Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (0223) 323010**
INTERNATIONAL: **+44 223 323010**
FAX: **+44 223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

Inter-operability and Interception
(TC presentation June 94)

Yigal Hoffner

Abstract

Inter-operability between different distributed system platforms is essential if such systems are to become commercially viable. The problem of inter-operability is non-trivial and taking the wrong decisions at this early stage could cost time, money and reputation.

In order to be able to address this problem in a cost-effective manner over the coming years it is necessary to approach the problem in a structured manner. This presentation outlines such an approach.

This approach has two major advantages: it will provides a structured way to tackle future problems associated with inter-operability which cannot be addressed at the moment because of their complexity. Additionally, it will provide an indication of where standardization will be cost-effective.

APM.1249.00.01 **Draft** 25 April 1995

Request for Comments (confidential to ANSA consortium for 2 years)

Distribution:
Supersedes:
Superseded by:

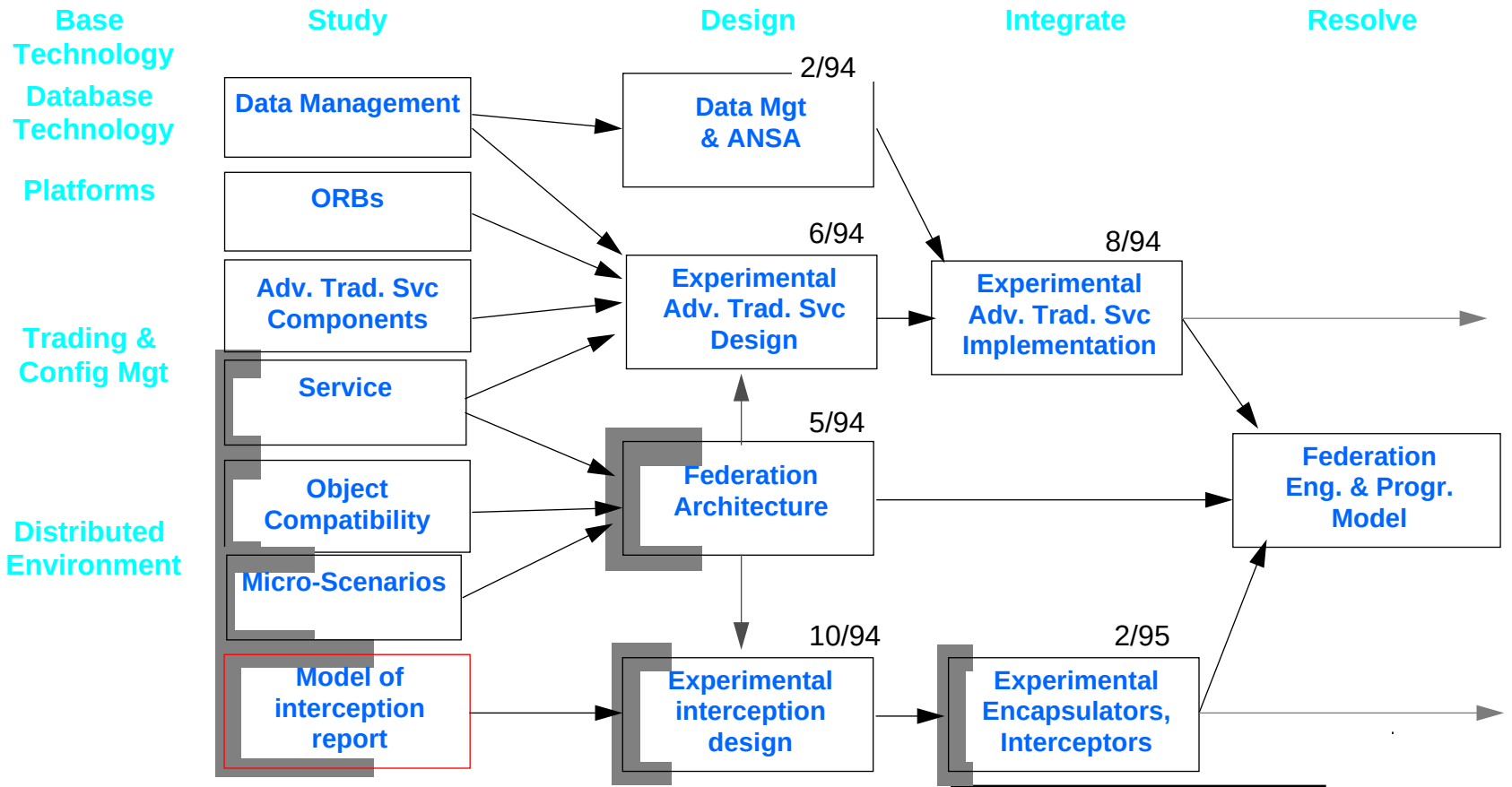
Inter-operability and Interception

Presentation to the Technical Committee Review Meeting

Yigal Hoffner
APM, Cambridge

June 1994

Federation Work



Objective of the OMG Initiative

- **Solving the inter-operability problem:**
  - **sort out the proposals (Iona/BNR/ICL/Sunsoft/(OSF/DEC/HP)/Expersoft)**

ANSA role

- **providing an architectural framework**
- **as catalyst to the process**

Objectives of the Presentation

- **An overview of the different approaches to inter-operability**
  - **Standardization and gateways approaches applied to different boundaries**
- **Outline the difficulties which lie ahead - much detail but hard to generalize**
- **Provide an ANSA perspective/structure to the problem**
- **Put forward the business case for this work**

Potential Differences Between Systems

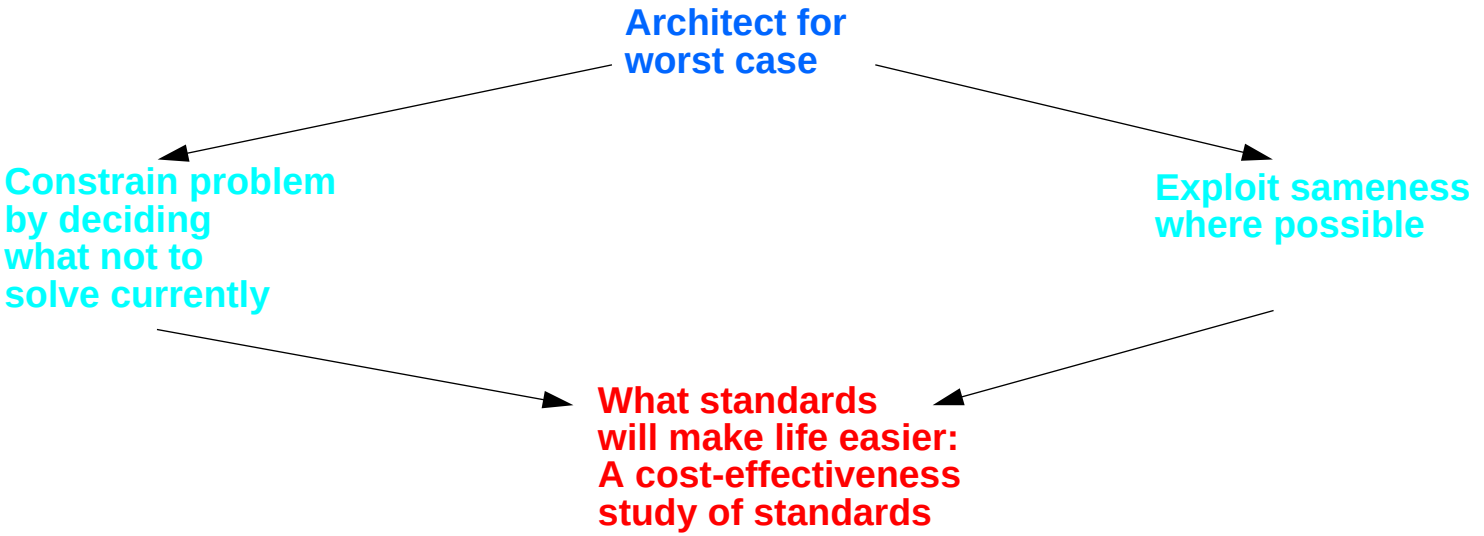
- **Structure the problem space to enable:**
  - separation of differences and dealing with them independently of each other
  - finding out how they may influence each other
  - finding out where standardization is worthwhile
- **A taxonomy of differences for practical purposes:**
  - gateways - to know how to deal with the differences
  - standardization - to know what to negotiate about in order to agree
- **A language for talking about the differences**

Potential Differences Between Systems (II)

- **Differences may be in:**
  - **type systems**
  - **concrete data types**
  - **object models**
  - **semantics**
  - **issues additional to semantics such as: management, remuneration, security, QoS, etc.**
  - **interface definitions**
  - **language for communication between the client and server**
  - **Ifrefs**

