

CORBA

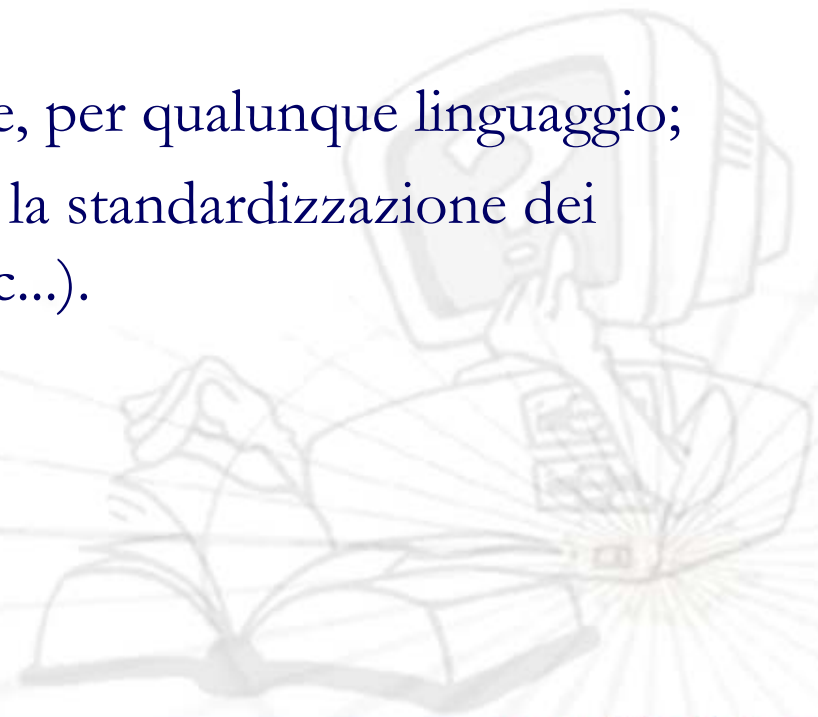
Introduzione all'Architettura

di Tito Flagella

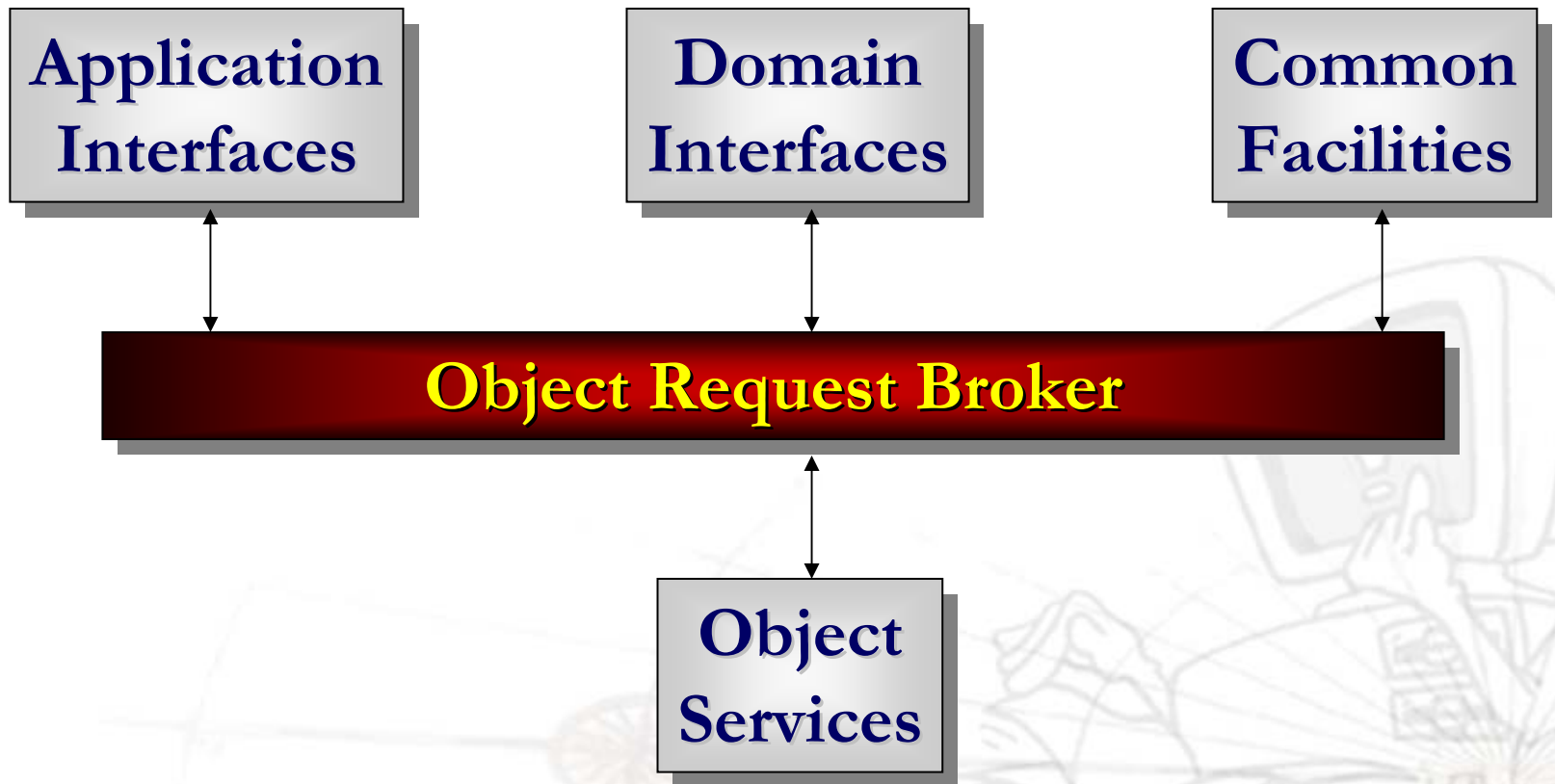


CORBA: uno standard per le Applicazioni Distribuite

- Curato da OMG, più di ottocento membri;
- Prima versione adottata in ottobre 91;
