OCI::I_GiopMessageQueue MessagesToReceive;
OCI::CorbaGiopLayerId id( );
boolean registerTransport
    (in I_Transport transporter,
     in CORBA::IOP::ProfileId profileid,
     out TransportId id);
boolean unRegisterTransport
    (in TransportId id )
    raises ...;
oneway void handleMessagesToReceive ( );
};

```

Figure 5 Methods for transport registration

b) Object Registration/Modification/Unregistration

For the creation of an application object the CORBA runtime environment provides mechanisms to register this object within the system. A local unambiguous identification (object key) will be assigned to that object and an object adapter is created in order to accept bindings to that object by clients. The object adapter uses the **registerObject** method provided by **I_Transport** interfaces of one or multiple previously registered transport services to notify, that incoming GIOP messages with an identification of this object have to be sent to the referenced GIOP layer. With the **changeObject** method the assignment of a new object key for a particular object is possible.

The **unregisterObject** method should be called if an object is deleted or to release existing bindings at the network level.

c) Profile Body Creation/Comparisons

Included in the profile body an object reference contains location information. These location information may consist of e.g. an IP address and TCP port in case of IIOP, or a phone number and an object adapter identification in case of GIOP over ISDN. To create such location information entries, the operation **createProfile** of the **I_Transport** interface is used by the CORBA runtime environment. It is called with the identification of the object (object key) for which the profile body should be created, and returns a profile body for that object.

d) Connection Establishment/Release

In order to establish a binding between a client and an object implementation, the operation **connect** of **I_Transport** is invoked by the CORBA runtime environ-

```

interface I_Transport
{
// exception and attribute definitions omitted
OCI::TransportId id( );
CORBA::IOP::TaggedProfile createProfile
    (in OCI::ObjectKey object)
    raises ...;
boolean compareProfiles
    (in CORBA::IOP::TaggedProfile profile1,
     in CORBA::IOP::TaggedProfile profile2);
boolean registerObject
    (in OCI::ObjectKey key,
     in OCI::I_CorbaGiopLayer orb);
boolean changeObject
    (in OCI::ObjectKey old_key ,
     in OCI::ObjectKey new_key)
    raises ...;
boolean unregisterObject (in OCI::ObjectKey key)
    raises ...;
boolean unregisterCorbaGiopLayer
    (in OCI::CorbaGiopLayerId id)
    raises ...;
boolean connect
    (in CORBA::IOP::TaggedProfile address,
     in OCI::Duration timeout,
     out OCI::ConnectionKey conn_key)
    raises ...;
void close (in OCI::ConnectionKey conn_key)
    raises ...;
oneway void handleMessagesToSend
    (in OCI::ConnectionKey conn_key);
};

```

Figure 6 Interface specification I_Transport

ment. The implementation of this method depends on the network technology applied. In case of TCP/IP, an TCP connection between the clients host and the TCP port of the object implementation, specified within its object reference is established, while e.g. a bearer independent call is requested in case of B-ISDN usage. Parameters of the **connect** operation are address information (in terms of a profile body) indicating where to reach the object, and a timeout value of type **duration** indicating the time frame within which **connect** has to succeed. The operation terminates with a value of type **boolean** and a connection identification of type **ConnectionKey**, which unambiguously references this connection. To release a particular connection, the method **close** provided by **I_Transport** has to be used.