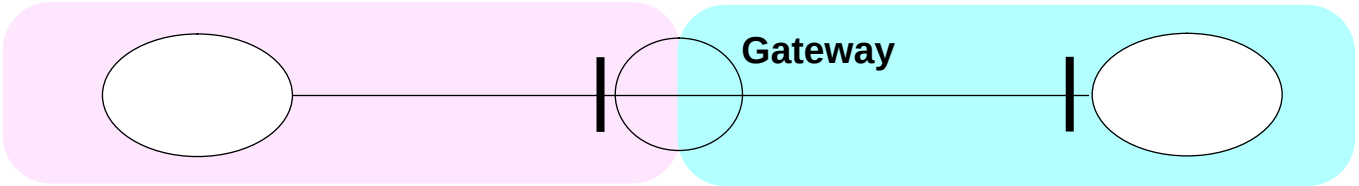
Future Proofing - the ANSA Approach

- **An architectural structure for tackling future problems**

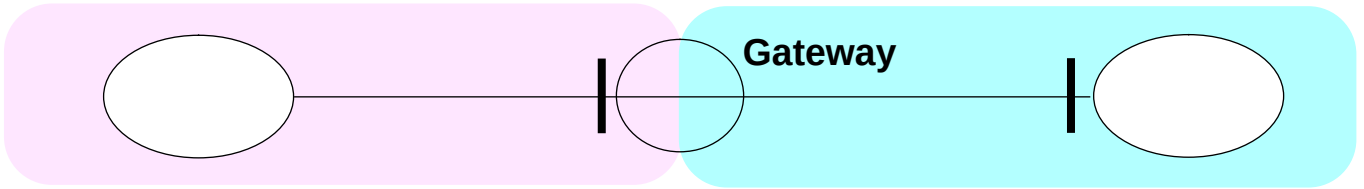


What is a Gateway

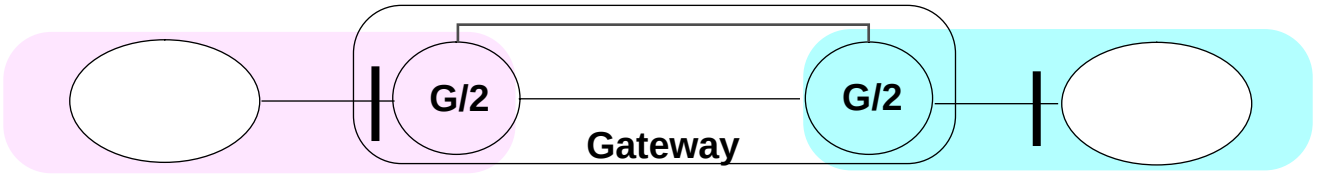
- **Model of a gateway**



What is a Gateway

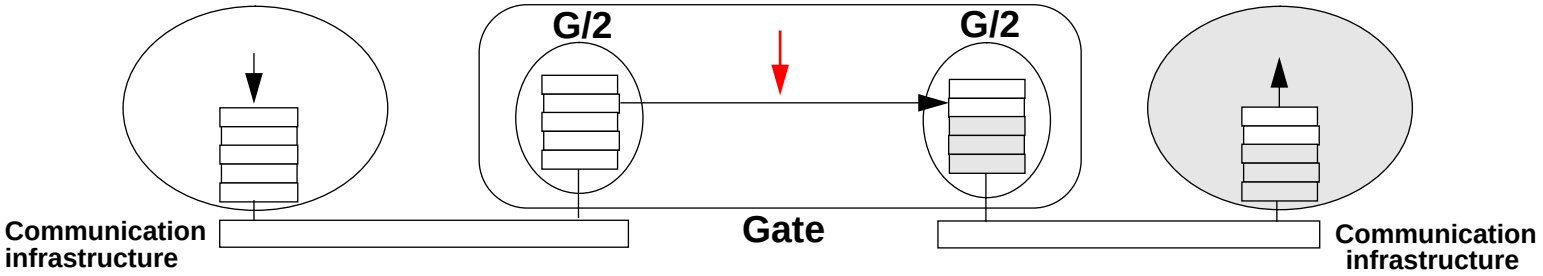


- Another view of the same gateway



- There are different ways of implementing the model
- Examples:
  - communications
  - Ifrefs

Where to Standardize?



- We know we can build gateway - where is worthwhile to standardize?
- Negotiation for standardization should be from the “top down” since gateway must translate to the level above the highest incompatibility
- Transport protocol agreement does not solve the problem
- Application level gateway - disadvantages:
  - non ORB functionality - non portability
  - losing transparency information
- Agreeing on a small number of protocols and providing them in ORB

Bridging IFREF Domains

- **Build a proxy when a boundary is crossed**
  - can be wasteful on resources if Ifrefs are not used
  - efficient on first call
- **Build a proxy when the need arises**
  - only sets proxy when invocation is made
  - requires Ifrefs to be extended when crossing a domain boundary
- **Gateway as a forwarding service**
  - forward invocations from gateway to gateway or server without building a proxy
  - requires Ifrefs to be extended when crossing a domain boundary
- **Name proxy**
  - avoids full scale proxy by creating a proxy with foreign Ifref in gateway to avoid long names
  - requires setting up the proxy when the need arises

Where to Standardize?

- **ALL the work is done in GATEWAY without requiring end-points to know anything**
  - **building proxies when crossing boundaries**
- **CO-OPERATION from END-POINTS is required**
  - **building proxies when needed**
  - **name proxies**
  - **forwarding service**
- **Co-operation requires standardization - where would it give us big returns?**
- **What strategy to adopt in what circumstances?**
  - **scaling and performance**
  - **garbage collection**
  - **circular referencing**
  - **implementation considerations - when and where to detect passing lfrefs**

The Challenge Ahead

- **Two potential differences were discussed: comms and Ifrefs**
- **We still have to deal with more complex types of boundaries:**
  - **Object incompatibility: semantics, interface definition**
  - **Remuneration, security, performance, dependability**
- **How different types of gateways may interact**
  - **how they may effect each other - does ordering matter**
  - **when implemented together for the sake of efficiency**
  - **issues of trust**
- **Implementation issues**
  - **how to make parts of the gateway without using non-ORB functionality**
  - **which gateways should be combined for optimization and how**
  - **Interface independent versus interface specific (Dynamic Skeleton Interface - DSI)**