- Prime implementazioni dal 93;
- Molte le implementazioni, anche free, per qualunque linguaggio;
- Interoperabilità assicurata attraverso la standardizzazione dei protocolli *Inter Orb* (GIOP, IIOP, etc...).



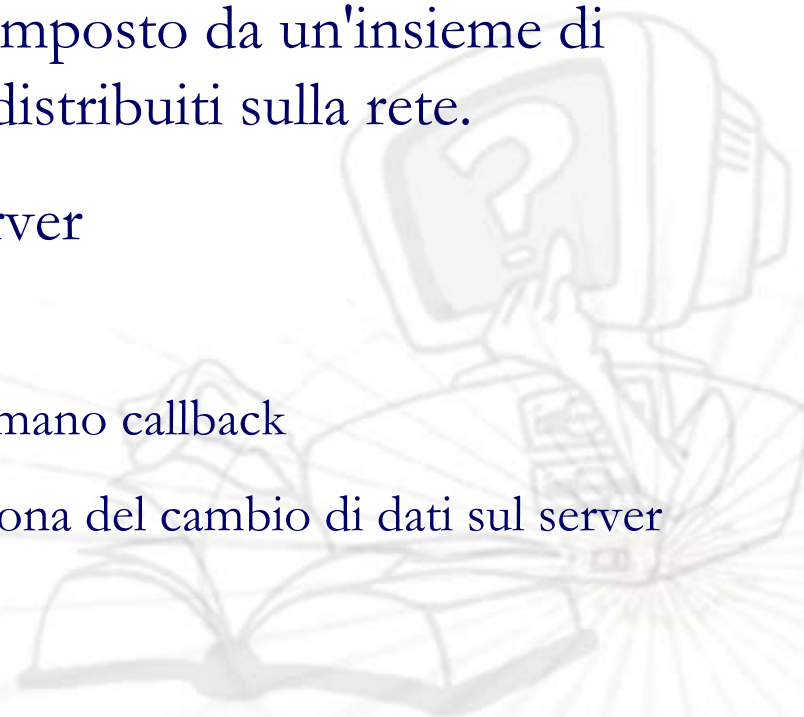
OMG Object Management Architecture - OMA