e) GIOP Message Transfer

After the establishment of a connection, the exchange of GIOP messages between client and object implementation is possible. If a client requests the execution of an operation by a server object, a GIOP request message and message

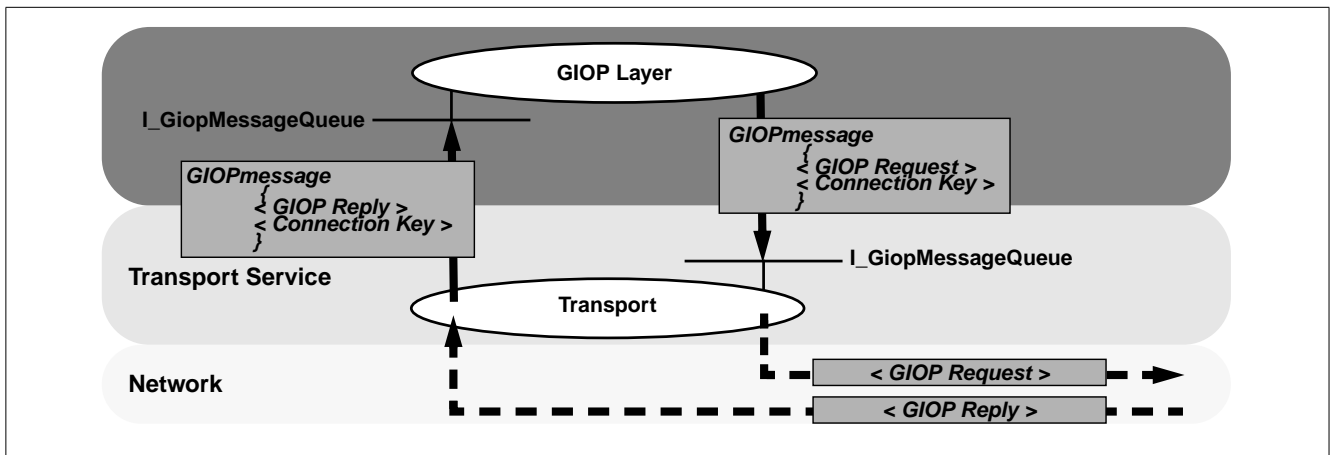


Figure 7 GIOP message flow at the invoking side

body including the object identification of the server object, the name of the requested operation, an invocation id and the operation invocation parameters have to be transmitted from the client side to the server (Figure 7).

To initiate the transmission of a GIOP Request message, the CORBA runtime environment uses the method `put` (Figure 8) to append a `GIOPmessage` to the `I_GiopMessageQueue`¹. Such a message consists of a `Request` structure and a connection identification of type `ConnectionKey`. The `Request` structure itself contains the GIOP Message Header, the GIOP Request Header, a timeout value of type `duration` and the parameters of the requested operation. Finally, the method `handleMessagesToSent` of `I_Transport` is called to start the actual transmission. Alternatively, the `put` operation may raise an event, resulting in the start of the transmission of that GIOP message by the transport service through the connection referred in the message.

At the object implementation side, the GIOP Request message is received. Because objects are registered previously through the `registerObject` method of `I_Transport`, the appropriate `I_CorbaGiopLayer` and therefore the appropriate `I_GiopMessageQueue` can be determined by the transport service. The `GIOPmessage` is put into the `I_GiopMessageQueue` using the `put` operation and subsequently evaluated by the object adapter, resulting in the invocation of the requested operation.

If the operation is an interrogation, the its result of it is transmitted back to the client using the `put` operation of `I_GiopMessageQueue` with a `GIOPmessage` consisting of a `Reply` structure and the connection identification of the transport service. The `Reply` structure itself contains a GIOP Message Header, a GIOP Reply Header, the Reply body (containing the termination parameters). These com-

```
interface I_GiopMessageQueue
{
    exception queueisempty {};
    exception queueoverflow {};
    boolean isempty ();
    void setSize(in unsigned long size);
    boolean reset();
    boolean put( in OCI::GIOPmessage message )
        raises ...;
    boolean get ( out GIOPmessage message )
        raises ...;
};
```

Figure 8 Message transfer using a `I_GiopMessageQueue`

ponents are transmitted back to the client via the identified connection and are queued in `I_GiopMessageQueue` of the CORBA runtime environment within the clients domain.

The same mechanism as used for GIOP Request and Reply will be applied for the remaining GIOP messages.

While the `put` operation of `I_GiopMessageQueue` is used to append a `GIOPmessage` to the message queue, the `get` operation removes the first `GIOPmessage` from it. With the `isempty` method provided by `I_GiopMessageQueue` it is possible to check, whether the queue is empty or not. The `reset` operation removes all entries from a queue and destroys them. With the `setSize` method the maximum number of possible entries can be set. This operation has been defined for optimization purposes.

