Poseidon House
Castle Park
Cambridge CB3 0RD
United Kingdom

TELEPHONE: **Cambridge (01223) 515010**
INTERNATIONAL: **+44 1223 515010**
FAX: **+44 1223 359779**
E-MAIL: **apm@ansa.co.uk**

ANSA Phase III

Using the Interception Model (TC 1/3/95)

Yigal Hoffner

Abstract

This presentation is in three parts.

The first part re-introduces the ANSA Interception Model and shows how it can be used to model, explain and explore some past and present architectural and design decisions.

The second part describes work done to date.

The third part will show the directions in which future work involving trading, interception, boundaries and domain concepts should proceed with a view towards management in large scale distributed systems.

APM.1431.02 **Approved** 1st March 1995

Project Management (confidential to ANSA consortium for 2 years)

Distribution:
Supersedes:
Superseded by:

Using the ANSA Model of Interception

Work in Progress Report

TC Meeting
1 March 1995

Yigal Hoffner (yh@ansa.co.uk)
and
Ben Crawford (bafc@ansa.co.uk)

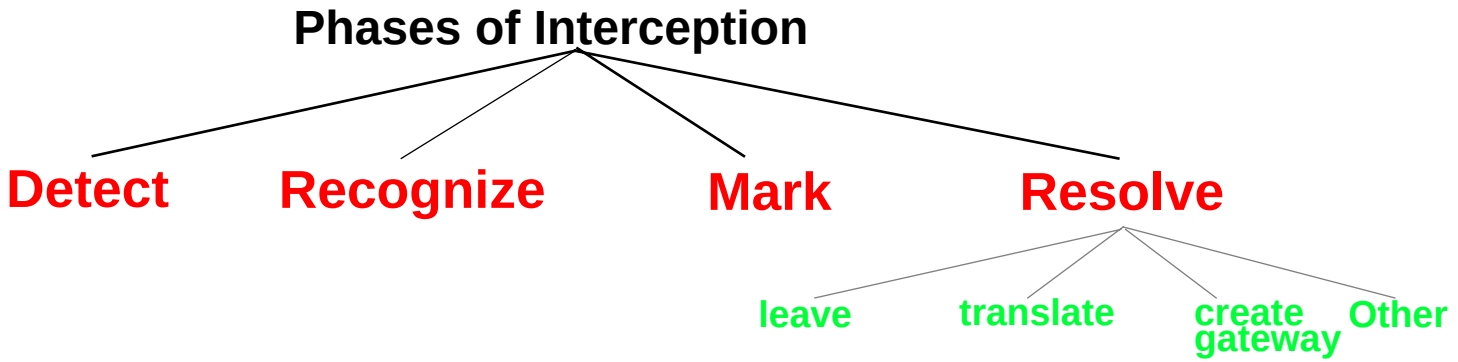
Overview

- **Part I: Using the ANSA Model of Interception**
- **Part II: Current state of progress**
- **Part III: Future Work Plan**

Part I overview: Using the model of Interception

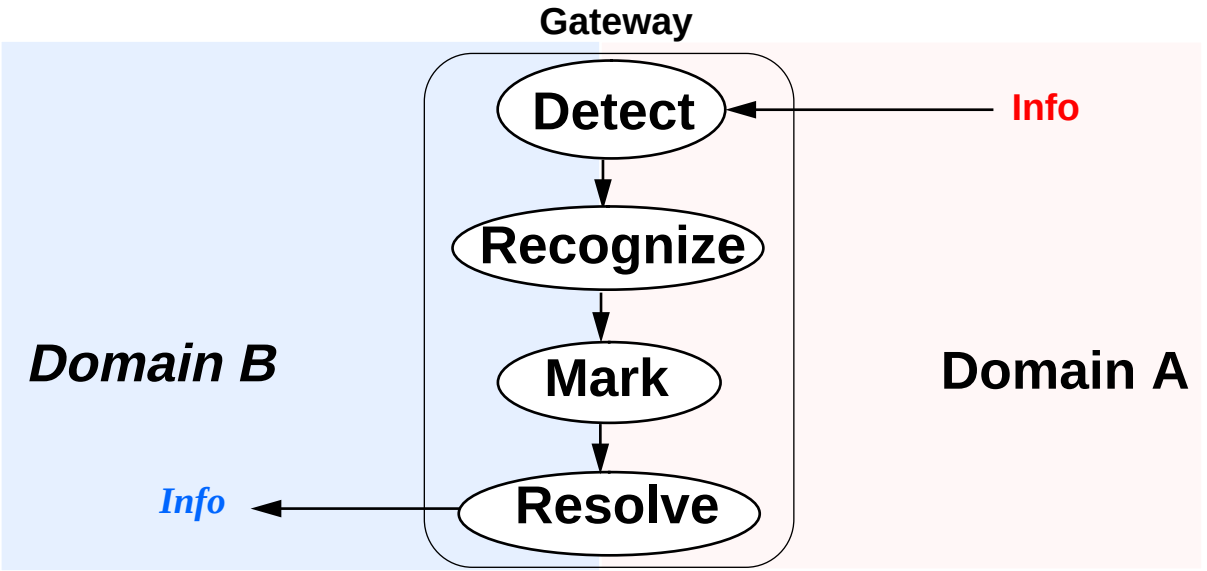
- Phases of the interception process
- Immediate and deferred resolution strategies
- Distributing the interception process in space and time
- Deferred resolution request at different levels:
  - Trader (using the ProxyExport() operation)
  - Binder
- Linking different access domains (bridging physical separation)
- Modelling the binding process

Phases of the Interception process (I)



- DETECT: the crossing of the domain boundary
- RECOGNIZE: the information which should effect or be effected by the crossing
- MARK: the recognized information
- RESOLVE: act according to or on the marked information