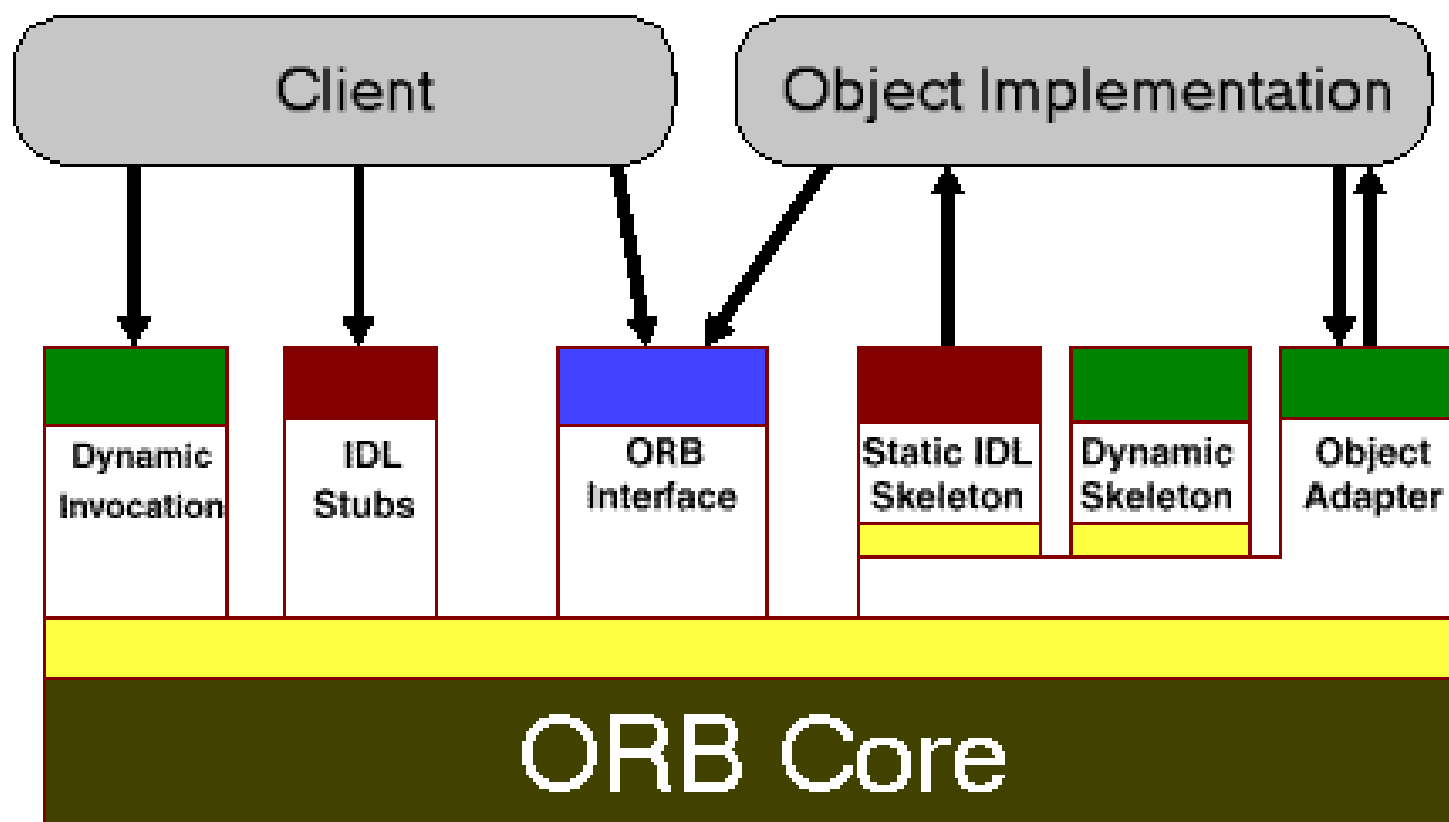
- CORBA è un'ambiente utile per:
 - integrare applicazioni parzialmente già esistenti
 - costruire nuove applicazioni distribuite

Un tipico sistema basato su CORBA è composto da un'insieme di programmi client che accedono a servizi distribuiti sulla rete.

