Introduction aux composants logiciels

Lionel Seinturier
Université Lille 1 – LIFL & INRIA Futurs Jacquard

25/8/2006

1. Introduction

Contexte : ingénierie du système/intergiciel (*middleware*)

Passé (année 1990) : objet mouvance « à la CORBA »

Besoins
- configuration
- déploiement
- empaquetage (*packaging*)
- assemblage
- dynamicité
- gestion des interactions et des dépendances

Présent : plusieurs tendances
- composant, aspect, MDE, réflexivité

1. Introduction

Composant vs objet

- plus haut niveau abstraction
- meilleure encapsulation, protection, autonomie
  - programmation + systématique + vérifiable
- communications plus explicites
  - port, interface, connecteur
- connectables
  - schéma de connexion (ADL) : « plan » applicatif
- séparation « métier » - technique
- meilleure converture du cycle de vie
  - conception, implémentation, *packaging*, déploiement, exécution

1. Introduction

Définition composant

- 1ère apparition terme [McIlroy 68]
- 30 ans + tard : Sun EJB, OMG CCM, MS .NET/COM+, …
- recensement [Szyperski 02] : 11 définitions +/- ≡

A component is a unit of composition with **contractually specified interfaces** and **context dependencies** only. A software component can be **deployed** independently and is subject to **composition** by third parties. [Szyperski 97]

1. Introduction

Nombreux modèles de composant (20+)

- construits au-dessus Java, C, C++
- EJB, Java Beans, CCM, COM+, JMX, OSGi, SCA, CCA, SF
- Fractal, K-Component, Comet, Kilim, OpenCOM, FuseJ, Jiazzi, SOFA, ArticBeans, PECOS, Draco, Wcomp, Rubus, Koala, PACC-Pin, OLAN, Newton
- Bonobo, Carbon, Plexus, Spring
- au niveau analyse/conception : UML2

1. Introduction

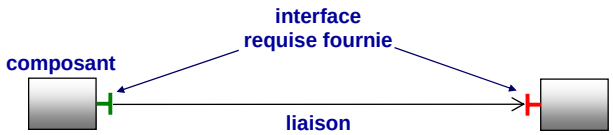
Conséquence de la multiplicité des modèles

- multiplicité du vocabulaire
  - composant, *bean*, *bundle*
  - interface/liaison, port/connecteur, facette, puits, source
  - requis/fourni, client/serveur, export/import, service/référence
  - conteneur, membrane, services techniques, contrôleur
  - *framework*, serveur d'applications
- grande variabilité dans les propriétés attachées aux notions
- exemples
  - Fractal : composant, interface, liaison, client/serveur
  - CCM : composant, facette, port, puits, source
  - UML 2 : composant, fragment, port, interface
  - OSGi : *bundle*, *package* importé/exporté, service/référence
- un même terme peut avoir des acceptations ≠ selon les modèles
- ➢ qualifier les notions (« connecteur au sens … »)
- ➢ pas toujours facile de définir les équivalences

1. Introduction

1ère grande catégorie de modèle de composants

- triptyque : composant, interface, liaison
  - un composant fourni et/ou requiert une ou plusieurs interfaces
  - une liaison est un chemin de communication entre une interface requise et une interface fournie



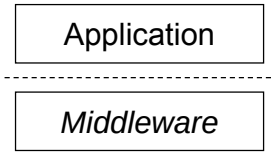
1. Introduction

2ème grande catégorie de modèle de composants

- triptyque : composant, port, connecteur
  - un composant fourni et/ou requiert une ou plusieurs ports
  - un connecteur implémente un schéma de communication entre des composants (client/serveur, diffusion, etc.)
  - un composant est relié à un connecteur via un ou plusieurs ports

- connecteur ≈ liaison avec comportement
- ➢ on peut considérer connecteur = composant (de communication)
- ➢ composant, interface, liaison

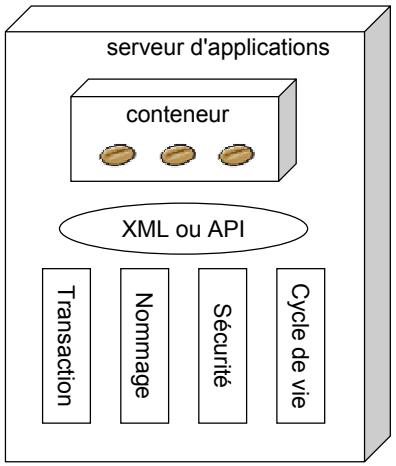
Classification des modèles pour système/intergiciel



- application : EJB, CCM, .NET/COM+, SCA, Spring
- *middleware* : Fractal, JMX, OpenCOM, OSGi
- *middleware* componentisé pour applications à base de composants
  - JonasALaCarte : Fractal + EJB [Abdellatif 05]
  - OSGi + EJB [Desertot 05]

Modèles EJB, CCM, .NET/COM+

- focus sur séparation métier/technique
- cibles : applications Internet, système d'information
- *packaging* ++
- déploiement ++
- architecture pauvre
- services figés, pas adaptables



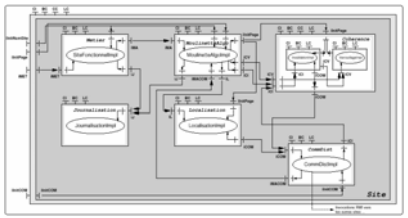
Quelques « poncifs » à propos des composants

- COTS Commercial Off The Shelf
  - vieux discours (voir procédures, fonctions, objet, …)
  - taille applis ↗ donc besoin : toujours plus de réutilisation
  - mais *quid* de la contractualisation ?
- « *Programming in the large* »
  - vs « *programming in the small* » (objet)
  - vrai d'un certain point de vue
  - mais nouveaux points à traiter (liés au non fonctionnel par ex.)

Notion d'architecture logicielle

A software architecture of a program or computing system is the structure or **structures** of the system, which comprise **software components**, the externally visible **properties** of those components, and the **relationships** among them. [Bass 98]

- langage : ADL
- souvent en XML
- ➢ *survey* : [Medvidovic 00]



Complémentarité

- architecture : construite à partir de composants
- composants : assemblés pour construire une architecture

2 visions complémentaires

- architecture : *top-down*
- composants : *bottom-up*