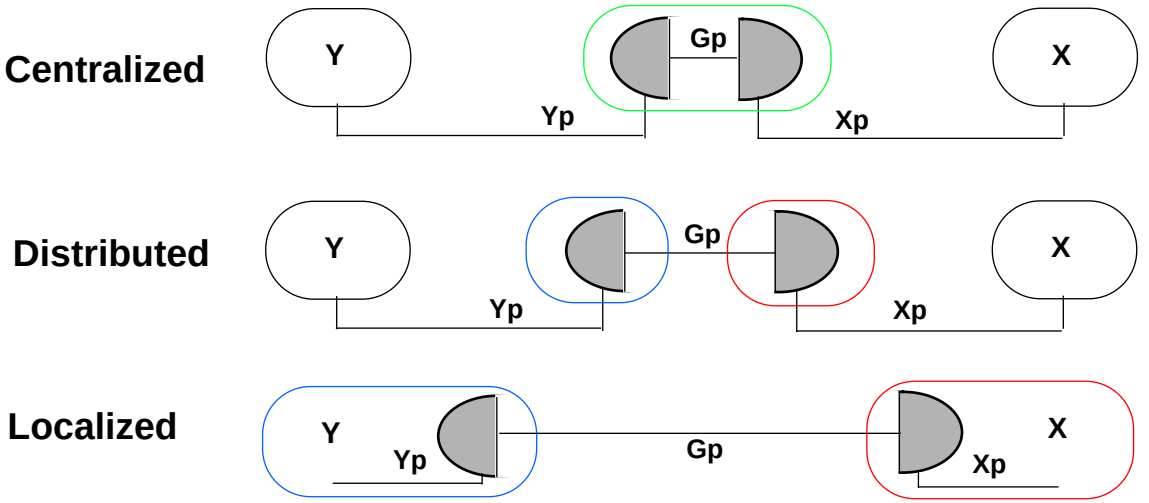
Phases of the Interception process (II)



Distributing the Interception process in time

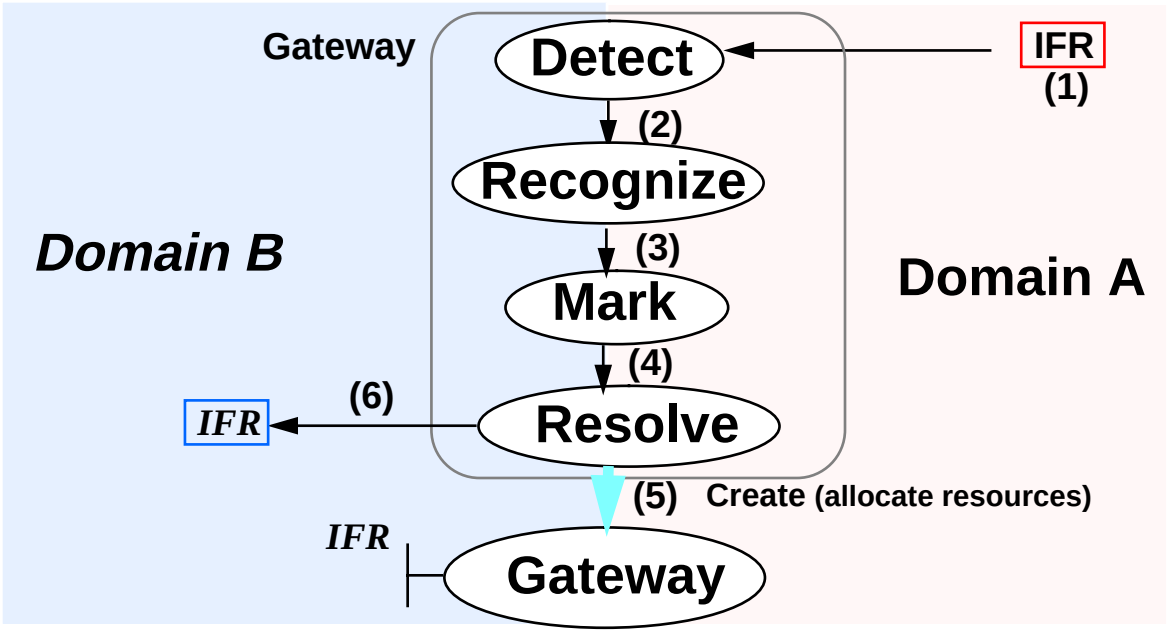
- Distribution in time and space allows deferring options or preventing certain options --> trade-off between issues such as: security, dependability, performance, availability, flexibility
- Time - using different tools/components during the development process:
  - designer
  - preprocessor
  - compiler
  - linker
  - run-time components: client/server, traders, binders, gateways
- Tool chain - an economic way of ensuring gateways are put where required without relying on the application programmer!

Distributing the Interception process in space



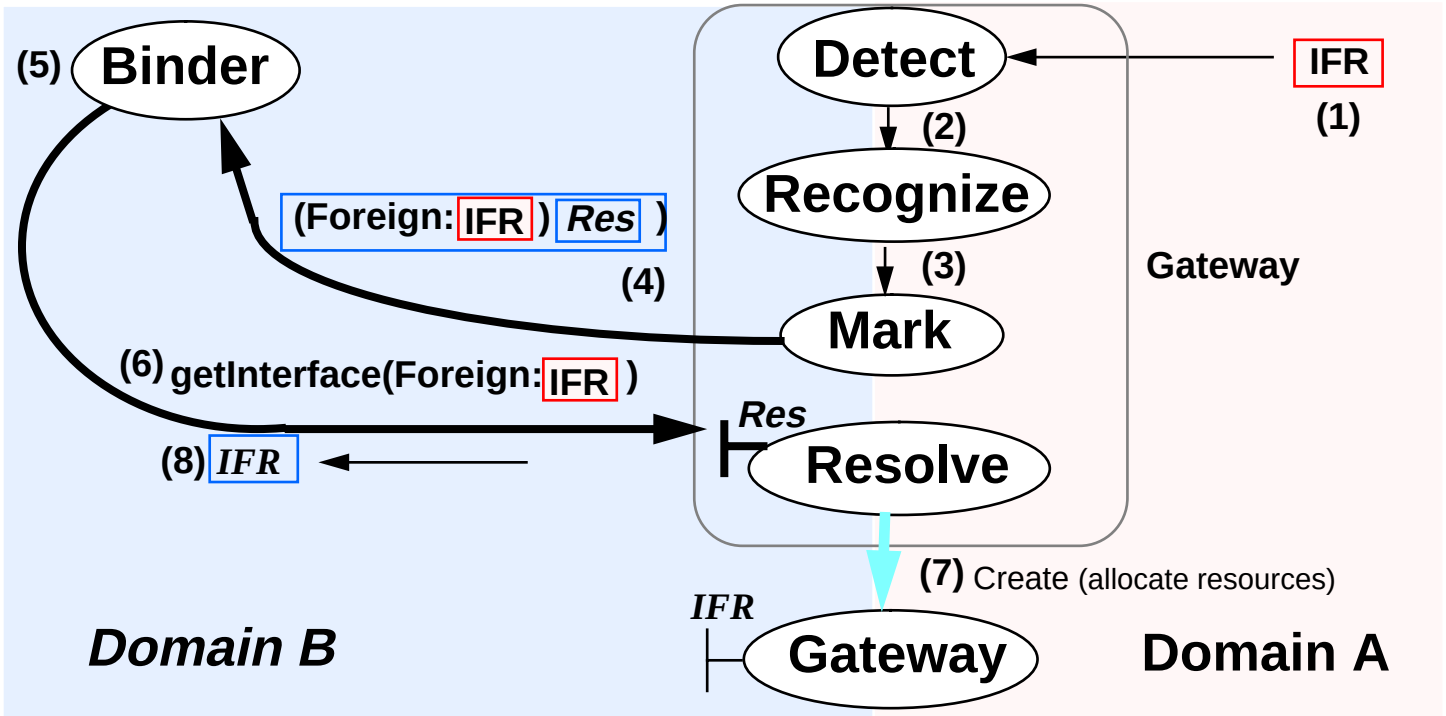
- **Space: where on the invocation path between the objects the interception takes place - inside what encapsulation boundaries:**
  - **address space (stubs)**
  - **node**
  - **domains**

