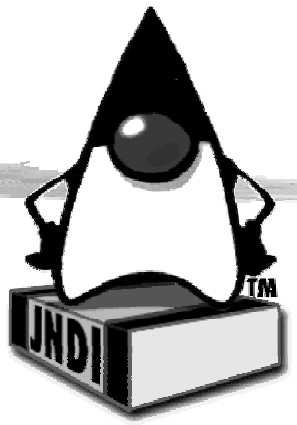
Les produits Java

Sylvain LECOMTE
Didier DONSEZ

Les produits Java



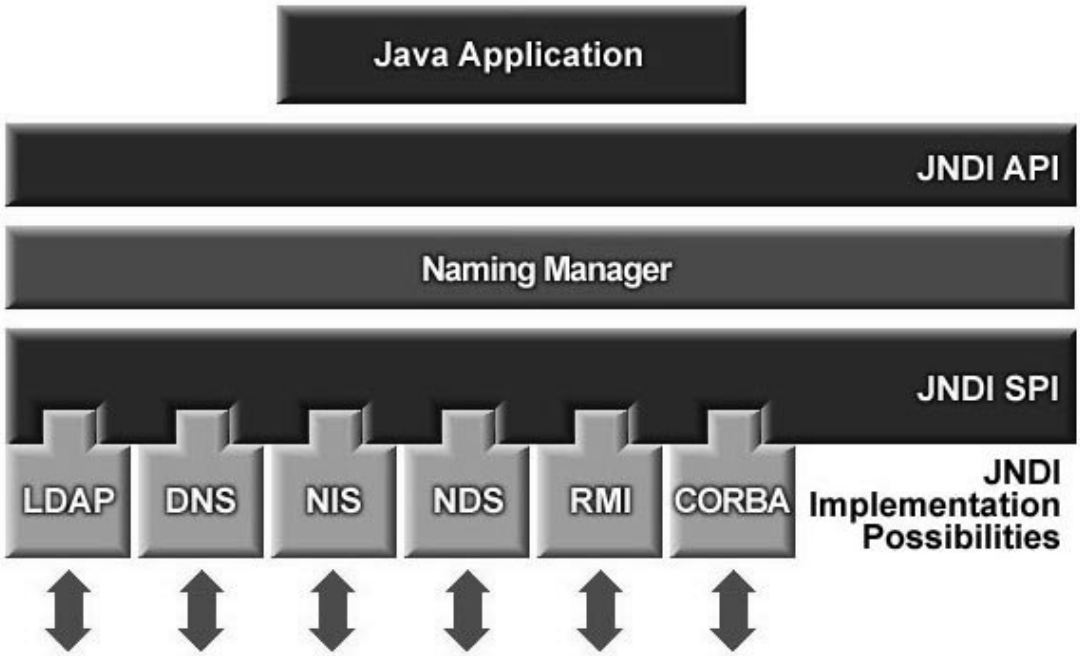
JNDI
Java Naming and Directory Interface

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Objectifs

- Fournir un Service de nommage :
  - Recherche d 'objets par leur nom
  - classification par directory
- Fournir une interface avec les systèmes de nommage existants (Service provider interface)
  - LDAP
  - Naming Services CORBA
  - DNS
  - Naming Services RMI
  - File System