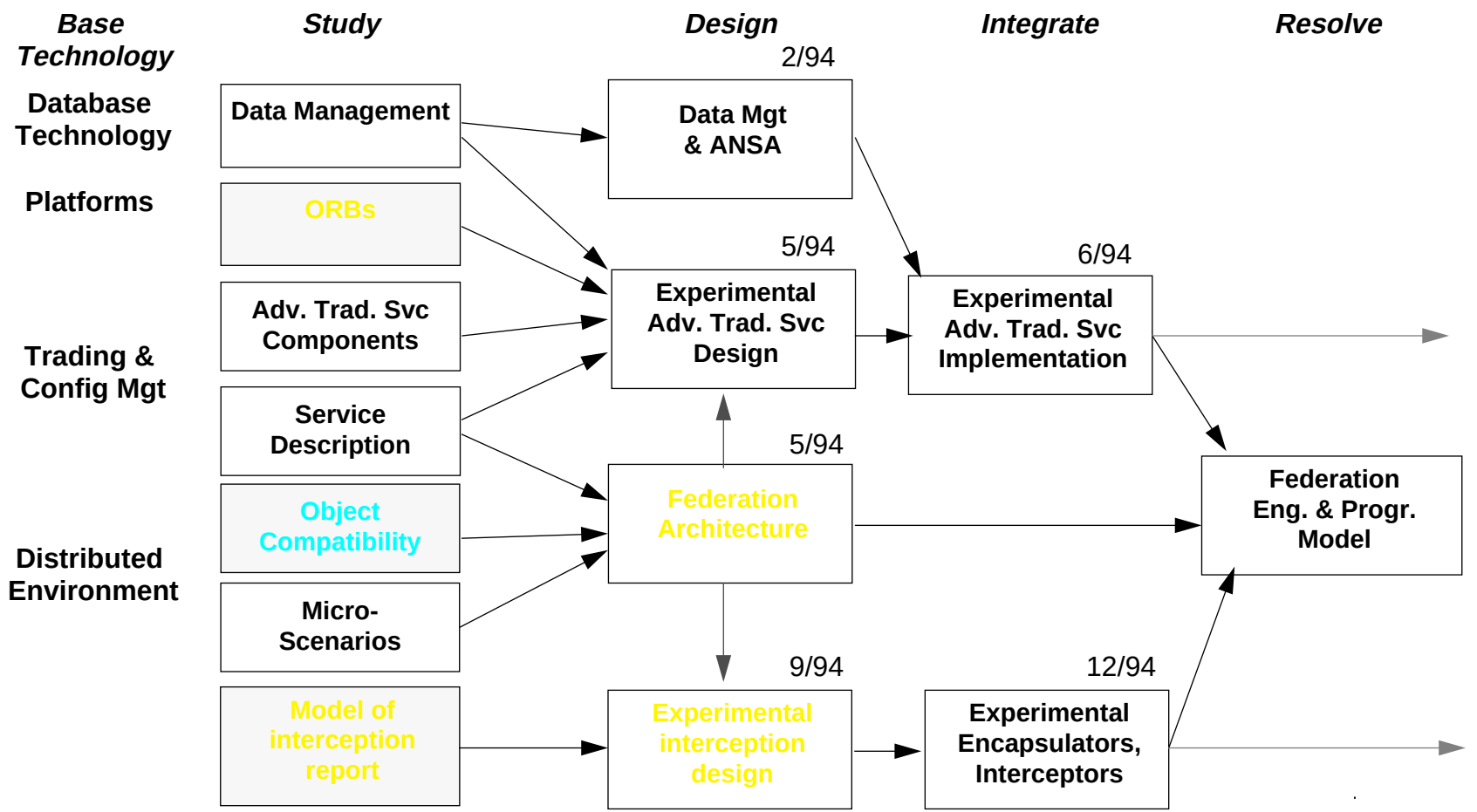
Conclusions

- **Develop a language and a taxonomy of differences to structure problem**
- **Architect for worst case**
- **Consider ways to overcome the differences by using gateways**
- **Evaluate trade-off between cost of standardizing and the effect on gateways in terms of:**
  - **find where commonality will have a strong effect on scaling, performance and implementation effort of gateways**
  - **take into account time and cost of agreeing standards**
- **Cost-benefit analysis of standardization - What is worth standardizing and in what circumstances?**
  - **long and short term - invest in tools or standards**
- **Standardization is not a sensible solutions until we have done the above!**

Inter-operability and Interception

Technical Presentation

Yigal Hoffner
APM, Cambridge
June 1994


Base Technology
Study
Design
Integrate
Resolve
Database Technology
Platforms
Trading & Config Mgt
Distributed Environment
Data Management
ORBs
Adv. Trad. Svc Components
Service Description
Object Compatibility
Micro-Scenarios
Model of interception report
2/94
Data Mgt & ANSA
5/94
Experimental Adv. Trad. Svc Design
5/94
Federation Architecture
9/94
Experimental interception design
6/94
Experimental Adv. Trad. Svc Implementation
12/94
Experimental Encapsulators, Interceptors
Federation Eng. & Progr. Model

Objectives of Inter-operability

"The ability for a client on ORB A to invoke and IDL-defined operation on an object on ORB B, where ORB A and ORB B are independently developed".

- **How to get an invocation message correctly delivered from the client object to the server object**
- **How to correctly decode the message when it arrives**
- **How to correctly interpret references to objects which appear as parameters to the invocation**

Objective of the OMG Initiative

- **Solving the inter-operability problem:**
  - **sort out the proposals (Iona/BNR/ICL/Sunsoft/(OSF/DEC/HP)/Expersoft)**
  - **ANSA role as providing an architectural framework**
  - **ANSA role as catalyst to the process**

Objectives of the Presentation

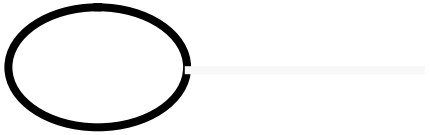
- **An overview of the different approaches to inter-operability**
- **Outline the difficulties which lie ahead - much detail but hard to generalize**
- **Provide an ANSA perspective/structure to the problem**
- **Put forward the business case for this work**

The Potential Differences Between Systems (I)

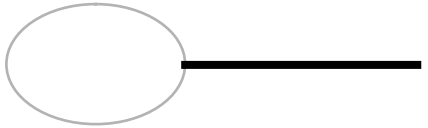
- **Structure the problem space:**
  - separate and deal with differences between systems independently
  - optimize where necessary
- **A taxonomy of differences for practical purposes:**
  - gateways - to know how to deal with the differences
  - standardization - to know what to negotiate about in order to agree
- **A language for talking about the differences**

The Potential Differences Between Systems (II)

- ***What* is it?**



- ***How* does it communicate?**



  - **How are the *What* and *How* described?** what language is used to descrivbe the two aspects
  - It is convenient to test and find as much compatibility as possible before the binding stage. The main question of the *What* is dealt with by Trading
  - Binding concerns are with the *How*

- The more done at trading the less has to be done later

The Potential Differences Between Systems(III)

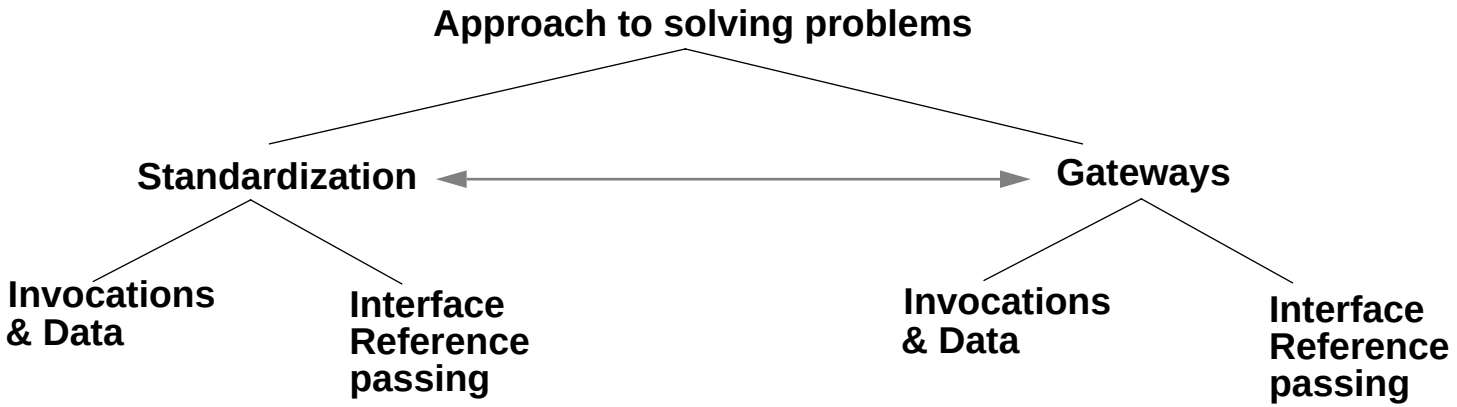
- **Differences may be:**
  - **semantics:** a definition of service offers and client requirements in a wider sense than provided by the definition of interfaces in terms of operation names and parameter types
  - **issues additional to semantics such as:** management, remuneration, security, QoS, etc.
  - **type system (ADT's):** the framework in which compatibility in computational terms can be tested
  - **IDL- concrete representation of ADT's:** a way of defining computational types
  - **interface definitions:** the description of the offered and required service
  - **Ifref's:** the content, format and representation of references to objects
  - **language for communication between the client and server:**

What data: **(1) payload: operation and parameters; (2) additional information: headers, tails, QoS, other issues**
Format of data:
Encoding of data:

- **Pre-binding concerns:**
  - **semantics, additional issues, type system, IDL, interface definition and *Ifref*'s**

- **Binding concerns:**
  - **language for communication and *Ifref*'s**

OMG RFP Solutions to Inter-operability



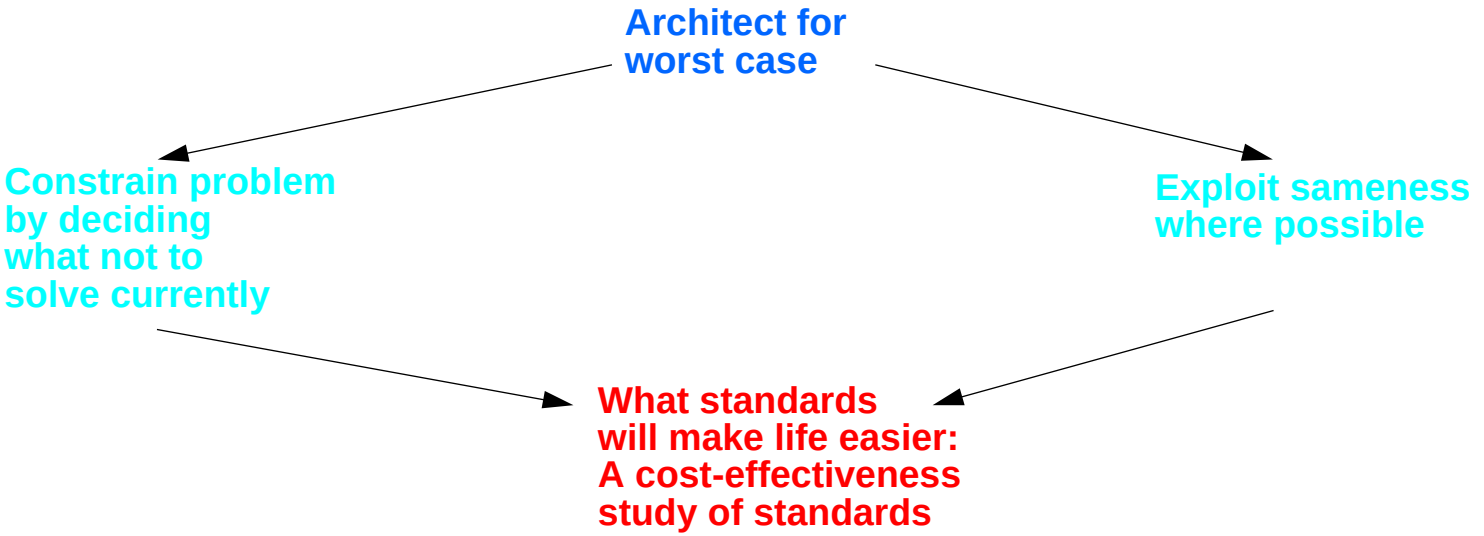
- Gateways - technical problems
- Common protocols - political problems
- Seen as a spectrum
- Arguments between the two camps as to which approach is correct