Immediate resolution



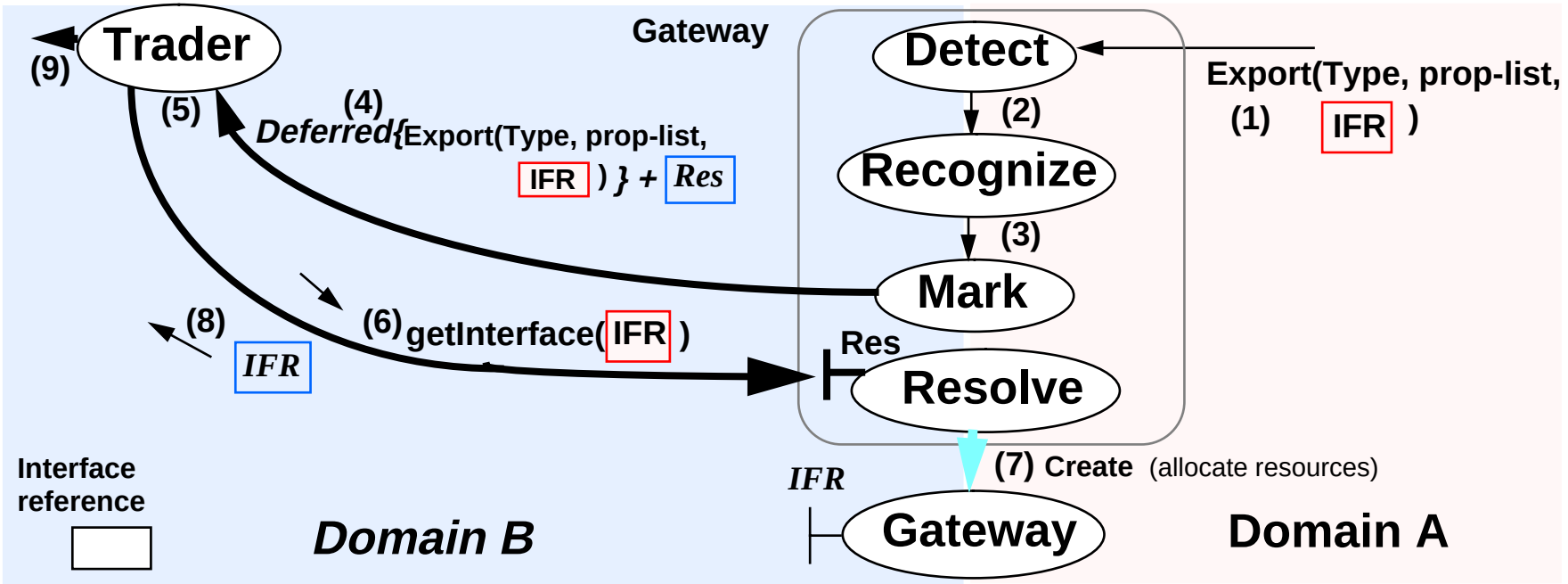
- Recipient gets immediately usable information
- Resources are allocated for the created gateway regardless of whether it will be used or not

Deferred resolution and Binding



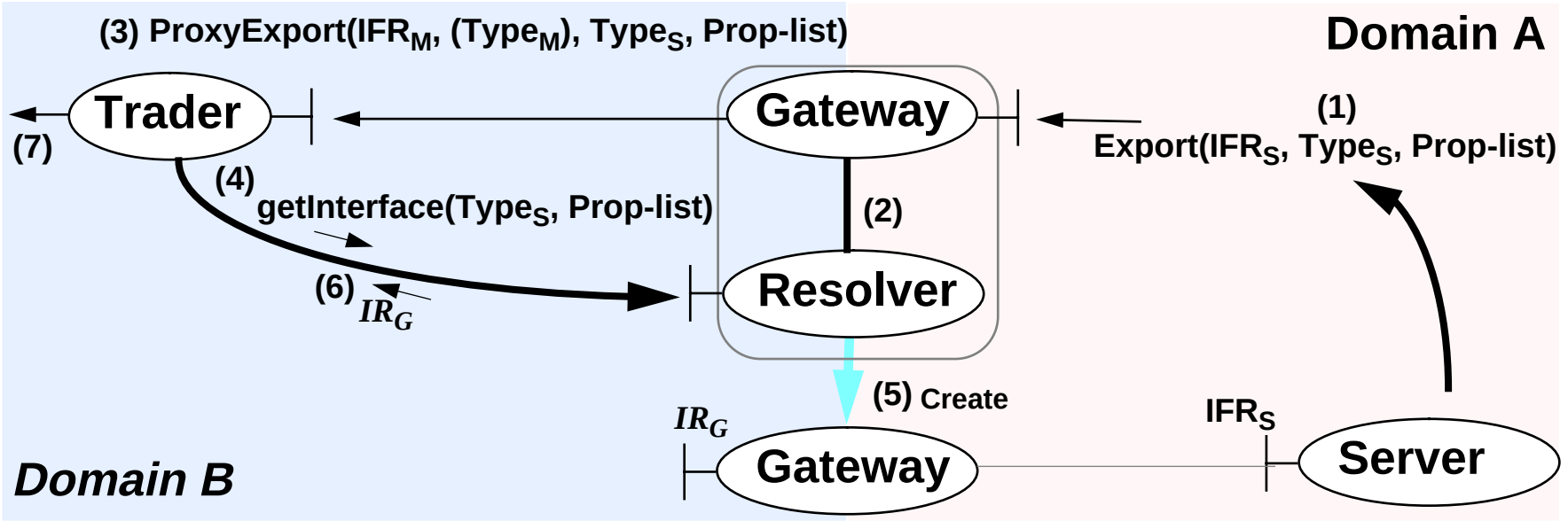
- **Marking the Interface Reference itself**