Architecture



Les produits Java

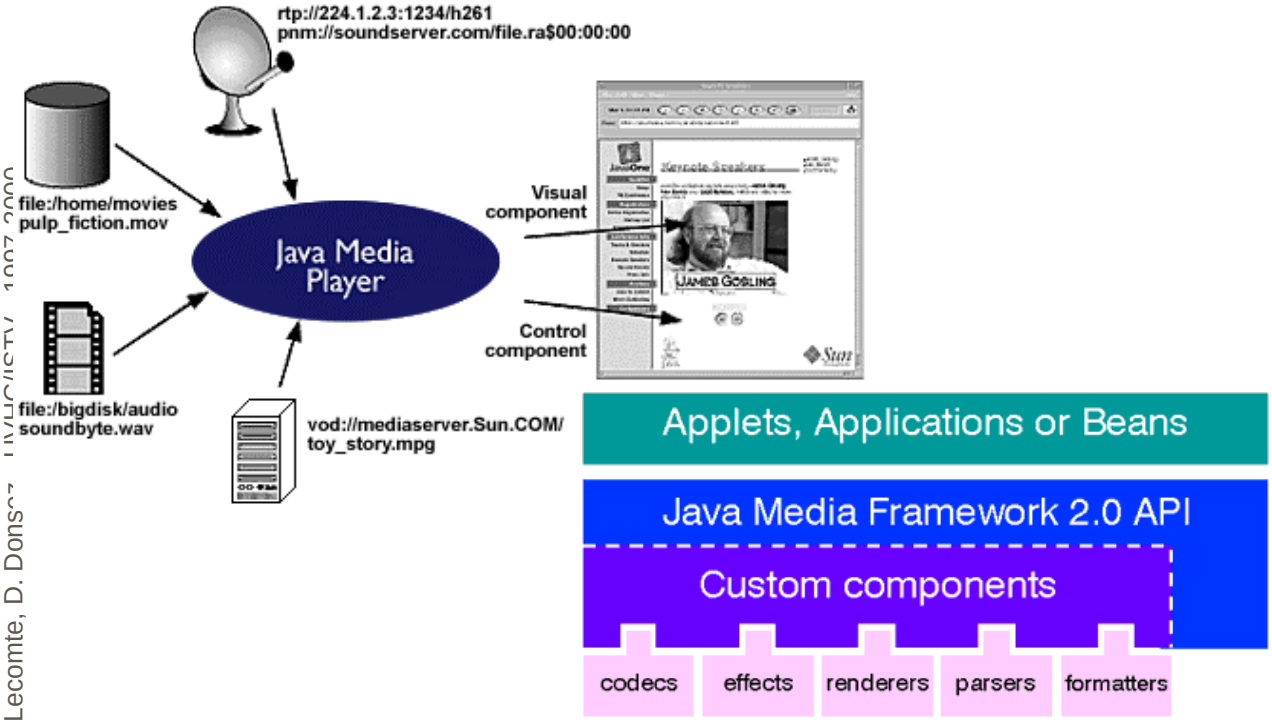
Java Media Framework

Objectifs

- Java Media Framework (JMF) est une architecture unifiée pour la synchronisation, le traitement, l'affichage de données temporelles comme les données audio, la vidéo, le format MIDI, etc. à l'intérieur d'applications indépendantes ou d'applets.
- utilise Java 1.1 et suivants
- développé par Sun, Silicon Graphics et Intel.
- constitué de 3 étapes : Player, Capture, Conference.

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Architecture



Java Media Player

- Java Media Player comporte la synchronisation, l'exécution, la présentation et le stockage de données temporelles compressées.
- supporte les formats :
  - pour l'audio: AIFF, AU, DVI, G.723, GSM, IMA4, MIDI, MPEG-1 Layer 1/2, PCM, RMF, WAV
  - pour la vidéo: Apple Graphics (SMC), Apple Animation (RLE) Cinepak, H.261, H.263, Indeo 3.2, Motion-JPEG, MPEG-1, Uncompressed
  - les formats de fichiers : AVI, QuickTime, Vivo
  - les protocoles File, FTP, HTTP, RTP (RFC 1889/1890)

Les produits Java

JCA/JCE
Java Cryptography Architecture
Java Cryptography Extension

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Objectifs

- Fournir une architecture d 'interfaces à des outils cryptographiques :
  - Chiffrement à clé publique ou à clé privée
  - key generation and key agreement
  - Message Authentication Code (MAC) algorithms.
- Notion de SPI (Service Provider Interface)
  - SP = un fournisseur de matériel cryptographique
- L 'architecture permet d 'utiliser un ou plusieurs SPI simultanément dans une appli