The ANSA Approach to Inter-operability

- **Analyse the WORST possible case (assume that anything that can be different may be different):**
  - **enumerate points of DIFFERENCE**
  - **STRUCTURE the space of differences - a structured solution will enable to solve the hard cases which we have not solved this time around, next time**
- **ARCHITECT to solve worst case**
- **CONSTRAIN the scope of the problem to deal with what can be currently dealt with (CORBA type system/ OMG IDL)**
- **Note EXPLICITLY where this is done**
- **Ensure appropriate HOOKS to deal with the harder problems later**
- **Ensure no DECISIONS which prevent the later removal of the constraints**
- **Provide GATEWAYS as the generic solution**
- **Exploit SAMENESS where possible (standardization)**

Future Proofing

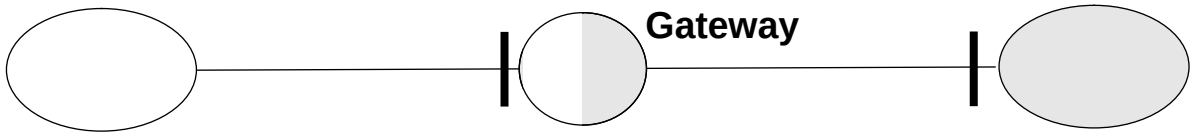
- **An architectural structure for future problems**



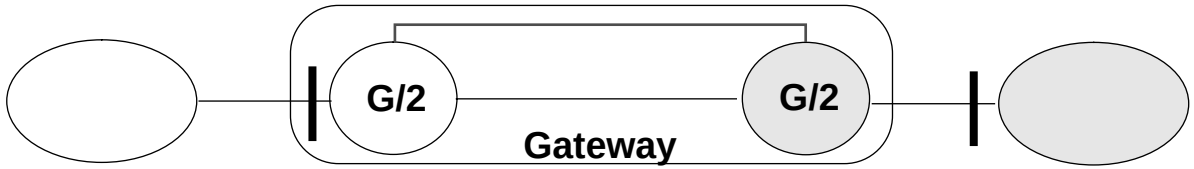
What is a Gateway

- **Model of gateways**

A gateway Invocations from one ORB to objects in the other ORB are directed to the gateway. The gateway turns around and makes the actual invocations to the target objects.

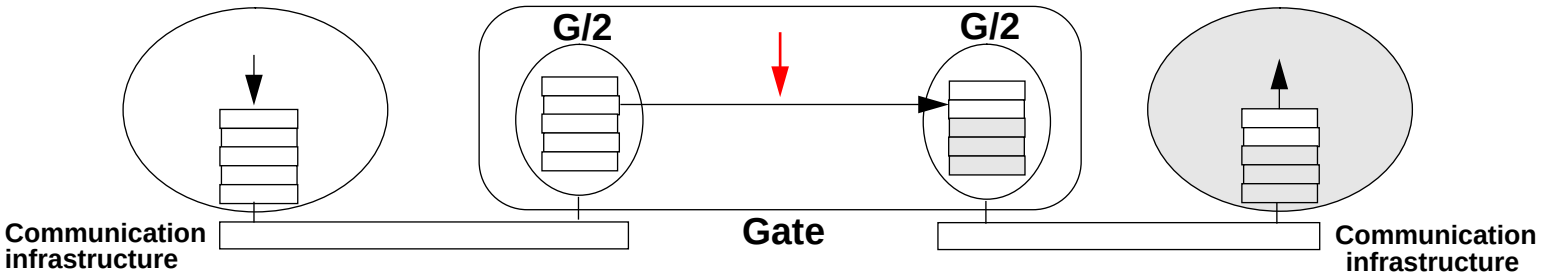


Open up the gateway - two parts each dealing with it's side of the world



- **There are different ways of implementing this - next two examples:**
  - **communications**
  - **Ifref's**

Where to Standardize?

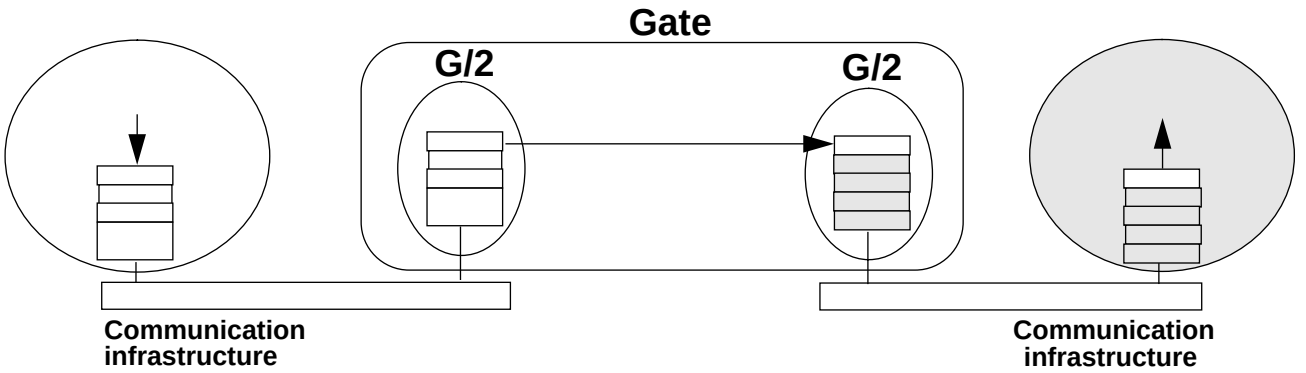


- Intermediate form is a level of compatibility between the two sides
- Negotiation should be from the “top down” - gateway must translate to the level above the highest incompatibility
  - Possible to optimize with lower compatibility - but of doubtful advantage
- Application level gateway - disadvantages:
  - non ORB functionality - non portable
  - transparency information
- If linking problem (name clash) sorted - why not put protocol stacks in objects as options and avoid gateways

Where to Standardize? (cont.)

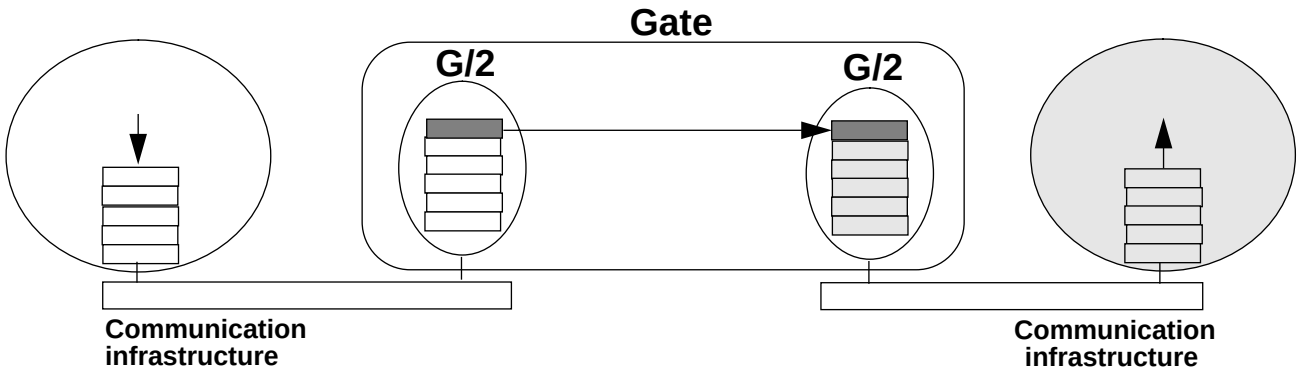
- **If linking problem (name clash) sorted - why not put protocol stacks in objects as options and avoid gateways**
  - maintenance problem - maintaining many protocol stacks is difficult
  - may require resources a process/machine cannot support for many objects
- **Possible approaches:**
  - Put 1-3 protocol stacks in each object as options (directed graph approach rather than linear layers) (2 maximum 3 - hard to maintain)
  - allow use of one set of protocols for interworking with same ORBs - an optimization
  - other stacks for interworking between different ORBs
- **Use gateways where this is not possible**
  - Push the incompatibility level as much as possible down
  - use low-level gateways where there is incompatibility
  - Use local comms but as part of ORB comms (as an optimization to remote comms)