Deferred resolution and Trading



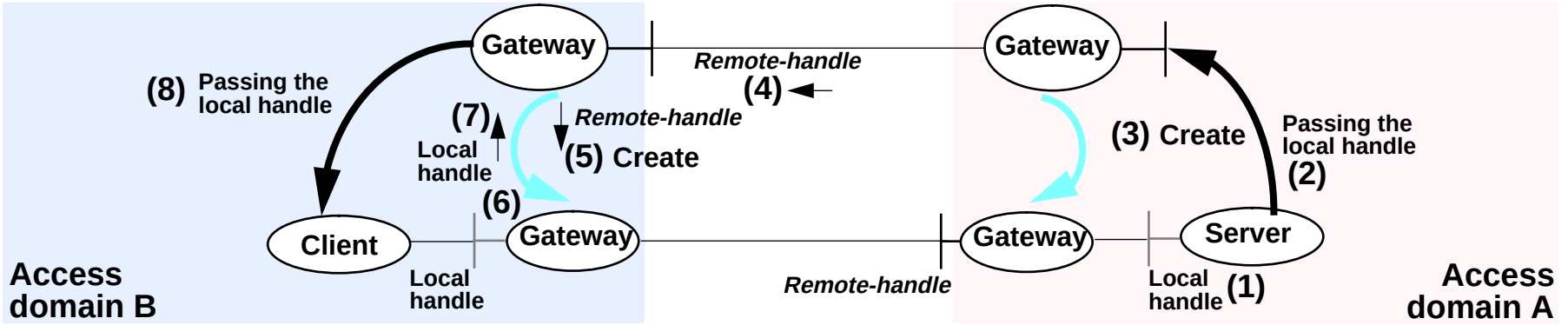
- **Marking the deferment of resolution by marking:** **Export(Type, prop-list, Ifref)**
  - operation name, adding/changing parameters
  - property

Trader ProxyExport() operation



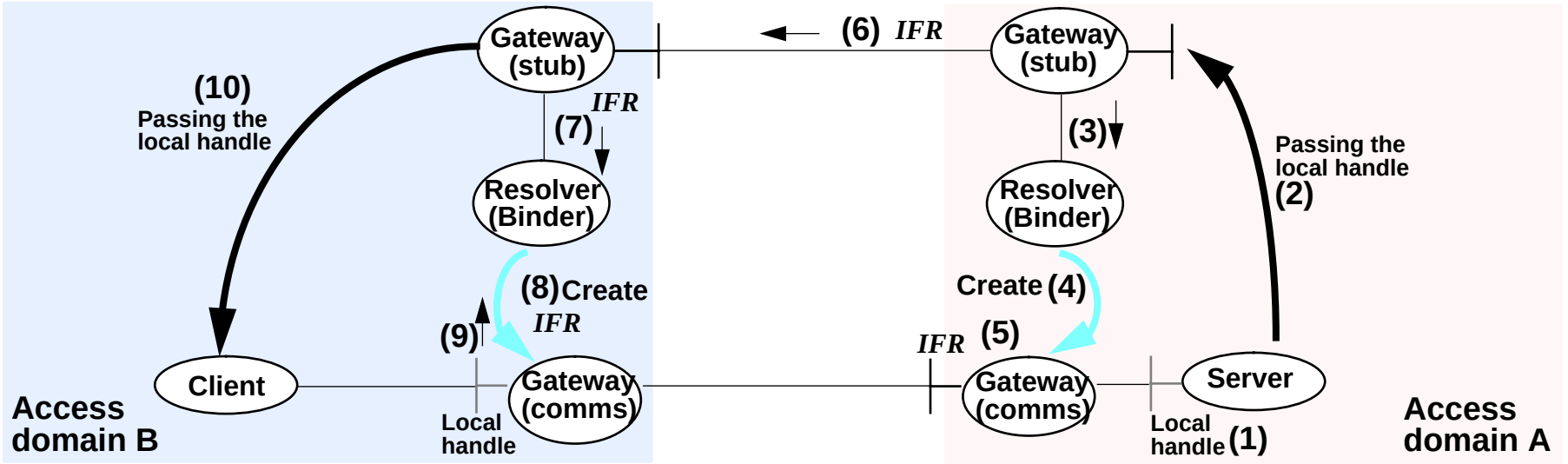
- ANSAware: NodeManager used ProxyExport() to defer creation of services
- In ORBIX Match-Maker: Interface references are properties. ProxyExport is distinguished by whether a property with name ObjRef or MonRef is available

Crossing access domain boundaries



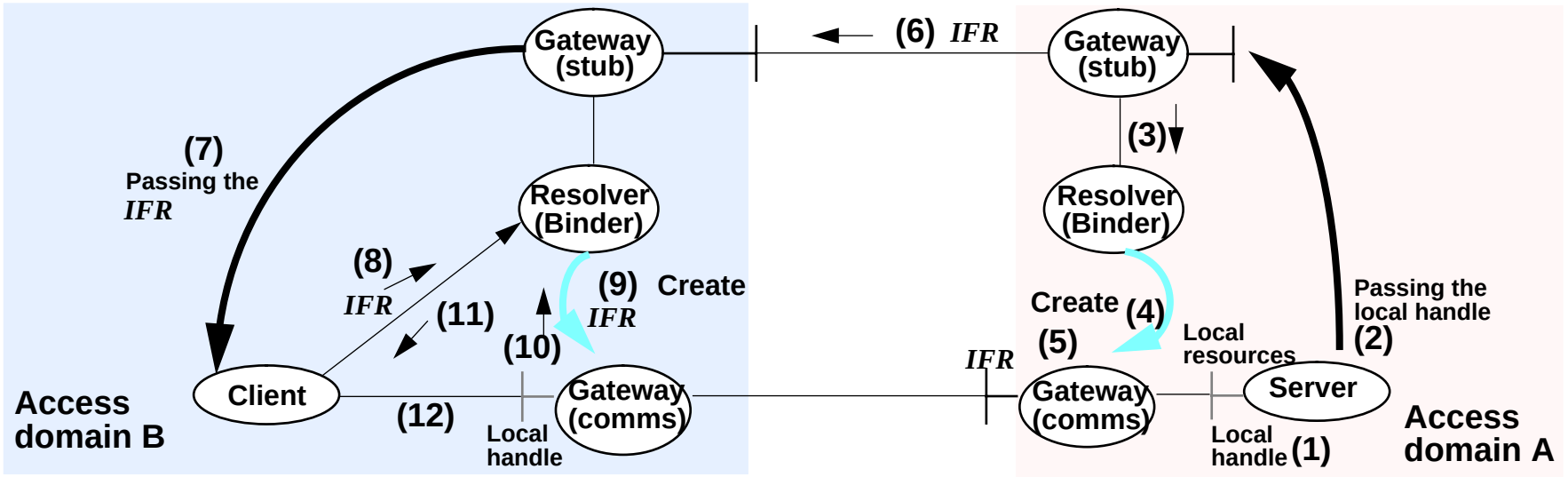
- **Immediate resolution method**
- **Access domains scope:**
  - address space (subroutine)
  - Capsule
  - node
  - LAN
  - WAN