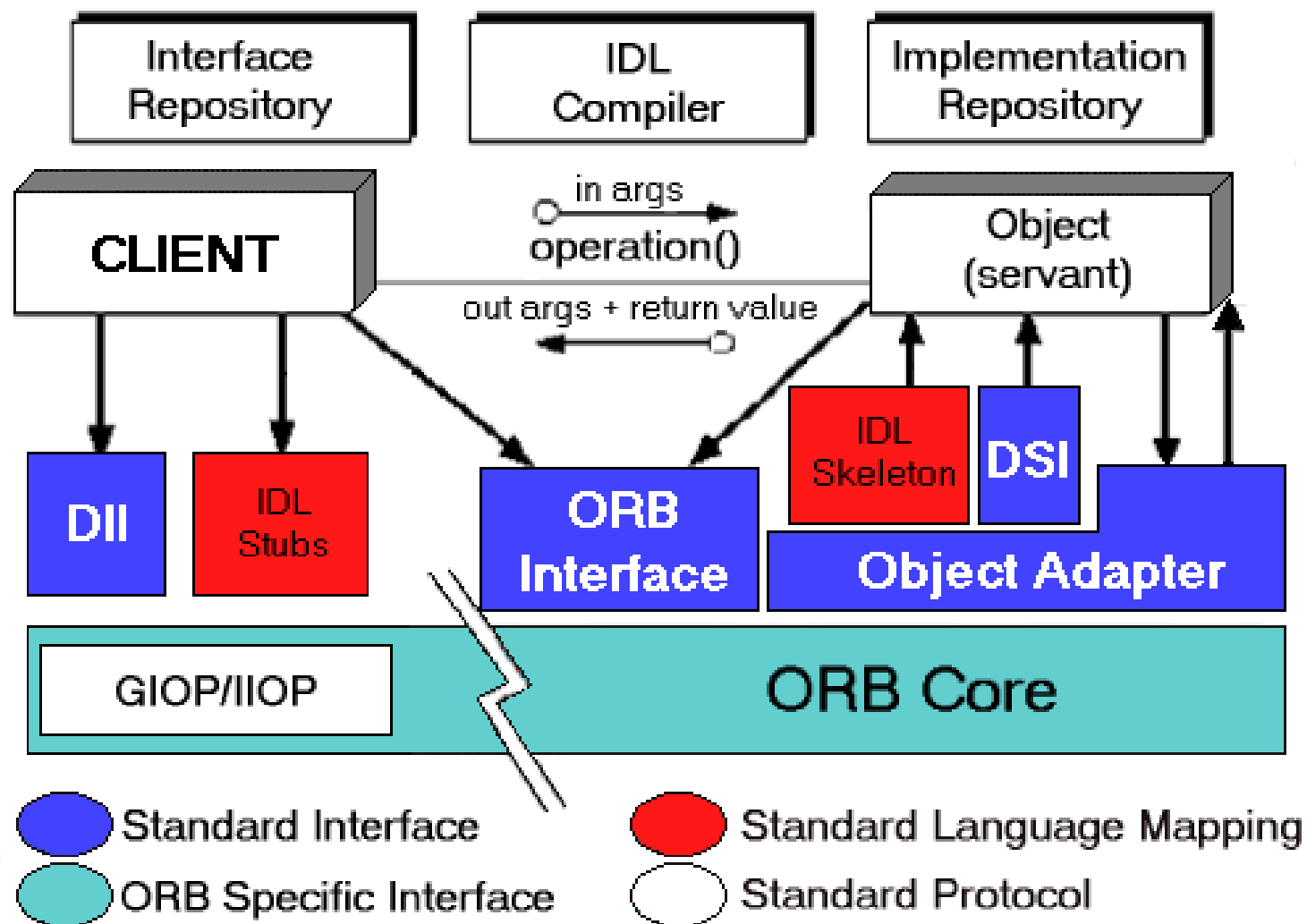
- Architettura peer-to-peer, non client-server
 - un client può definire servizi
 - le chiamate al servizio del client si chiamano callback
 - sono spesso usate per la notifica asincrona del cambio di dati sul server