Gateway (I) - Comms Incompatibility



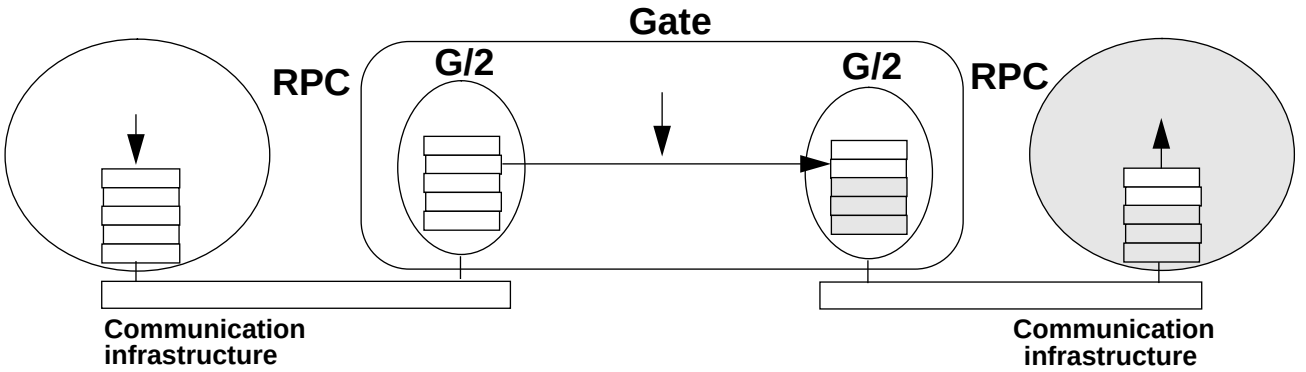
- Gateway has to do all the work
- An additional level of compatibility has to be added to gateway to reach common ground

Gateway (II) - Comms Incompatibility



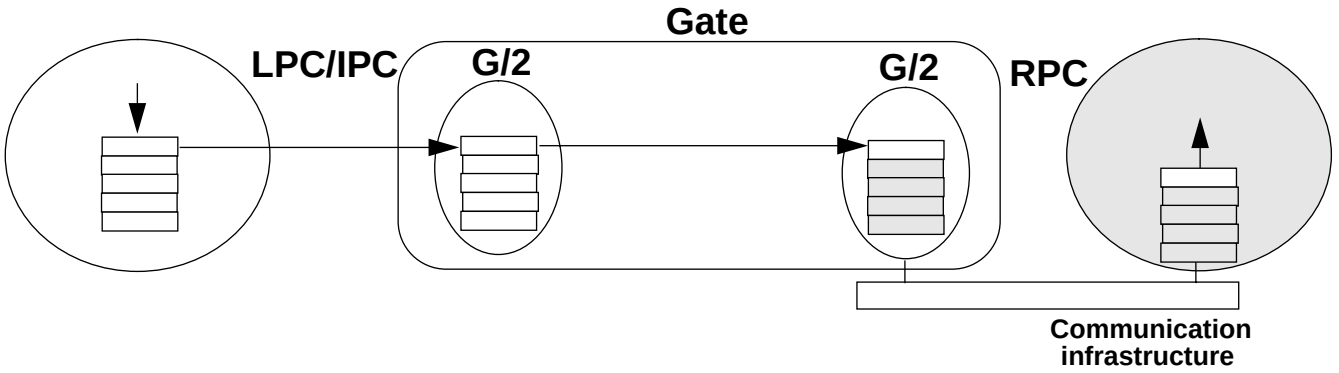
- **Having to go to a level of compatibility above comms**

Gateway (III)



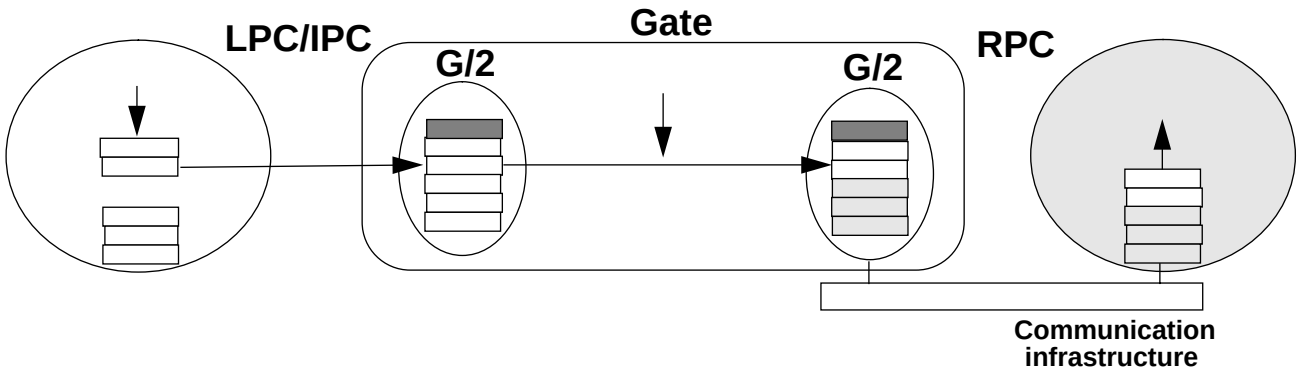
- Comms incompatibility in some levels
- We can utilize existing compatibility
- Level of translation can be pushed down
- Gateway has to do less work

Gateway Optimization (I)



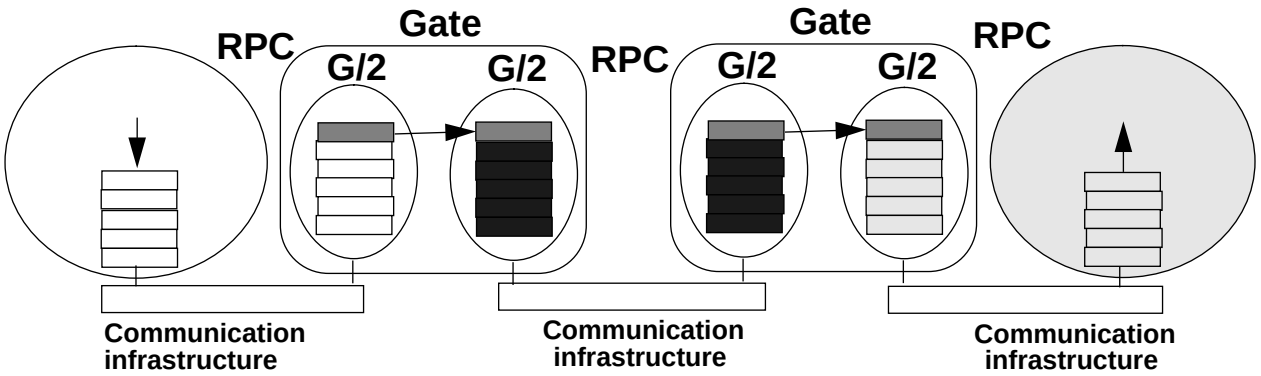
- **Employ non-ORB communications to circumvent the transformations on one side (marshalling and un-marshalling)**
- **Bypasses ORB entirely - danger of losing any transparency and QoS provided by ORBs**
- **Sometimes called application level gateway (misleading)**
- **Two ISSUES: Level of incompatibility/translation+ means of communications**

Gateway Optimization (II)



- Same can be done where partial compatibility of comms exists
- Employ local communications to circumvent SOME of the transformations on one side
- Requires less marshalling and un-marshalling work in gateway

Gateway (III)