1. Note, that the queue for GIOP messages to be sent to the network is under control of the transport service, while the queue for GIOP messages received from the network is under control of the GIOP Layer.

4.2. The GFP inter-ORB protocol

a) The General Functional Protocol

The ITU-T Recommendation Q.2932.1 [2] defines the GENERIC FUNCTIONAL PROTOCOL (GFP) as a common mechanism for the exchange of information between B-ISDN-applications at the User Network Interface (UNI). The procedures defined by the GFP do support additional basic call characteristics, supplementary services and other functions independently from or in association with an existing call/bearer. Reflecting this, the GFP offers its users three different transport mechanisms:

- bearer-related,
- connectionless bearer-independent (CLBI), and
- connection-oriented, bearer-independent transport (COBI).

The CLBI and COBI transport mechanisms fit well in the operational inter-object communication scenarios of the TINA architecture. COBI requires the establishment of a signalling ATM adaptation layer (SAAL) connection and an assured transport association between a client and a server signalling application entity. Since COBI initiates a call establishment, a unique call reference number (i.e. without bearer connection) is used as a means to associate unambiguously among the related protocol messages. In contrast with COBI, the CLBI mechanism does not need a transport association between the signalling applications. Compared with COBI, the CLBI capability will be easier to implement, and is expected to be faster concerning the performance in an environment where a large number of TINA objects are involved.

At the network node interface (NNI), the function provided by the GFP at the UNI must be mapped into an equivalent function in the network signalling system. The Transaction Capabilities Application Part (TCAP) protocol of the Signalling System No.7 turns out to be very suitable for that purpose. Within the network several routing strategies can apply, e.g. the Signalling Connection Control Part (SCCP) or the Message Transfer Part (MTP) routing, which depends on the kind of addressing.

In [1] the GFP is specified as a realization of the Remote Operations Service Element (ROSE), according to the ITU-T recommendations X.219 [8] and X.880 [9]. ROSE supports interactive applications in a distributed open systems environment. In this context, the GFP provides a means to exchange ROSE Application Protocol Data Units (APDU) on behalf of signalling applications in peer entities. To carry these APDUs, the GFP uses a special kind of Information Element (IE) in the signalling protocol messages, the Facility IE. The basic principles of remote operations are the following: A remote operation is requested by an invoking entity, and the invoked entity attempts to perform

the operation and reports the outcome. The supported protocol procedures for remote operations are:

- invocation,
- return result,
- return error and
- reject.

b) GIOP Mapping onto GFP

In order to use GFP as transport protocol for GIOP messages a mapping of GIOP onto GFP has been defined by CAMOUFLAGE. A precondition for objects to be reached over GFPIOP is an additional tagged profile to their IOR. This profile contains location and routing information, such as B-ISDN phone number, interface reference coded by subaddresses etc. Other information concern the binding characteristics (transport mode, QoS). A simplified IDL specification is given in Figure 9.

```
module GFPIOP
{ //.....
enum GFPTransport {
    NotSpecified,
    ConnectionLessBearerIndependent,
    ConnectionOrientedBearerIndependent,
    BearerRelated};
struct BindingCharacteristics {
    GFPTransport transport;
    // some more QoS characteristics
};
enum TypeOfNumber {
    unknownTypeOfNumber,
    internationalNumber,
    nationalNumber,
    networkSpecificNumber,
    subscriberNumber,
    abbreviatedNumber};
enum NumberingPlanIdentification {
    /* according to national and international
    standards like ITU-T E.164 */ };
struct PhoneNumber {
    TypeOfNumber numbertype;
    NumberingPlanIdentification numberingPlan;
    string numberDigits;};
struct ProfileBody {
    Version gfpiop_version;
    PhoneNumber phonenumber;
    BindingCharacteristic binding;
    unsigned short boaid;
    sequence<octet> object_key; };
};
```

Figure 9 GFPIOP profile

Operation invocations and terminations, i.e. the appropriate GIOP messages, are mapped onto GFP Facility IEs and transmitted by GFP according to the information of the profile. Due to the fact that GFP provides different transport modes, one of these modes has to be selected for the trans-

mission. This may be done within the profile body (binding characteristics). However, the main problem here was the development of a method for dynamic selection of one of the GFP transport mechanisms, if no preference is given.