Binding (immediate resolution)



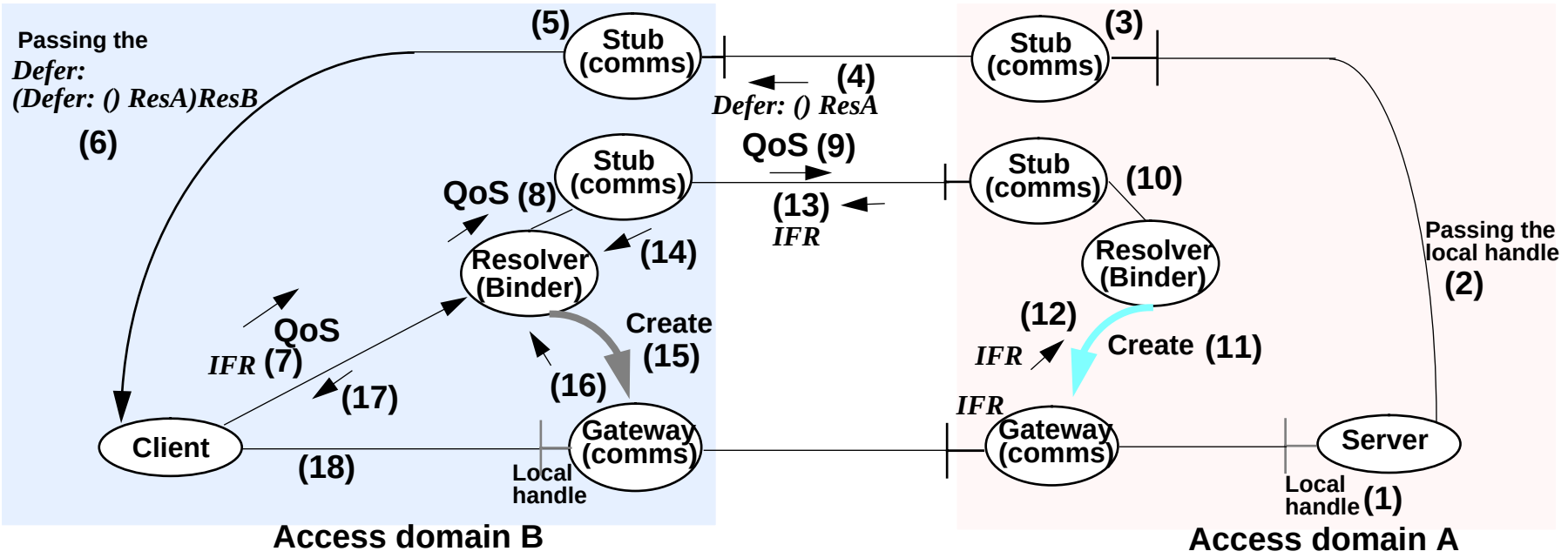
- Gateway of Access domain A resolves the local-handle to an interface reference and allocates the resources for the gateway immediately
- Gateway of Access domain B resolves the interface reference to a local-handle and allocates the resources for the gateway immediately

ANSAware Binding (deferred resolution)



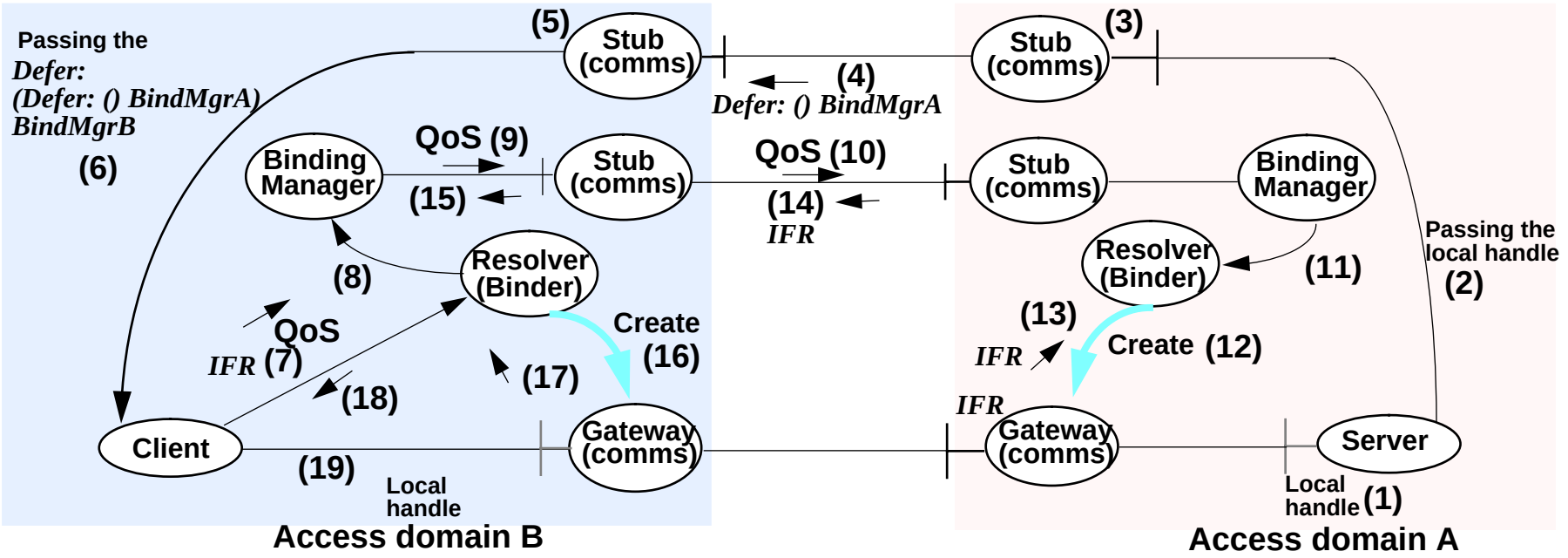
- Default Resolver is Binder
- Interface Reference is put in client stub
- Resolution takes place when client wishes to use service
- Infrastructure does the binding (Implicit binding)