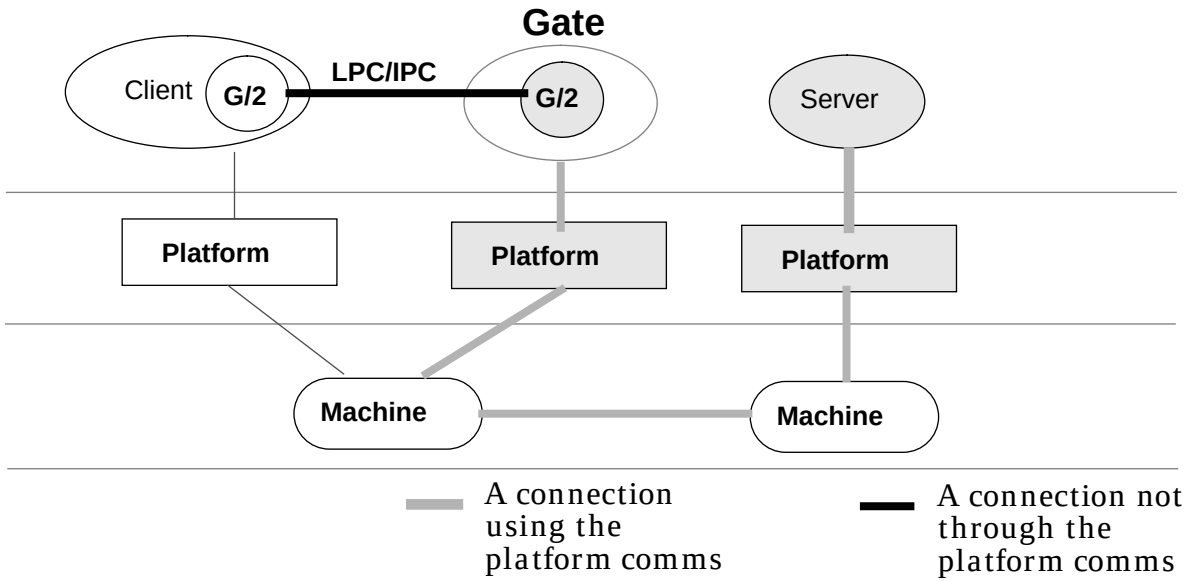


- Using a third RPC link
- Can use a different ORB from the two non-compatible ORB's

Implementation of Gateways

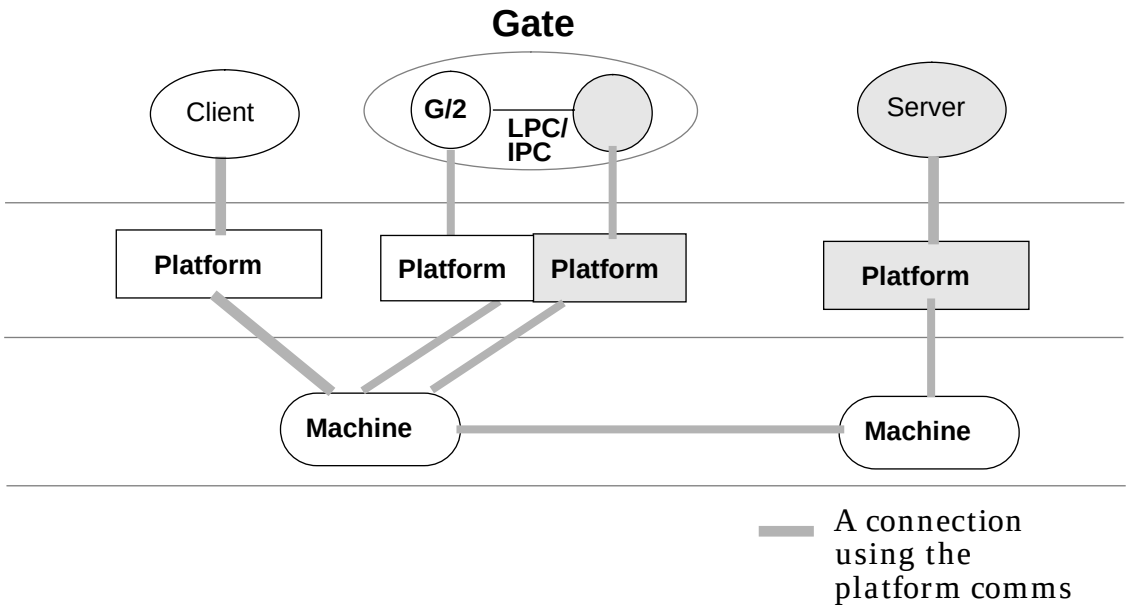
- **Reasons for different implementations:**
  - difficulties with compiling two platform into one process (name clashes)
  - resources: two platform in one process or in one machine may be to much for some OS's or machines (e.g. PC)
  - using local communications such as IPC is not easily portable in some cases
- **Questions:**
  - what facilities are required in ORB to facilitate gateways?
  - using non-ORB functionality

Implementing Gateways (option I)



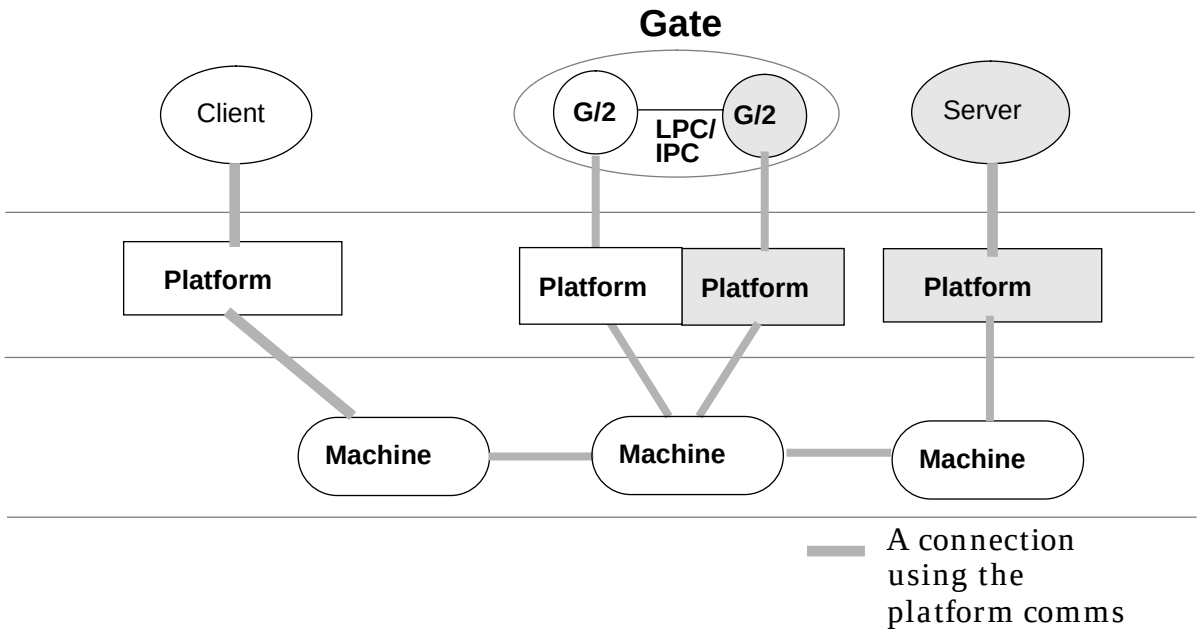
- **One platform per process**
- **Use local comms to bypass ORB comms:**
  - **Can be efficient but may be OS dependent and therefore non-portable**

Implementing Gateways (option II)



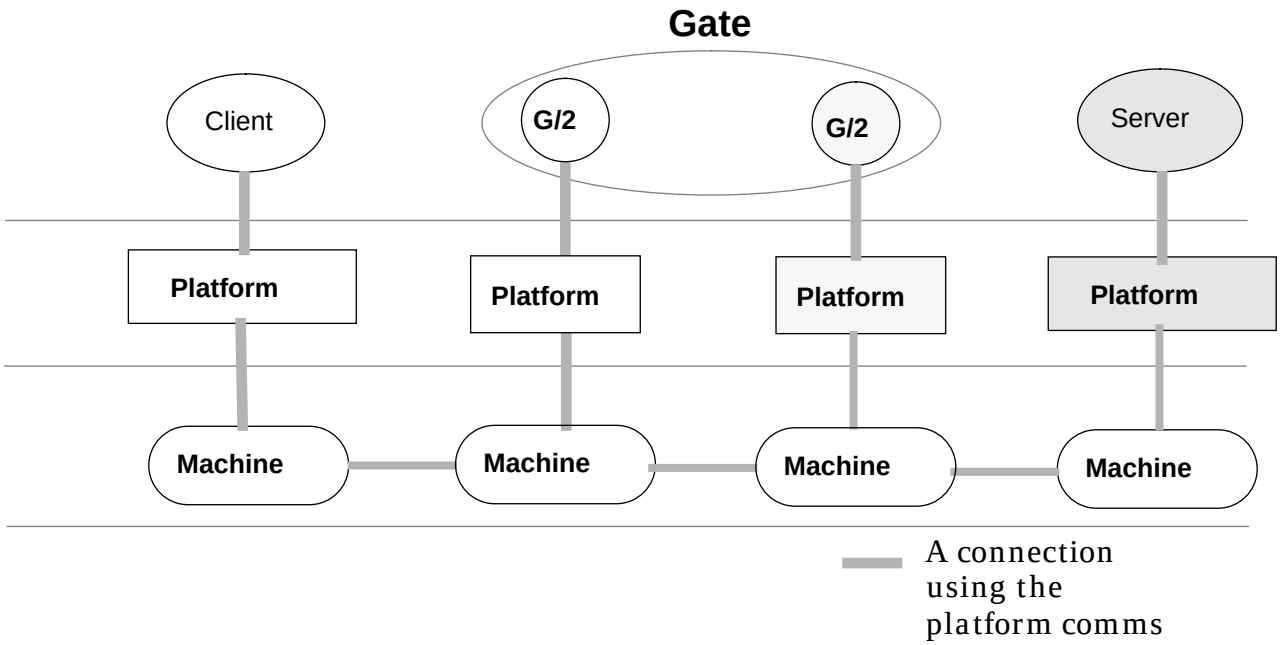
- **Two platforms in one process**
  - **single process: linking problem (name clashes)**
  - **use of ORB comms**