Due to its runtime efficiency, the default transport mechanism will be connectionless bearer-independent (CLBI). In contrast to IIOP, no advance connection setup for the invocation of operations is needed. Furthermore, no instantiation of call handler state machines within the GFP protocol stack is necessary (see Figure 10).

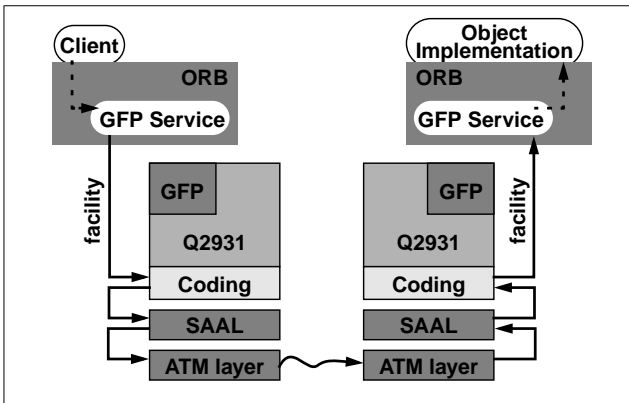


Figure 10 Control flow CLBI transport mode

The connection-oriented bearer-independent (COBI) transport mechanism is used for information exchange secured by protocol entities. This concerns e.g. explicit binding of operational interfaces and interactions between objects managing stream flow connections in the TN. For the COBI transport mode, the instantiation of call handler state machines is mandatory. The Facility IE of the initial COBI-setup protocol message contains the operation invocation request, implying at arrival time of a setup message at the called side the operation invocation. Furthermore, a GFP call will be set up. The termination of the operation and further operation invocations are transmitted by facility protocol messages within that call. On explicit request (e.g.

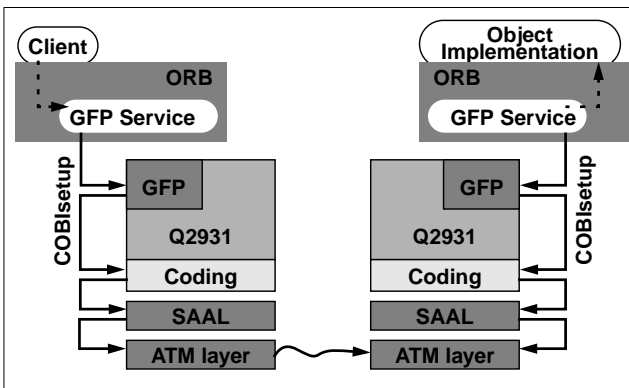


Figure 11 Control flow COBI transport mode

GIOP CloseConnection message) or at the deletion time of

of the participating objects, the interface binding and therefore the COBI call will be deleted.

In case of large operation parameters the bearer related transport mode is selected (see Figure 12), which ensures

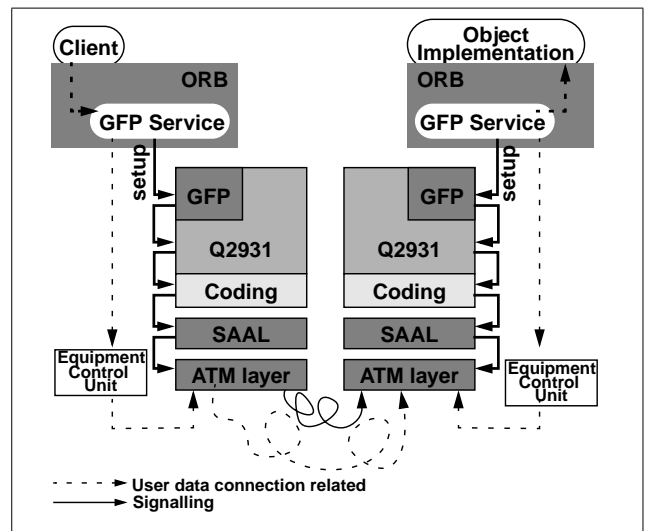


Figure 12 Control flow bearer related mode

the fast and secure transmission of these parameters. An example for this case is a pure operational implementation of an ftp service: The invocation of the *get*-operation will result in an instantiation of a user connection. The operation parameters (i.e. the requested files) are submitted over this connection, while the GIOP message and request/reply headers are transported via the facility IE of the appropriate protocol signals.