A model of explicit Binding



- Deferring resolution at both boundaries
- Resolution initiated by client

Explicit Binding with Binding Managers

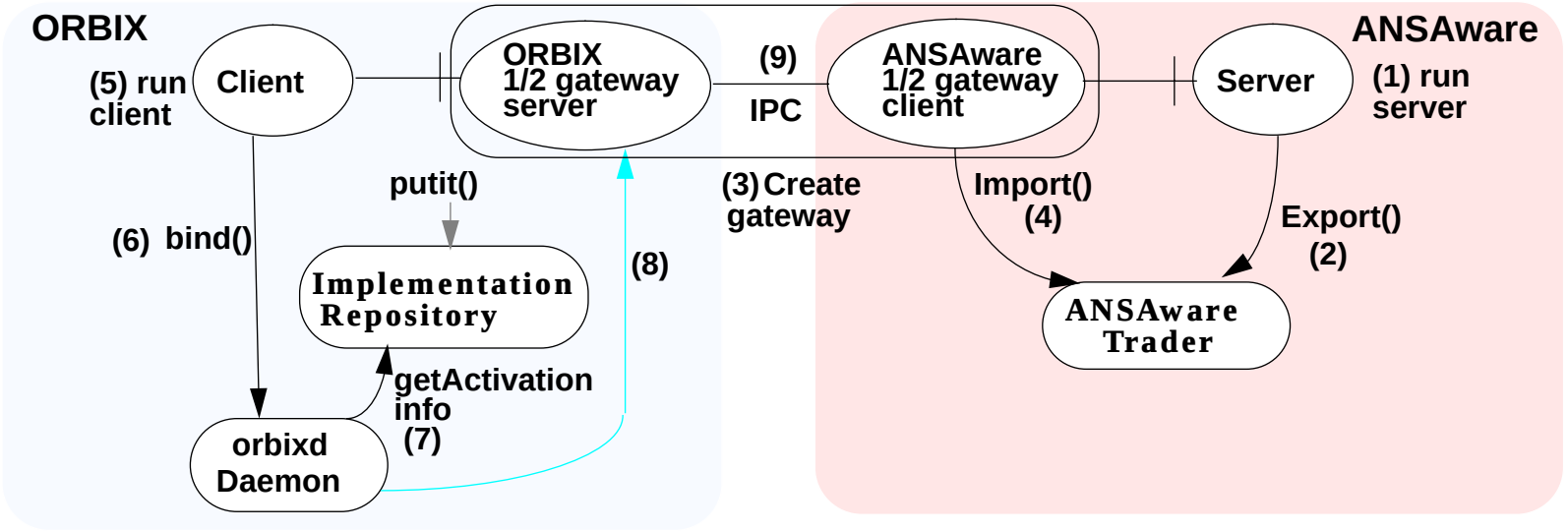


- **Using a Binding Manager to request the resolution from the other Binding Manager**

Current status (Implementation)

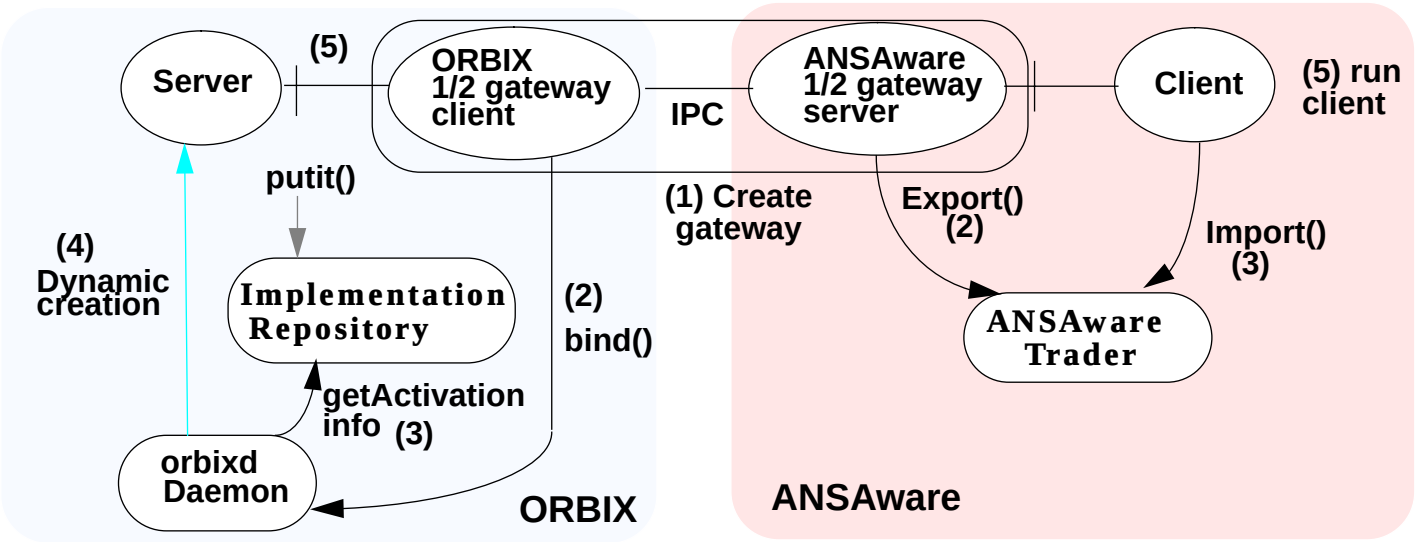
- **Hand crafted implementation of a simple gateway for:**
  - **ANSAware client and Orbix server**
  - **Orbix client and ANSAware server**
- **Monitoring facilities integrated to allow visualization (with the DEMON tool)**
- **Hand crafted implementation of a gateway for:**
  - **ANSAware Trader and Orbix Match-Maker - almost ready**
- **Implement Immediate resolution method between Orbix and ANSAware:**
  - **immediate generation of gateways when Interface References cross**