Implementing Gateways (option III)



- **Same as option II**
- **Two platforms in a single process**

Implementing Gateways (option IV)



- **Using a third RPC**
- **Possible to use a different platform which can support both RPC's**

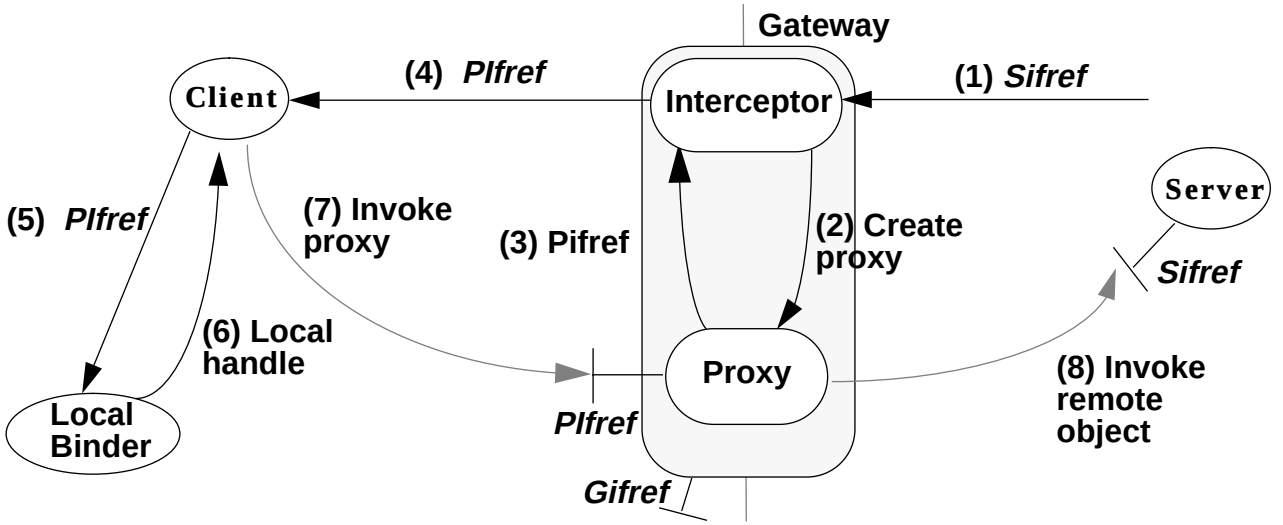
Bridging IFREF Domains

- **Main approaches:**
  - **build a proxy when a boundary is crossed:** pass local Ifref of proxy to client
  - **build a a proxy when the need arises:** marking foreign Ifref's to be distinguishable locally
  - **gateways as forwarding service**
  - **name proxy**
- **Problems:**
  - **Scaling and efficiency problem:**

late proxy making is more efficient on resources but requires more time on first invocation
late proxy requires add-ons to foreign Ifref: <l/f> <domain-name> <f-Ifref> and may get too long

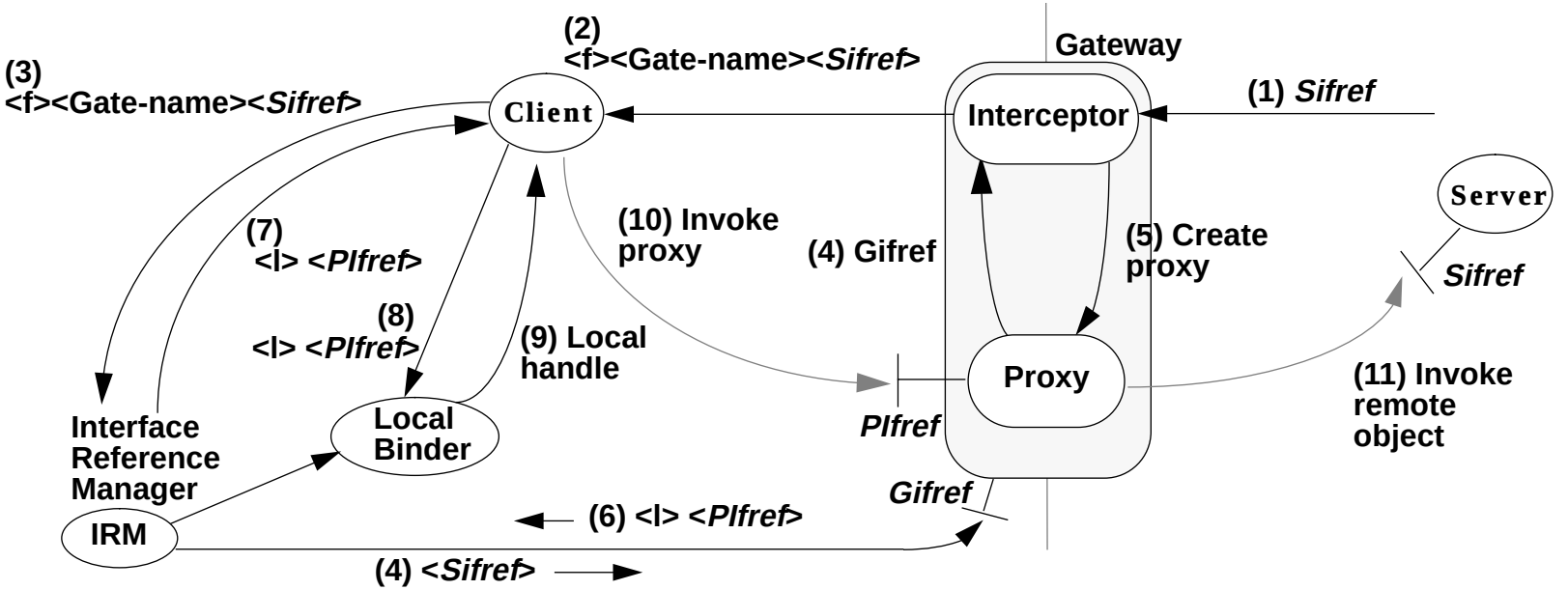
  - **Garbage collection problem**
  - **Circular referencing problem**
  - **Implementation considerations**
  - **when and where to detect passing Ifref's:** sort Ifref's when doing other translations can be an optimization. UKC tags Ifref's at gateway and use IRM at end points to sort them out

Building Proxies When Crossing Boundaries



- **Create the proxy when an Ifref crosses a boundary**
  - Can be wasteful on resources if Ifref's are not used
  - Efficient on first call

Building Proxies When Needed



Method developed at UKC. Some subtle points about the agreement necessary between the two G/2 in gateway.

