
CCM : Support des composants dans l'environnement CORBA

Bruno TRAVERSON (EDF R&D)

22 mai 2003

Sommaire

- Approche composants
- Rappels sur CORBA Objet
- Spécification CORBA Composant
- Perspectives

Sommaire

- Approche composants
- Rappels sur CORBA Objet
- Spécification CORBA Composant
- Perspectives

Composants

- Objectifs
 - Intégration de composants sur étagère,
 - Réutilisation d'applications patrimoniales.
- Concepts
 - Composant,
 - Port.
- Variantes
 - composite,
 - composant co-localisé,
 - configuration.
- Limites
 - héritage,
 - généricité.

Connecteurs

- Objectif
 - Réutilisation des règles d'interaction,
 - Séparation des aspects de communication/coordination des services fonctionnels,
 - Prise en compte des aspects de conversion et d'intermédiation.
- Concepts
 - Connecteur,
 - Prise : rôle d'un composant dans une interaction.

Assemblages

- Association de composants et de connecteurs avec des contrats explicites,
- Utilisation dans des contextes fonctionnels et techniques variables,
- Négociation, choix de l'assemblage le mieux adapté.

L'importance des standards

- La viabilité d'un composant suppose qu'il s'appuie sur des services « standards » et qu'il propose de services « standards »:
 - services requis doivent être largement supportés,
 - services offerts doivent être largement demandés.
- Besoin de standards aux différentes phases d'utilisation des composants :
 - spécification/conception/description,
 - programmation,
 - conditionnement et déploiement,
 - exécution.

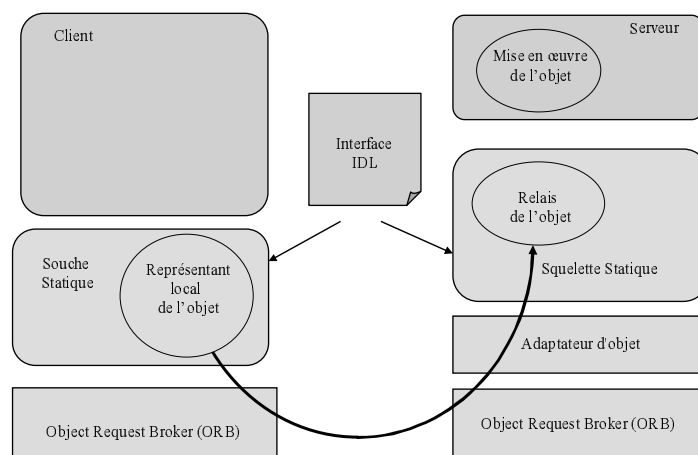
Sommaire

- Approche composants
- Rappels sur CORBA Objet
- Spécification CORBA Composant
- Perspectives

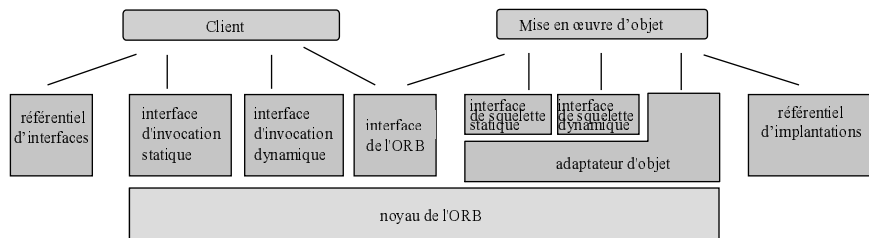
Phases d'utilisation

- Spécification/conception/description
 - OMG IDL (Interface Definition Language).
- Programmation
 - Règles de projection de l'OMG IDL mapping.
- Conditionnement et déploiement
- Exécution
 - Architecture CORBA 2.x,
 - Modèle OMA.