- Interfacce e Dati sono descritti in un apposito linguaggio indipendente dall'implementazione, detto *Interface Definition Language* (IDL)
- Lo standard CORBA include la definizione del binding da IDL ai linguaggi di implementazione, come C, C++, Java, COBOL,
- Le chiamate per default sono bloccanti (stessa semantica dei linguaggi di programmazione)
- Possono però anche essere di tipo asincrono:
 - dichiarazioni oneway in IDL
 - uso di servizi specifici: event-service, notification service, message service



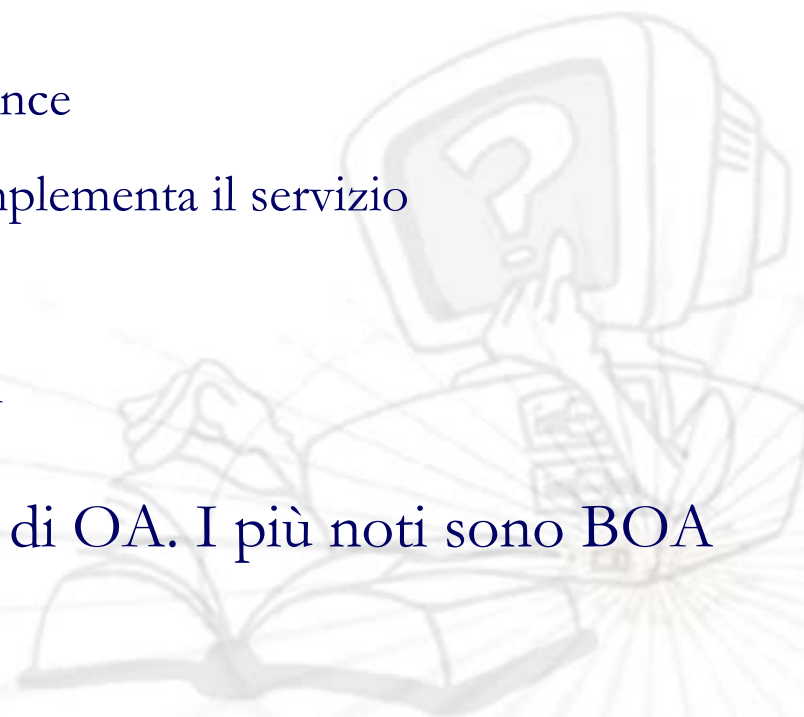
- Interface identical for all ORB implementations
 - There may be multiple object adapters
 - There are stubs and a skeleton for each object type
 - ORB-dependent interface
- ↑ Up-call interface
↓ Normal call interface



Components in the CORBA Reference Model

- Un Client effettua una richiesta (statica o dinamica), utilizzando un "Object Reference" per individuare il servizio;
- Il Client deve conoscere il tipo dei parametri;
- Nel caso statico usa lo Stub, nel dinamico l'Interface Repository (un servizio che consente la consultazione a run-time delle interfacce).

- Sul lato server, il codice dell'object implementation viene invocato dall'OA tramite una "up-call":
 - associata a uno skeleton statico generato dal compilatore idl (SSI)
 - costruita dinamicamente, utilizzando l'Interface Repository (DSI)
- L'Object Adapter tratta aspetti quali:
 - generazione e interpretazione dell'Object Reference
 - corrispondenza tra oggetto e processo che ne implementa il servizio
 - invocazione dei metodi
 - attivazione, disattivazione delle implementazioni
- Ci possono essere un numero arbitrario di OA. I più noti sono BOA (obsoleto) e POA



- La semantica dell'ORB per l'operazione statica o dinamica è esattamente la stessa
- Il server non ha modo di verificare se la richiesta sia stata statica o dinamica
- Il client non ha modo di verificare se il server stia usando SSI o DSI