The ORBIX client - ANSAware server set up



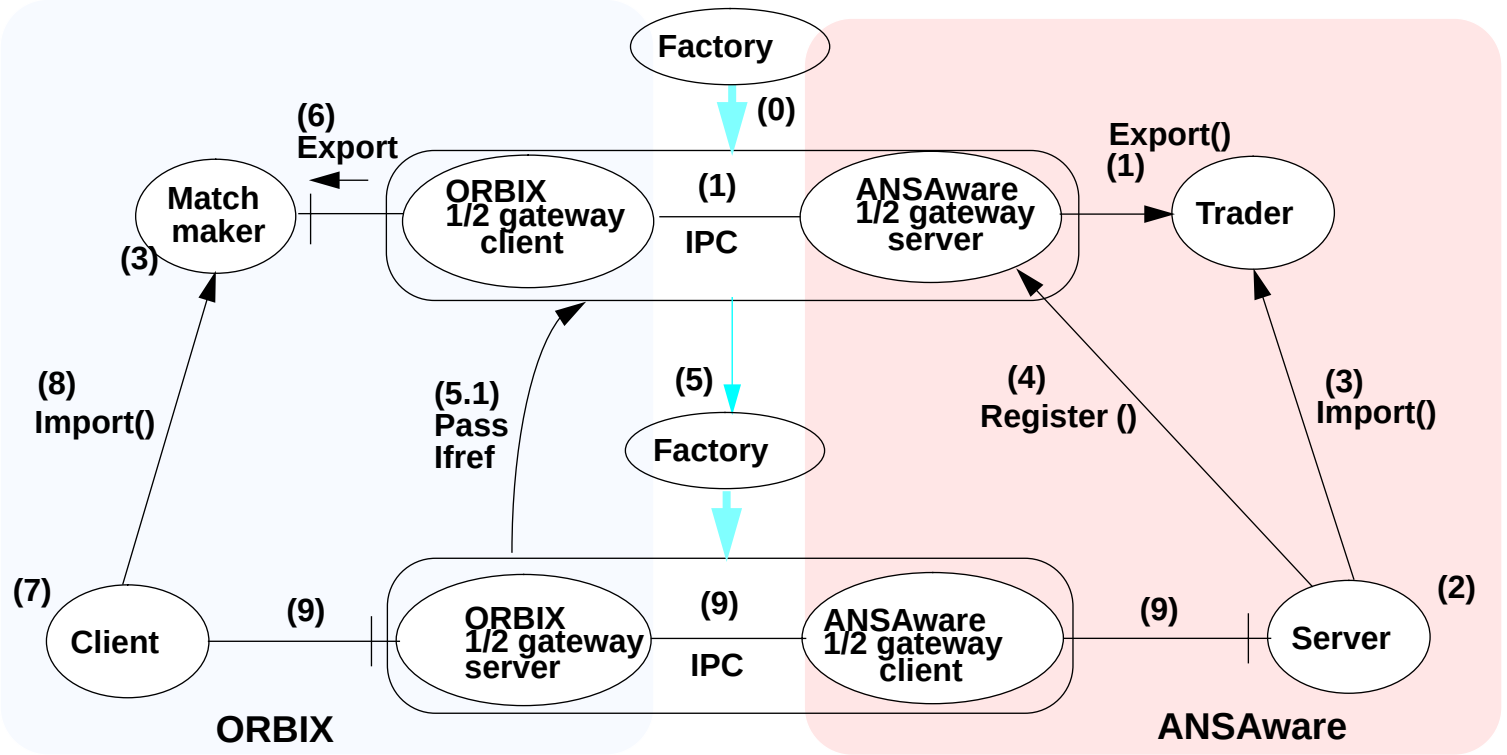
- **Order of instantiation: ANSAware server; Gateway;ORBIX client;**
  - **bind() in ORBIX is a mixture of basic trading and factory process**

The ORBIX server - ANSAware client set up



- **Order of instantiation: Gateway; ORBIX server; ANSAware Client**

Trader Match-Maker gateway (I)



Trader Match-Maker gateway (II)

- **Sequence of actions:**

(0) Activate gateway Factory to create Trader Match-Maker (TMM) gateway

(1) TMM gateway exports its services to AW Trader

(2) Create AW Echo server manually

(3) Echo server Imports TMM gateway service

(4) Echo server Registers its services to TMM Gateway

(5) TMM gateway invokes gateway Factory to create Echo gateway

(5.1) Echo gateway passes its ORBIX ObjectPtr to TMM gateway

(6) TMM gateway invokes Export on Match-Maker passing it Echo gateway ObjectPtr

(7) ORBIX client is created

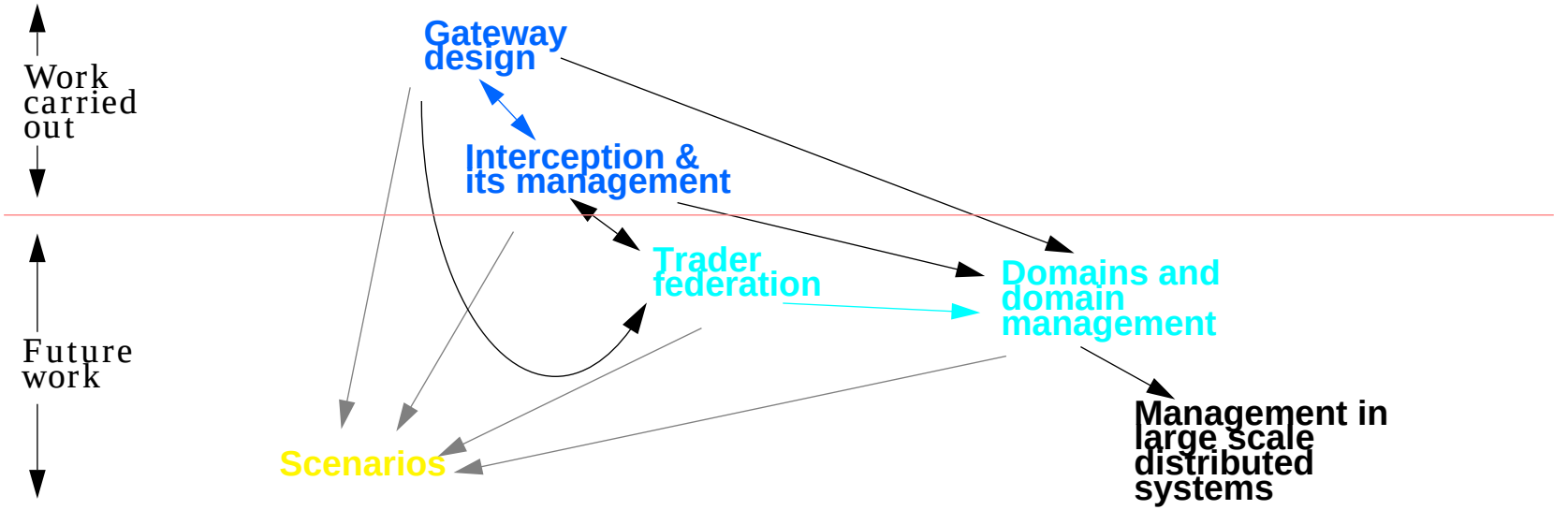
(8) ORBIX Echo client imports Echo service and gets Echo gateway ObjectPtr