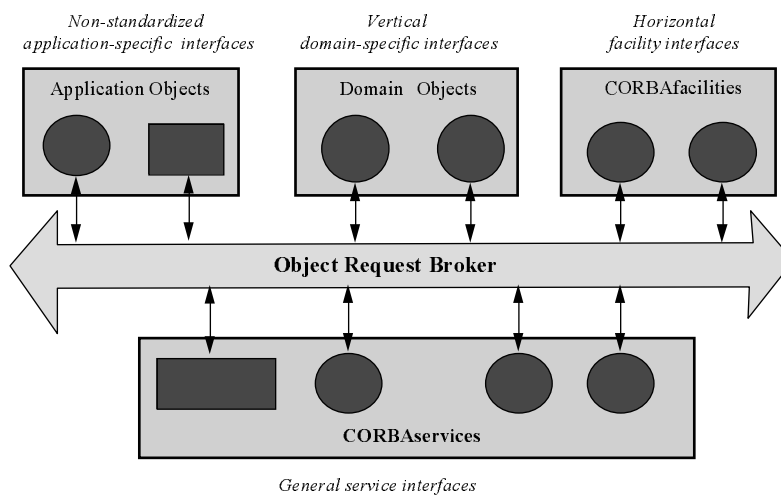
Schéma de développement



L'architecture CORBA 2.x



Le modèle OMA



Offre CORBA

- ORBs d'éditeurs logiciels
 - Object Broker de BEA Systems,
 - Orbix de IONA Technologies,
 - Visibroker de Borland.
- ORBs de constructeurs
 - Component Broker de IBM,
 - ORB Plus de HP.
- ORBs du domaine public
 - JacORB, Mico, OmniORB, OpenORB, TAO ...

Offre Serveurs d'Application

- ASP de IONA Technologies
- BES de Borland
- WEBLogic de BEA Systems
- OAS de Oracle
- WEBSphere de IBM

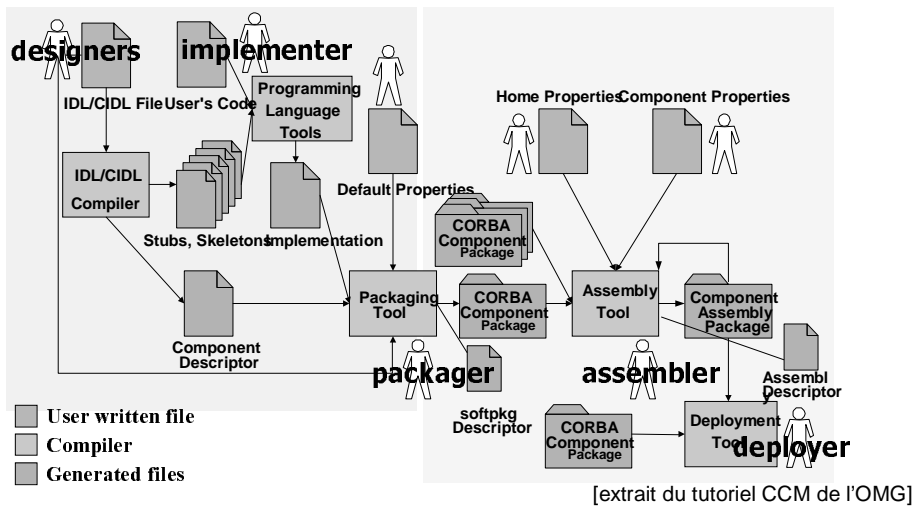
Sommaire

- Approche composants
- Rappels sur CORBA Objet
- Spécification CORBA Composant
- Perspectives

Phases d'utilisation

- Spécification/conception/description
 - Extensions à l'OMG IDL,
 - Modèle de composants.
- Programmation
 - Component Implementation Framework (CIF),
 - Component Implementation Definition Language (CIDL),
 - Règles de projection de l'OMG IDL mapping.
- Conditionnement et déploiement
 - Format d'archive,
 - Descripteurs de déploiement.
- Exécution
 - Architecture de conteneurs.