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Les produits Java

JDBC

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Objectifs

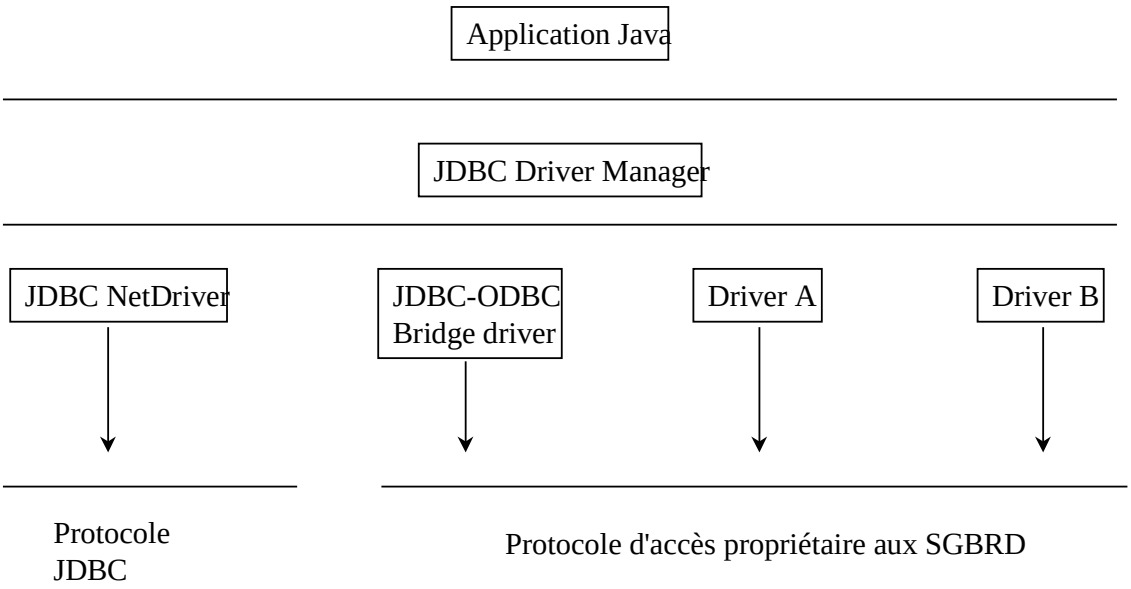
- Fournir un accès homogène aux SGBDR
- Abstraction des SGBDR cibles
- Requêtes SQL
- Simple à mettre en œuvre
- *Core* API (1.1)

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

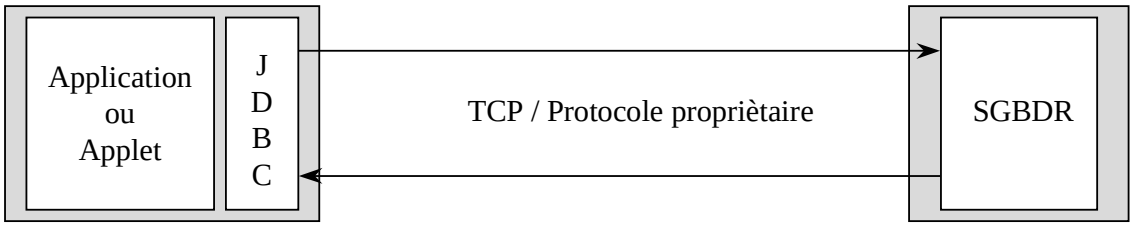
Fonctionnement

- JDBC interagit avec le SGBDR par un *driver*
- Il existe des *drivers* pour Oracle, Sybase, Informix, DB2, ...
- 4 types de drivers :
  1. *Bridge* ODBC (fourni avec JDBC)
  2. *Native-API partly-Java driver*
  3. *JDBC-Net all-Java driver*
  4. *Native-protocol all-Java driver*
- 1. et 2. nécessitent des architectures 3-tiers pour les applets