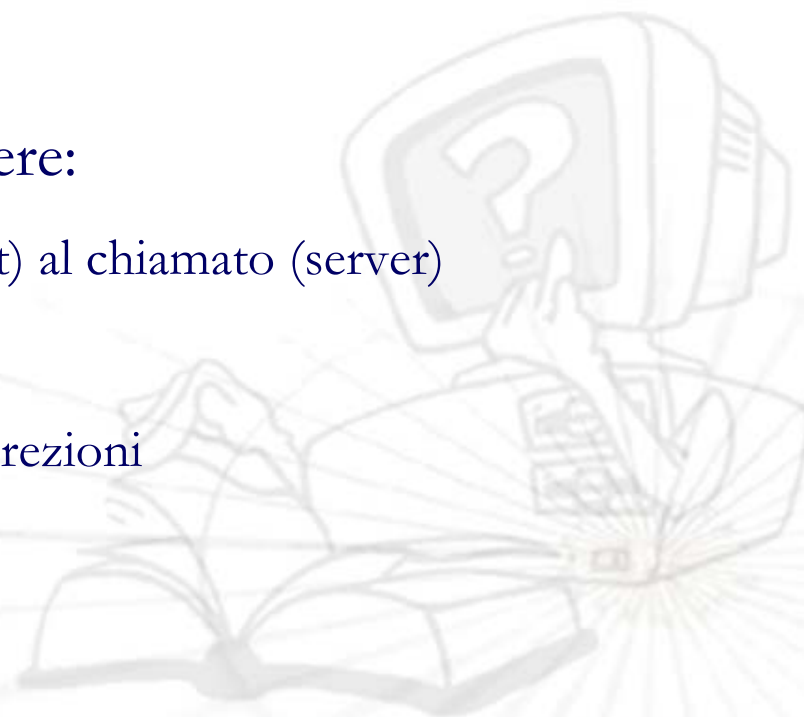
- Il sistema più elementare per stabilire una connessione diretta tra client e server è l'uso dello IOR;
- Uno IOR, Interoperable Object Reference, costituisce un indirizzo univoco corrispondente ad un'oggetto CORBA.
- Il server genera uno IOR per l'oggetto da rendere accessibile e lo comunica ai potenziali clienti tramite uno dei seguenti mezzi:
 - file system condiviso
 - pubblicazione su un server Web trasmissione off-line
 - uso del Naming Service

- E' l'astrazione usata in CORBA per separare le interfacce degli oggetti dalle loro implementazioni;
- Parte dello Standard CORBA di OMG;
- Ne esistono numerose altre versioni;
- Standardizzato come standard ISO;
- E' utilizzato per descrivere le interfacce degli oggetti e i tipi dei parametri;
- IDL non e' un linguaggio di programmazione (non serve per implementare gli oggetti o per realizzare client che accedano agli oggetti);
- IDL e' volutamente semplice per essere facilmente mappabile a linguaggi di programmazione poco evoluti.

```
interface FrontOffice {  
    readonly attribute string name;  
    readonly attribute unsigned long numberOfSeats;  
    Price getPrice(in Place chosenPlace);  
    boolean bookSingleSeat(in Place chosenPlace, in string creditCard);  
};
```

- L'interfaccia IDL di un oggetto contiene tutte le informazioni utili per realizzare un client che lo usi;
- Una tipica interfaccia contiene la specifica di:
 - le operazioni supportate dall'interfaccia (compresa la specifica dei tipi dei parametri e del valore di ritorno)
 - gli attributi dell'interfaccia

- Ogni procedura specifica il nome, il tipo e la modalita` di trasmissione dei suoi parametri
 - Es: boolean checkIfOpen(in Date when, out Date nextAvailableDate);
- La modalità di trasmissione può essere:
 - in: il parametro viaggia dal chiamante (client) al chiamato (server)
 - out: il parametro viaggia dal server al client
 - inout: il parametro viaggia in entrambe le direzioni



- Tutte le definizioni IDL sono public
- Non esiste il concetto di private o protected, che riguardano l'implementazione più delle interfacce
- Mancanza di costruttori o distruttori: sostituiti dalle factory, oggetti capaci di creare altri tipi di oggetti
- L'overloading dei nomi delle operazioni è illegale
 - funzionalità utile ma difficile da tradurre in linguaggi che non la supportano