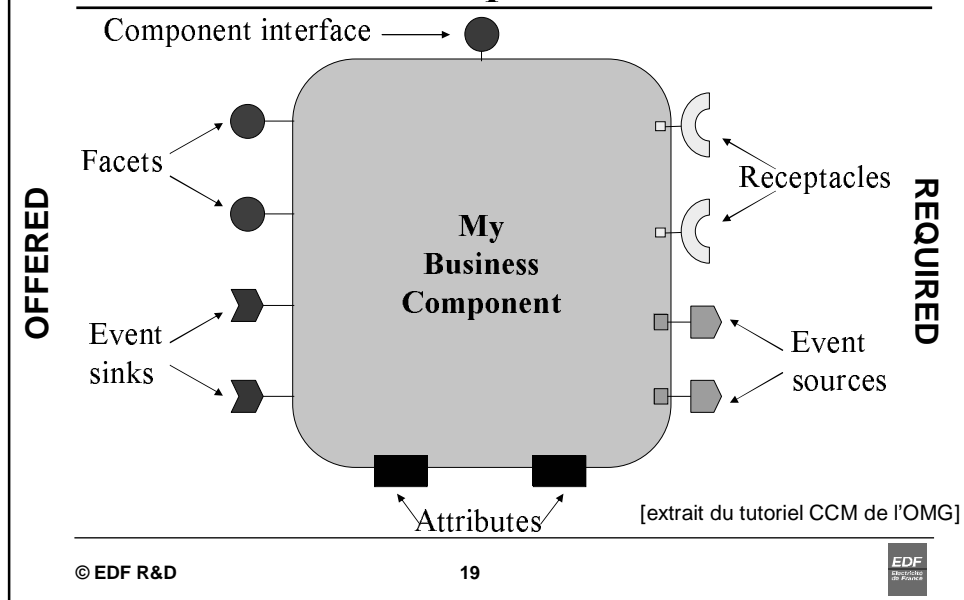
The CCM Big Picture



Spécification/conception/description

- Extensions à l'OMG IDL
 - Nouveaux mots-clés (composant, port, fabrique),
 - Exceptions pour les attributs.
- Modèle de composants
 - Héritage de composants,
 - Pas de support des composites, des connecteurs et des assemblages.

Modèle de composant CORBA



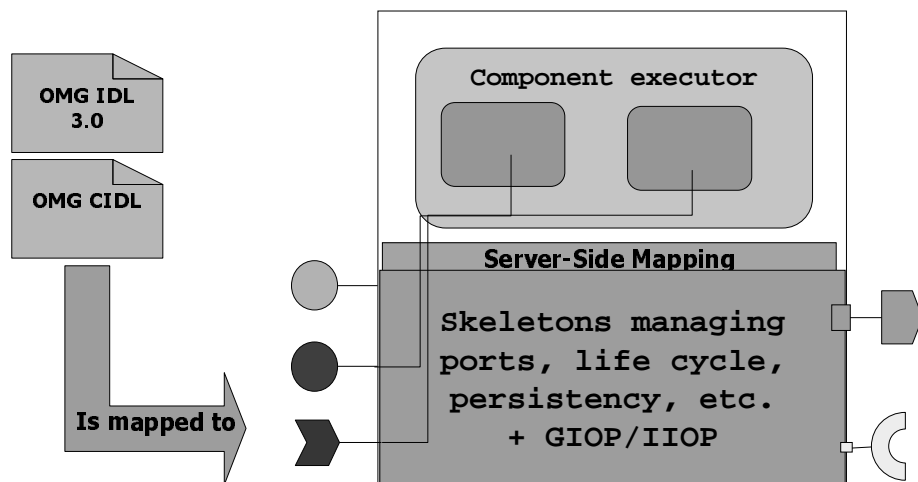
Exemple en OMG IDL

```
module Couplage {  
  
    component Coupleur {  
        provides Echange code1;  
        provides Echange code2;  
    };  
  
    home CoupleurHome manages Coupleur {};  
  
    component Code {  
        provides Comportement usager;  
        uses Echange coupleur;  
    };  
  
    home CodeHome manages Code {};  
  
};
```

Programmation

- Component Implementation Framework (CIF)
- Component Implementation Definition Language (OMG CIDL)
 - Catégorie des composants,
 - Liens avec les fabriques et les implantations,
 - Liens avec les gérants de persistance.
- Règles de projection de l'OMG IDL mapping

Component Implementation Framework



[extrait du tutoriel CCM de l'OMG]