- **Construct proxies when needed; use extended Ifref's locally**
- **Use IRM to translate local representations of foreign Ifref's to proxy Ifref**

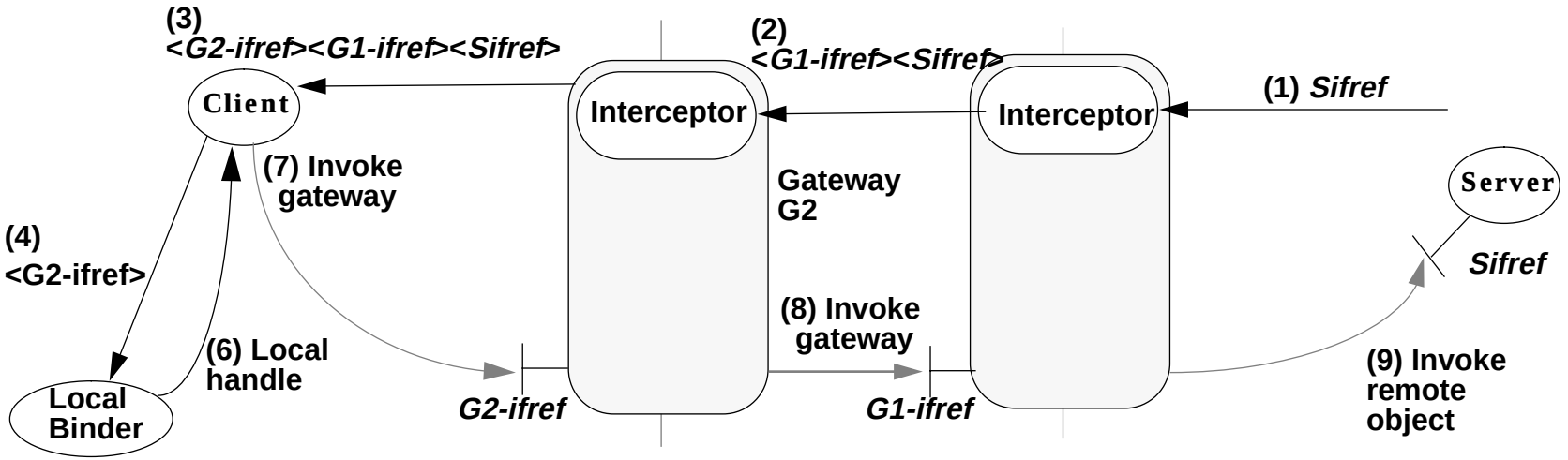
How much knowledeg of the surrounding Ifref domains is put in IRM determines how much work is done in gateway.


ANSA

Building Proxies When Needed (cont.)

- May create long Ifref's when crossing more than one boundary
- Does not waste resources where Ifref is not going to be used
- The two sides of the gate have to talk to each other to set the link
- How much knowledge of other domains is needed in IRM's

Gateways as a Forwarding Service



- Invocations are passed directly through - no construction of proxies
- A transformation is necessary on the client side of each gateway: <*G2-ifref*><*G1-ifref*><*Sifref*>$OP(paras) --><*G2-ifref*>$OP(<*G1-ifref*><*Sifref*>, paras)
- Enhancing RPC to do multi-hops we cam make gateways more scalable

Name Proxies

- Compromise solution
- Keep the foreign Ifref in Gateway and pass a local Ifref to the information
- When remote binding is to be established - get the appropriate information from the Gateway and create proxy

Options

- **Do ALL the work in GATEWAY without requiring end-points to know anything (they assume everything is local)**
  - **building proxies when crossing boundaries**
- **Require CO-OPERATION from END-POINTS (knowledge of difference between local and foreign Ifref's)**
  - **building proxies when needed**
  - **name proxies**
  - **forwarding service**
- **Relationship between where knowledge is and where the work is done - engineering trade-off**
- **What scales better in what circumstances?**
- **What investment in a standard will give best benefits?**

Assessing the Different approaches

- Scaling and efficiency in different circumstances - what scales better in what circumstances?
- passing many Ifref's
- using only a small % of Ifref's passed
- crossing many boundaries

- **Garbage collection**: how does gateway know when a proxy can be dismissed
- **Circular referencing**: recognizing references to same domain thru others - depends on IRM knowledge
- **Implementation considerations - when and where to detect passing Ifref's:** sort Ifref's when doing other translations can be an optimization. UKC tags Ifref's at gateway and use IRM at end points to sort them out

Challenge of Constructing Different Types of Gateways

- **To deal with the different types of boundaries:**
  - Comms incompatibility: language for communication
  - Ifref's
  - Object incompatibility: semantics, interface definition
  - Remuneration, security, performance, dependability
- **How to implement gateways**
  - how to make parts of the gateway communicate (single machine/single processes solutions require either non-ORB comms or changes to linking)
  - how to combine different gateways together - optimization
  - issues of trust
- **How different types of gateways may interact**: dec/encryption and comms
  - how they may effect each other - does ordering matter
  - where will standardization make a big difference
  - when implemented together for the sake of efficiency

Generic Gateways and Portability

- **Interface independent versus interface specific:**
  - interface specific: interface definition at compiled time
  - interface independent - can deal with any interface but requires:
    * send data + format/type information
    * send some identifier to look up interface definition in Interface Repository

A service specific gateway is designed to provide access to services which are known at the time the gateway is compiled; i.e. the interface definitions of the services are used to compile the gateway. A generic gateway does not need to know the interface definitions of the services in advance; it can create proxy objects of new types on the fly. In order to program a generic gateway one needs a server-side analogue to the CORBA Dynamic Invocation Interface (DII). The various proposals call this a Dynamic Server Interface or Dynamic Skeleton Interface (DSI). (This DSI also has other uses, such as supporting dynamically defined query language operations in my model of remote database access in an ODP environment.)

- **Requirement from ORB for generic gateways (interface independent):**
  - need Dynamic Skeleton Interface (DSI) in addition to Dynamic Invocation Interface (DII)
- **Portability issue: do not use any non-ORB functions to implement gateways!**

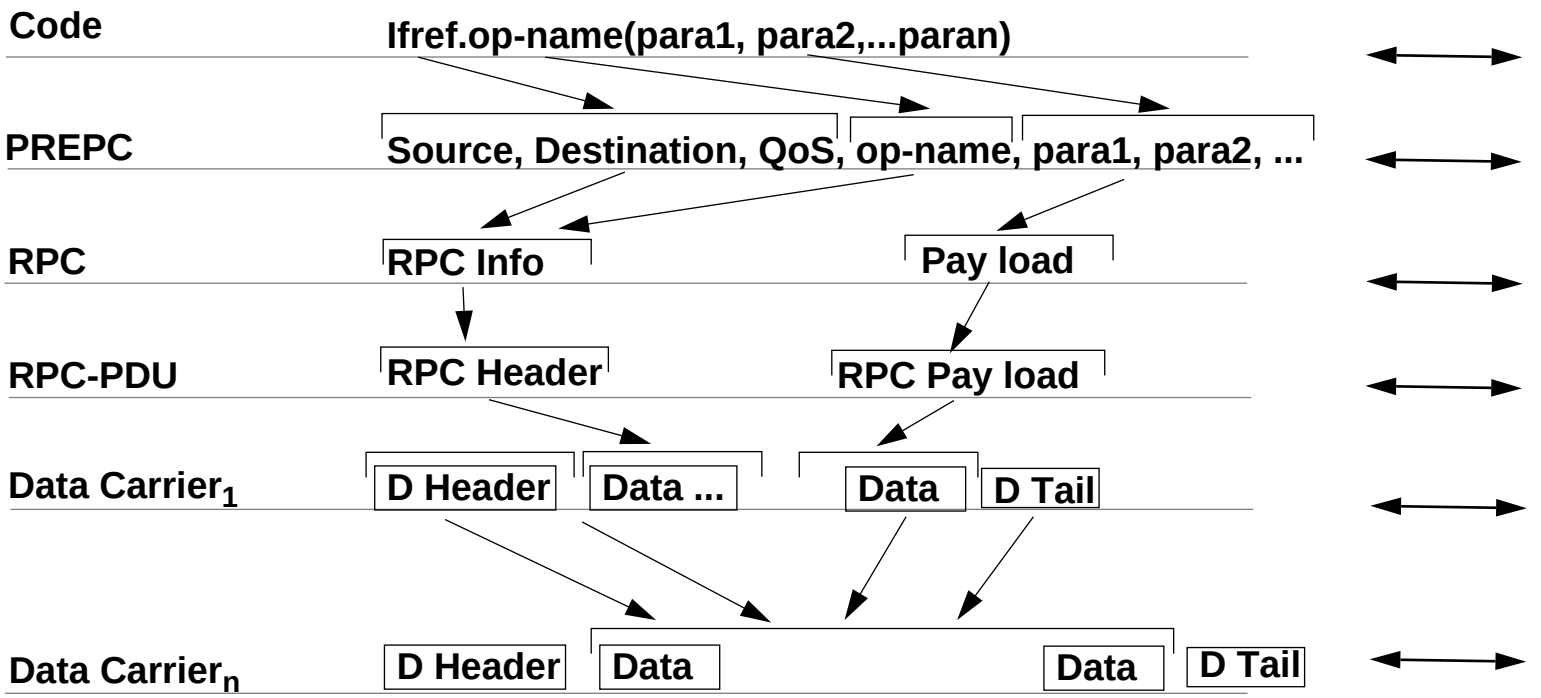
Inter-operability Submissions

Fall into two broad categories:

- **common protocol:**
  - **OSF joint proposal (OSF/DEC/HP) advocating DCE RPC**
  - **Expersoft: advocating TCP/IP based protocol**
- **gateways: acting as proxies for each object in the other world. The submissions advocate application level gateways:**
  - **Sunsoft/Iona**
  - **BNR**
  -

ICL proposal encompasses both approaches.

Comms Protocol Layers



PDU - encryption and fragmentation