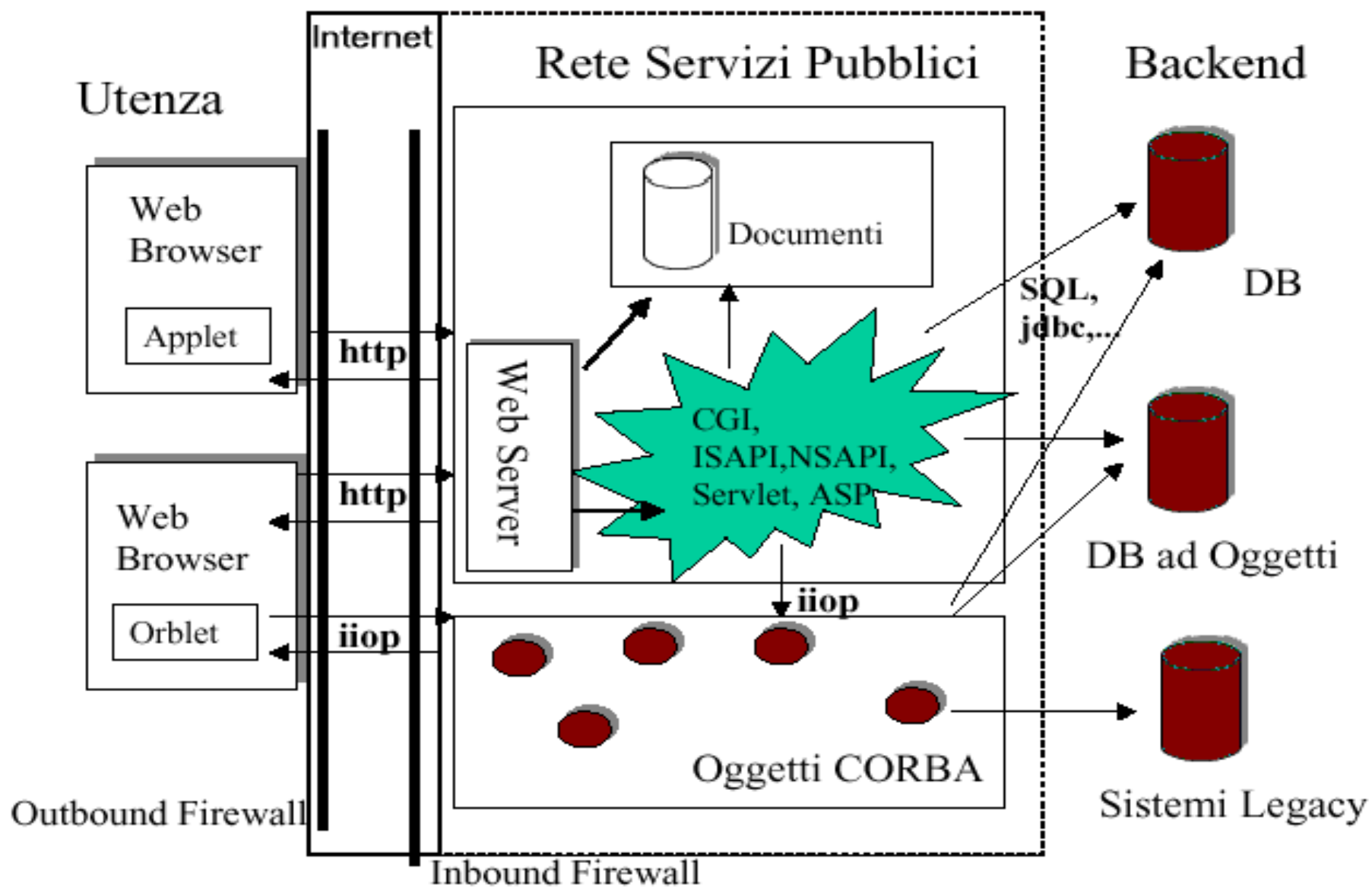
Il compilatore IDL traduce queste specifiche in API e definizione di tipi, dipendenti dal linguaggio scelto

- Specificano come usare CORBA da ognuno dei linguaggi supportati
 - Esistono specifiche ufficiali per:
 - C, C++, Smalltalk, COBOL, ADA e Java
 - CORBA è comunque già utilizzabile anche da:
 - Visual Basic, Modula3, Perl, TCL, Python, Dylan, Oberon e Objective-C
- Enormi differenze nelle difficoltà di programmazione a seconda dei linguaggi usati