c) Technology aspects

The Camouflage project has prototypically implemented the GFPIOP as well as its related protocol stack entities. As ORB OMNIBROKER was used. Its source code was enhanced with the aim to

- decide, whether IIOP or GFPIOP will be used for a concrete binding,
- interact with objects, which implement the GIOP to GFP mapping and communication between the ORB and the GFP protocol stack (GFP service).

The communication between GFP service within the OMNIBROKER ORB and the GFP protocol stack is based on SDL signals with ASN.1 data parameters.

On the bottom of the protocol stack, objects implemented within a CORBA environment are responsible to control and manage the ATM hardware. These objects communicate with the SAAL protocol entity in terms of SDL signals. The GFPIOP protocol stack entities and the kind of communication between these entities is shown in Figure 13.

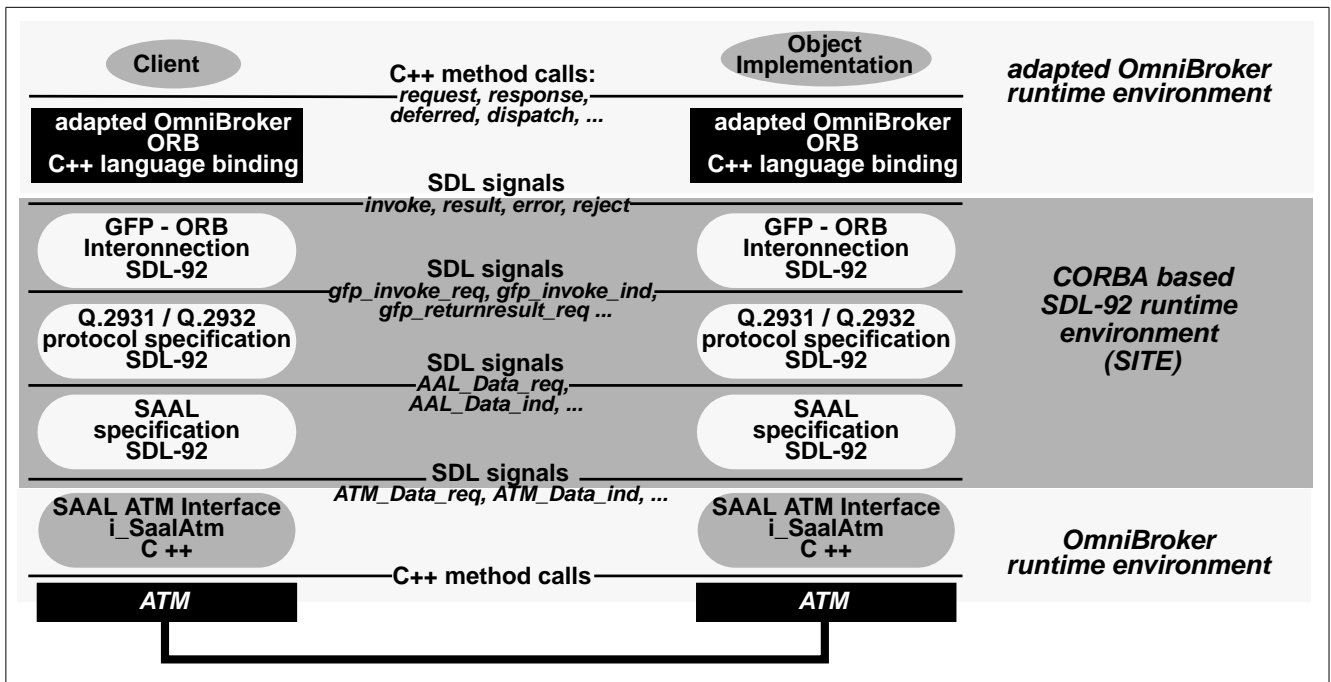


Figure 13 GFPIOP protocol stack and entities and interfaces

Some object services, like naming and object life cycle service are utilized for boots-trapping and management of the platform.