Catégories de composants

COMPONENT CATEGORY	CONTAINER IMPL TYPE	CONTAINER TYPE	EXTERNAL TYPE	EJB BEAN EQUIVALENT
<i>Service</i>	Stateless	Session	Keyless	Session (stateless)
<i>Session</i>	Conv	Session	Keyless	Session (stateful)
<i>Process</i>	Durable	Entity	Keyless	-----
<i>Entity</i>	Durable	Entity	Keyfull	Entity

[extrait du tutoriel CCM de l'OMG]

Exemple en CIDL

```
module CouplageCIDL {  
  
  composition session CodeImpl {  
    home executor CodeHomeImpl {  
      implements Couplage::CodeHome;  
      manages CodeSessionImpl;  
    };  
  };  
  
  composition session CoupleurImpl {  
    home executor CoupleurHomeImpl {  
      implements Couplage::CoupleurHome;  
      manages CoupleurSessionImpl;  
    };  
  };  
};
```

Exemple de client

```
// Creation d'une instance de composant à partir de sa fabrique
Couplage::CoupleurHome_var coupleur_home
    = Couplage::CoupleurHome::_narrow(reference);
Components::CCMObject_var coupleur0
    = coupleur_home->create_component();
CORBA::Object_var com = coupleur0->provide_facet("code1");
Couplage::Echange_var _coupleur0
    = Couplage::Echange::_narrow(com);
try {
    coupleur0->configuration_complete();
} catch (Components::InvalidConfiguration &ex) {
    ...
}
```

Exemple de composant (déclaration)

```
// classe de la fabrique
class CodeHomeImpl_cimpl: virtual public CouplageCIDL::CodeHomeImpl{
public:
    CodeHomeImpl_cimpl();
    ~CodeHomeImpl_cimpl();
    virtual Components::Executors::ExecutorSegmentBase_ptr
        create_executor_segment(Components::Extended::SegmentId segid)
        throw(CORBA::SystemException);
};
// classe du composant
class CodeSessionImpl_cimpl: virtual public CouplageCIDL::CodeSessionImpl {
public:
    CodeSessionImpl_cimpl();
    ~CodeSessionImpl_cimpl();
    virtual void configuration_complete()
        throw(Components::InvalidConfiguration, CORBA::SystemException);
    virtual PortableServer::Servant _get_facet_usager();
    virtual void charger_comportement(const char* nom)
        throw(CORBA::SystemException);
};
```

Exemple de composant (implantation)

```
// implantation des methodes d'instance de la fabrique
Components::Executors::ExecutorSegmentBase_ptr
CodeHomeImpl_cimpl::create_executor_segment(Components::Extended::SegmentId segid)
throw(CORBA::SystemException){
    Components::Executors::ExecutorSegmentBase_ptr s =
        new CodeSessionImpl_cimpl();
    s->init(this);
    return s;
}
// implantation des methodes d'instance du composant
PortableServer::Servant CodeSessionImpl_cimpl::_get_facet_usager() {
    return new POA_Couplage::Comportement_tie<CodeSessionImpl_cimpl>(*this);
}
void CodeSessionImpl_cimpl::configuration_complete()
throw(Components::InvalidConfiguration, CORBA::SystemException) {
    _coupleur = this->get_connection_coupleur();
    if (CORBA::is_nil(_coupleur)) throw Components::InvalidConfiguration();
}
void CodeSessionImpl_cimpl::charger_comportement(const char* nom) ...
```

Conditionnement/déploiement/exécution

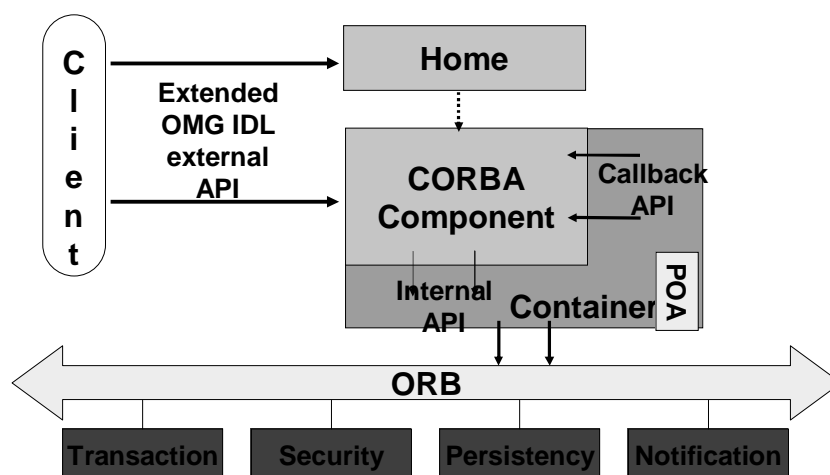
- Conditionnement et déploiement
 - Format d'archive,
 - Pas standardisé ?
 - Descripteurs de déploiement.
- Exécution
 - Architecture de conteneurs.