(9) Client invokes server through Echo gateway

Current status (Documentation)

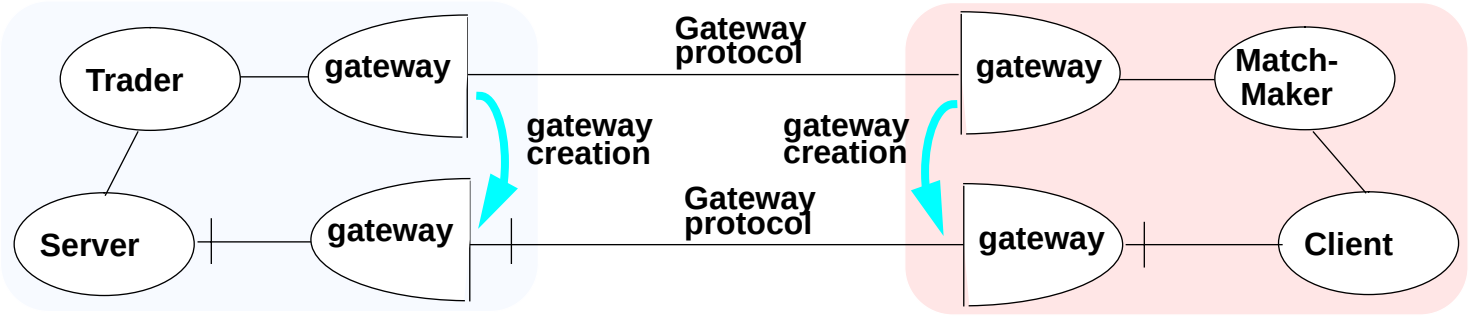
- **Dcuments:**
  - **APM.1313: “Introduction to application level gateways”**
  - **APM.1427: “Passing interface references across domain boundaries”**
- **Continued effort: Dynamic management of gateways:**
  - **gateway maintenance**
  - **gateway destruction**

Future Work: towards a Management Engine (B2 workpackage)



- **Trader federation**
- **Domains and domain management**

Trader Federation

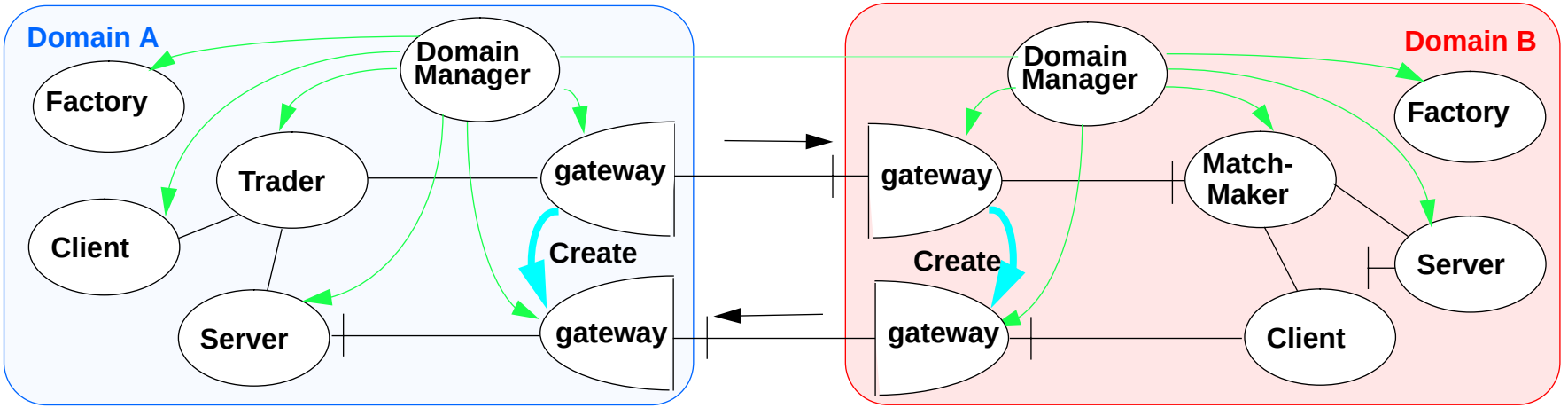


- **Unresolved issues:**
  - linking traders and minimal trading configuration
  - trading domains
  - visibility (advertizing/search policy) issues
  - trading and QoS issues
  - domain hopping (circularity and inefficient paths)
  - extension to trading model
  - trading and CORBA repositories

Limitations of “Domain” work in literature

- **SLOMAN’s work:**
  - **Relationship between domains - cannot be dealt with in isolation**
  - **An important aspect of domains is ability of domain managers to interact with each otehr**
  - **Dealing with dynamic links created across domain boundaries**
  - **How are the boundaries created and maintained: technology for propagating domain boundary scope was missing**

Domains and domain management



- **Unresolved issues:**
  - heavy-weight and light-weight models of domains
  - domain management: how much control is put where
  - cloning issues
  - information needed to construct domains and gateways
  - implementation issues

Future Federation work

- **Filling the gaps on Application level gateways and Interception management**
- **Trader federation**
- **Domains and their management:**
  - **Technical domains:**
  - **Security: tying the model of interception, trading and domains with security issues**
  - **QoS domains**
  - **Monitoring domains**