The GFPIOP functionality is tested using some OMNI-BROKER examples and applications developed within the project, like a file transfer service and a print service (using the bearer related transport mechanism). The main application of GFPIOP, especially of the COBI transport mode, is the communication of objects within the CAMOUFLAGE Connectivity Management (cf. section 5.), providing the management framework for stream based object interaction.

4.3. Other transport protocols

Besides the GFP currently also other telecommunication protocols are investigated by CAMOUFLAGE for their application as interoperability transport protocol. This concerns on one side protocols for N-ISDN [4] and on the other side protocols from the area of Intelligent Networks (IN, [7]). Again a similar approach as done for GFP is feasible here, basing on the Generic Procedures of Q.932 [5] (N-ISDN) resp. on TCAP [6] (IN). Moreover, also the application of the user-to-user signalling mechanism of N-ISDN seems to be a promising candidate.

5. Stream related work - CMA implementation

The TN is an abstraction of the transport network capabilities. Bindings through the TN (stream bindings) are set up and managed by objects of the communication session on request of objects of the service session. The communication session acts again as a client to objects of the connectivity session. Within CAMOUFLAGE the work does concentrate on the connectivity session.

A general architecture for the connectivity session has been defined. The starting point was the Connection Management Architecture defined by TINA-C [13], however to exploit the capabilities provided by the B-ISDN, this framework was adapted to the selected technology. A simplified computational view on the CAMOUFLAGE connectivity management (CyM) is given in Figure 14.

As it can be seen, the Connection Performer Objects (CP) are the access points to the protocol stack. As well network as user site initiated calls are supported by the CP objects. A Tandem Connection Manager as defined in the NRA is not necessary because the ability to cross administrative borders between subnetworks is part of B-ISDN.

Two additional types of objects have been introduced to manage and control the communication hard- and software:

- ECU (Equipment Control Unit) controls the ATM switch, the ATM cards of the CPE and other special hardware components,
- NEM (Network Element Manager) controls and manages the components of the protocol.

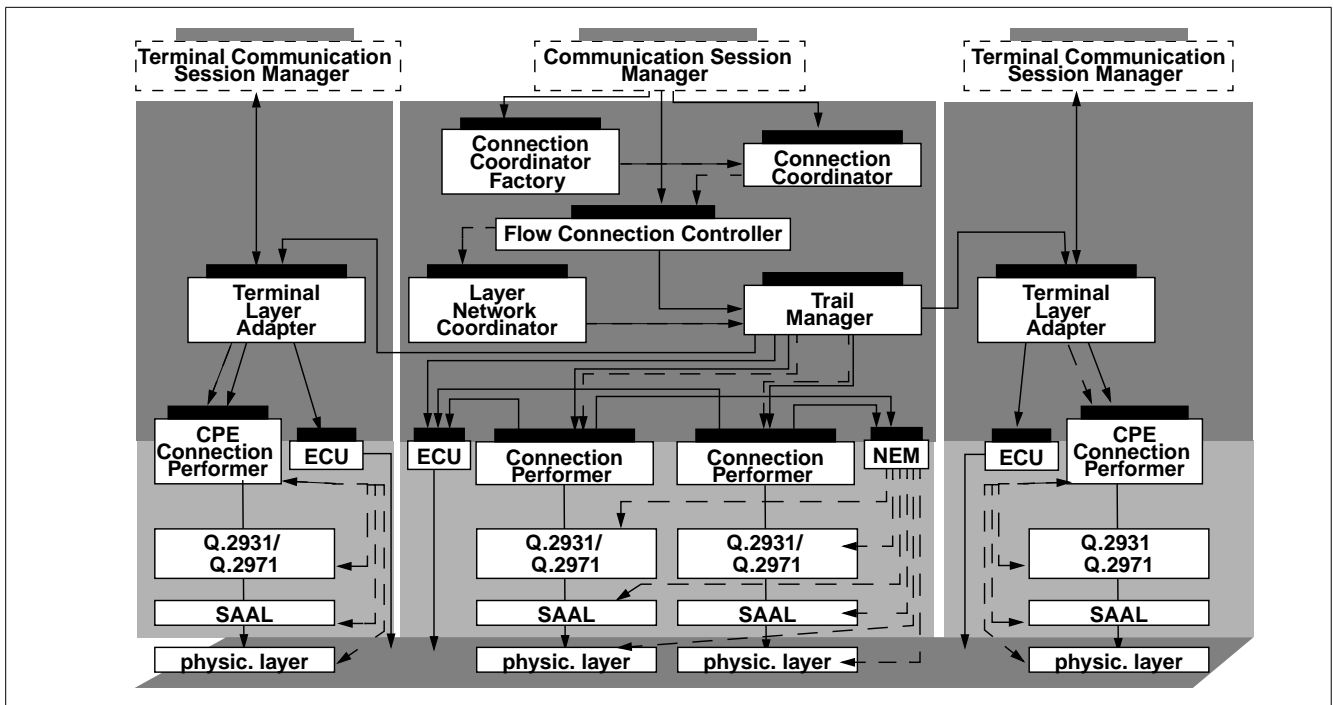


Figure 14 CAMOUFLAGE Connectivity Management (simplified)