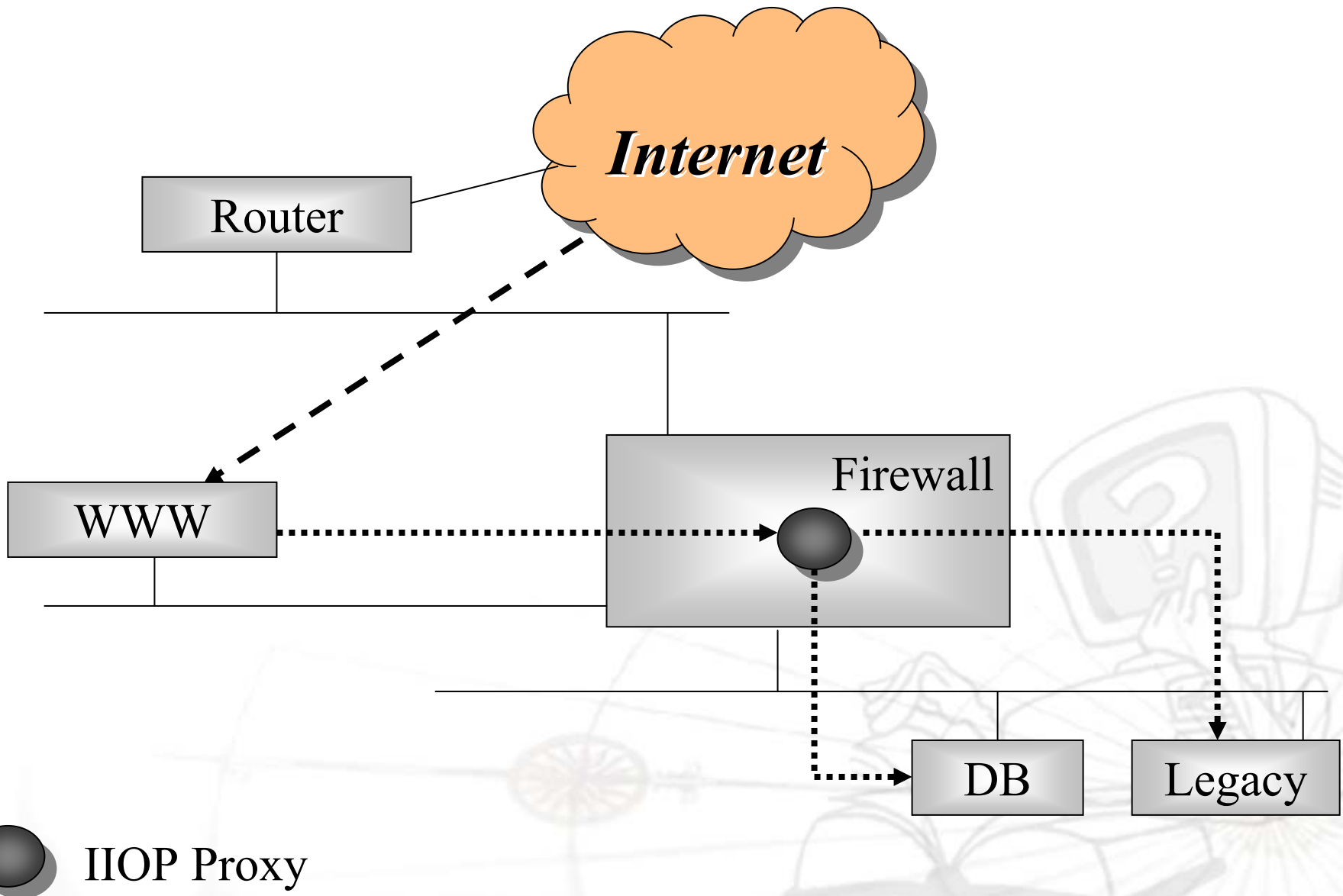
- In C/C++ ci si deve preoccupare di:
 - Allocazione/deallocazione degli oggetti (è suggerito l'uso di smart pointer generati dal compilatore IDL);
 - Uso di tipi CORBA per i tipi primitivi (int, long, ...);

- In Java la programmazione è molto più elegante;
- L'uso di CORBA diventa quasi completamente trasparente;
- La mancanza dell'ereditarietà multipla complica l'implementazione dell'ereditarietà in IDL;

CORBA per le applicazioni Internet



IIOP come protocollo per transitare dal Firewall



IIOP come protocollo per transitare dal Firewall