XML Descriptors Overview

- Software Package Descriptor (.csd)
 - Describes contents of a component software package
 - Lists one or more implementation(s)
- CORBA Component Descriptor (.ccd)
 - Technical information mainly generated from CIDL
 - Some container managed policies filled by user
- Component Assembly Descriptor (.cad)
 - Describes initial virtual configuration
 - homes, component instances, and connections
- Component Property File Descriptor (.cpf)
 - name/value pairs to configure attributes

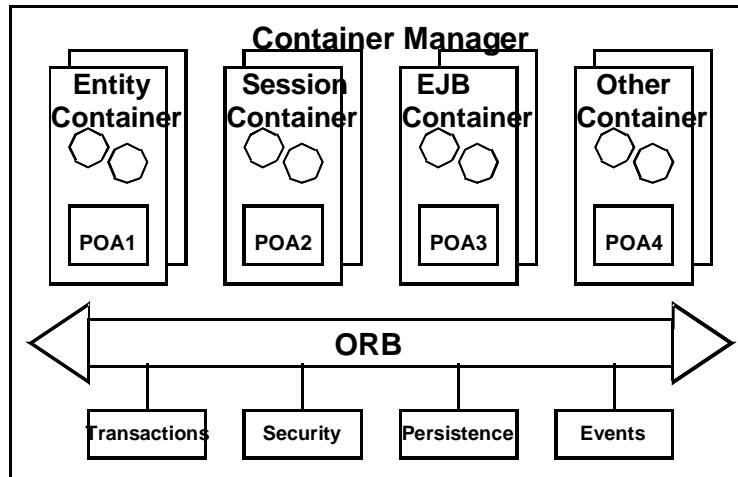
[extrait du tutoriel CCM de l'OMG]

The Container Architecture



[extrait du tutoriel CCM de l'OMG]

The Container Server Architecture



[extrait du tutoriel CCM de l'OMG]

Sommaire

- Approche composants
- Rappels sur CORBA Objet
- Spécification CORBA Composant
- Perspectives

Comparaison avec EJB et .NET

- Like SUN Microsystems's Enterprise Java Beans (EJB)
 - CORBA components created and managed by homes
 - Run in containers managing system services transparently
 - Hosted by application component servers
- Like Microsoft's Component Object Model (COM)
 - Have several input and output interfaces
 - Both synchronous operations and asynchronous events
 - Navigation and introspection capabilities
- Like Microsoft's .NET Framework
 - Could be written in different programming languages
 - Could be packaged in order to be distributed

[extrait du tutoriel CCM de l'OMG]

Open Source CCM Implementations

- OpenCCM from LIFL & ObjectWeb
 - Java on ORBacus 4.1 & OpenORB 1.2.1
 - <http://www.objectweb.org/OpenCCM/>
- MicoCCM from FPX & Alcatel
 - C++ on MICO
 - <http://www.fpx.de/MicoCCM/>
- CIF from Humboldt University
 - C++ on ORBacus 4.1
 - <http://sourceforge.net/projects/cif>

[extrait du tutoriel CCM de l'OMG]

Commercial CCM Implementations

- Qedo from Fraunhofer FOKUS
 - C++ on MICO & ORBacus 4.1
 - <http://qedo.berlios.de>
- EJCCM from CPI Inc.
 - Java on OpenORB 1.3.x
 - <http://www.ejccm.org>
- K2 from iCMG
 - C++ on various ORBs
 - <http://www.icmgworld.com>

[extrait du tutoriel CCM de l'OMG]