The CP objects do mark the borderline between the two technologies applied for the implementation:

- All objects of the Connectivity session are specified using IDL resp. ODL [12] and implemented in C++.
- All protocol objects are specified using SDL-92 ([10], [17]) (as SDL-blocks) and then automatically translated to C++.

Additionally the CP specifications serve as ODL-wrappers for the protocol objects, to integrate the protocol into the overall architecture.

5.1. Research Status and Availability of Products

The concept of an open communication interface is studied within the project in cooperation with Object Oriented Concepts Inc. While Object Oriented Concepts Inc. provides the ORB implementation part, the CAMOUFLAGE/SIGTINAL project develops prototypes of transport services for the B-ISDN, N-ISDN and in future also for IN. The protocol parts have been specified in SDL and tested using the SITE SDL simulator [24]. All SDL components are automatically translated with SITE to C++, which is also the programming language for non-SDL components.

6. Conclusion

An integrative approach to introduce TINA ensures the investments of the telecommunication industry in their networks and communication software by keeping on the same

time pace with the new developments in communication and information technology. The CAMOUFLAGE project does cater this need by combining the advantages of TINA with the advantages of B-ISDN to enable interworking and interoperability. A set of deployment scenarios for the gradual introduction of the CAMOUFLAGE architecture has been defined enabling the Deutsche Telekom a step-wise transition to TINA. These scenarios range from a deployment without any changes to the network up to the explicit integration of DPE and CMA functionality into the public B-ISDN network.

Currently CAMOUFLAGE is in its prototype phase, the next tasks will be the test under real-life conditions on a ATM platform and the integration of CAMOUFLAGE and TANGRAM. CAMOUFLAGE cooperates with other european projects within the EURESCOM Project P715 and participates in the TINA-DPE working group.

7. Acknowledgments

CAMOUFLAGE is the central part of the research project SIGTINAL of the Deutsche Telekom AG and is carried out in cooperation with Technical University Ilmenau. We would like to thank our colleagues at these institutions for their support in preparing this paper. We would also like to thank Mr. Anastasius Gavras (TINA-Core Team) and Mr. Mark Laukien (Object Oriented Concepts Inc.) for the helpful discussions with them.

8. References

- [1] ITU-T Rec. Q.2931: B-ISDN. DSS2. User-Network-Interface (UNI). Layer 3 Specification for basic Call/Connection control, ITU-T '95
- [2] ITU-T Rec. Q.2932.1: B-ISDN, Digital Subscriber Signalling System No.2 (DSS2), Generic Functional Protocol. Core Functions., Geneva, August '96
- [3] Draft ITU-T Rec. Q.2932.2: B-ISDN, Digital Subscriber Signalling System No.2 (DSS2). Generic Functional Protocol. Additional Constructs, Temp. Document, Miyazaki, Februar '96
- [4] ITU-T Rec. Q.931: Digital Subscriber Signalling System No. 1 (DSS 1) - ISDN User-Network Interface Layer 3 Specification for Basic Call Control, ITU-T 1993
- [5] ITU-T Rec. Q.932: Digital Subscriber Signalling System No. 1 (DSS 1) - Generic Procedures for the Control of ISDN Supplementary Services, ITU-T 1993
- [6] ITU-T Rec. Q.771: Signalling System No. 7 - Functional Description of Transaction Capabilities
- [7] ITU-T Rec. Q.1211: Introduction to Intelligent Network Capability Set 1, ITU-T 1993
- [8] ITU-T Recommendation X.219. Remote Operations. Model, Notation and Service Definition. Geneva, 1988.
- [9] ITU-T Recommendation X.880. Data Networks and Open System Communications. OSI Applications. Remote Operations. Geneva, 1994.
- [10] ITU-T Rec. Z.100: CCITT Specification and Description Language (SDL), ITU-T '92
- [11] TINA-C „Press release 13. February 1995“, TINA-C, 1995, <http://www.tinac.com>
- [12] TINA-C „Object Definition Language - Manual Version 2.3“, TINA-C, 1996
- [13] TINA-C „Network Resource Architecture, Version 3.0“, TINA-C, 1997
- [14] TINA-C „Connection Management Specifications“, TINA-C, 1995
- [15] TINA-C „DPE Architecture“, TINA-C, 1994
- [16] TINA-C „Definition of Service Architecture, V.5.0“, TINA-C, 1997
- [17] ETSI Project Team 87: „Formal specification (SDL) for B-ISDN DSS2 CS2“, ETSI 1996
- [18] RACE II MAGIC Deliverable 6: „Stage 2 Specification“, September '93
- [19] RACE II MAGIC Deliverable 10: „Protocols and Concepts of B-ISDN Signalling“, Oktober '93
- [20] RACE II MAGIC Deliverable 3: „Service Description Framework and B-ISDN Service Description“, Oktober '93
- [21] OMG: „Common Object Request Broker: Architecture and Specification“ (CORBA 2.0/IIOP), OMG Technical Document Formal/97-02-2
- [22] OMG: „Common Object Request Broker: Architecture and Specification“ (CORBA 2.1/IIOP), OMG 09/1997
- [23] OMG: „Telecom Domain Task Force - Request for Information“, OMG DTC Document telecom/96-12-02
- [24] H. Böhme, M. v. Löwis, R. Schröder: „SDL Integrated Tool Environment“, <http://site.informatik.hu-berlin.de>
- [25] Object Oriented Concepts: „Omni Broker“, <http://www.ooc.com>
- [26] E. Holz, O. Kath: „Abschlußbericht Einsatz von B-ISDN-Netzen im Rahmen einer TINA-konformen Kommunikationssplattform“ Project CAMOUFLAGE/S, Humboldt University Berlin, 11/96
- [27] O. Kath: „TINA-C und B-ISDN-Netze“, Master Thesis, Humboldt-University Berlin '96
- [28] O. Kath, E. Holz, J. Fischer, M. Geipl, V. Vogel: „Provision of TINA Object Interaction through B-ISDN in ATM networks“, to appear GLOBECOM-97
- [29] K.-P. Eckert, M. Festini, P. Schoo, G. Schuermann: „TAN-GRAM: Development of Object-Oriented Frameworks for TINA-C Based Multimedia Telecommunication Applications“, Proc. of ISDAS '97, Berlin