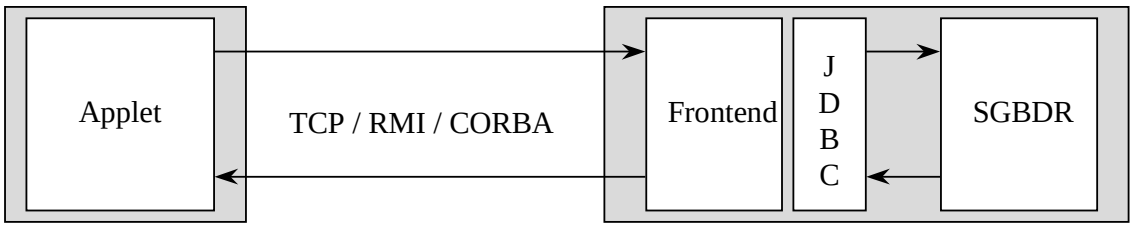
Drivers JDBC



Architectures 2-tier et 3-tier



Architecture 2-tier



Architecture 3-tier

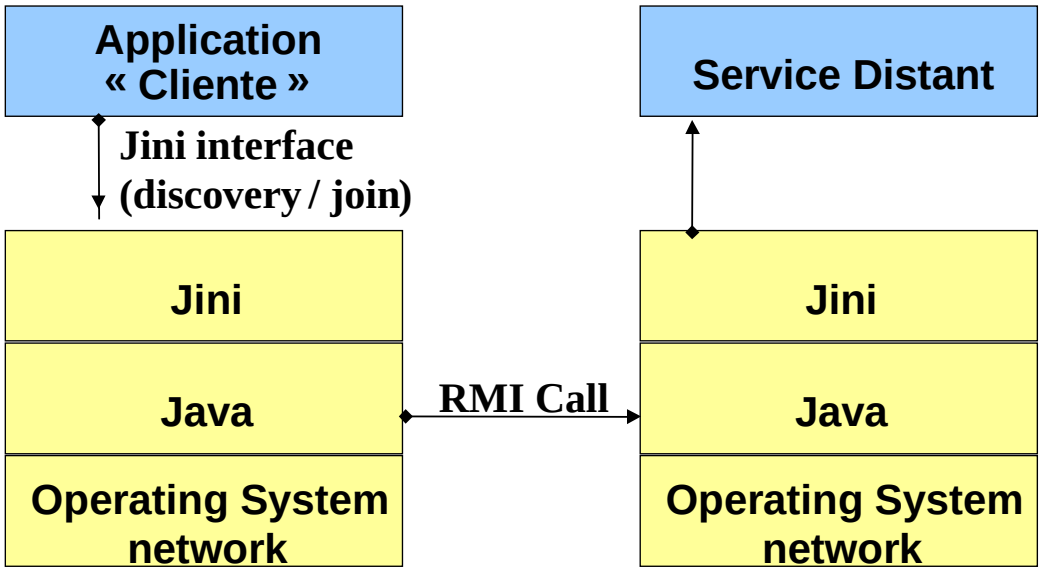
Les produits Java

Jini

Objectifs

- Jini permet :
  - de s'abstraire des protocoles réseaux qui peuvent être utilisés pour faire communiquer ensemble plusieurs composants (TCP/IP, IrDa, WAN)
  - de s 'abstraire des systèmes d'exploitation (W95, Unix, …)
  - exemple :
    - une application cliente sera la même sur un PC utilisant Windows et TCP/IP, et sur un appareil photo digital, utilisant un système d'exploitation javaOS et une connexion réseau IrDa

Architecture



Les produits Java

JavaOS

Vision de Alcatel...

"The utilization of JavaOS for Consumers 3.0 software in Alcatel's screenphone will have a significant impact in the world of advanced consumer electronic products. Our intention is to bring value to end users by making their life easier. With the Alcatel screenphone, access to the Internet and Internet services has never been easier, more immediate and more intuitive. We are on the verge of a revolution in the way people communicate."

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

JavaOS

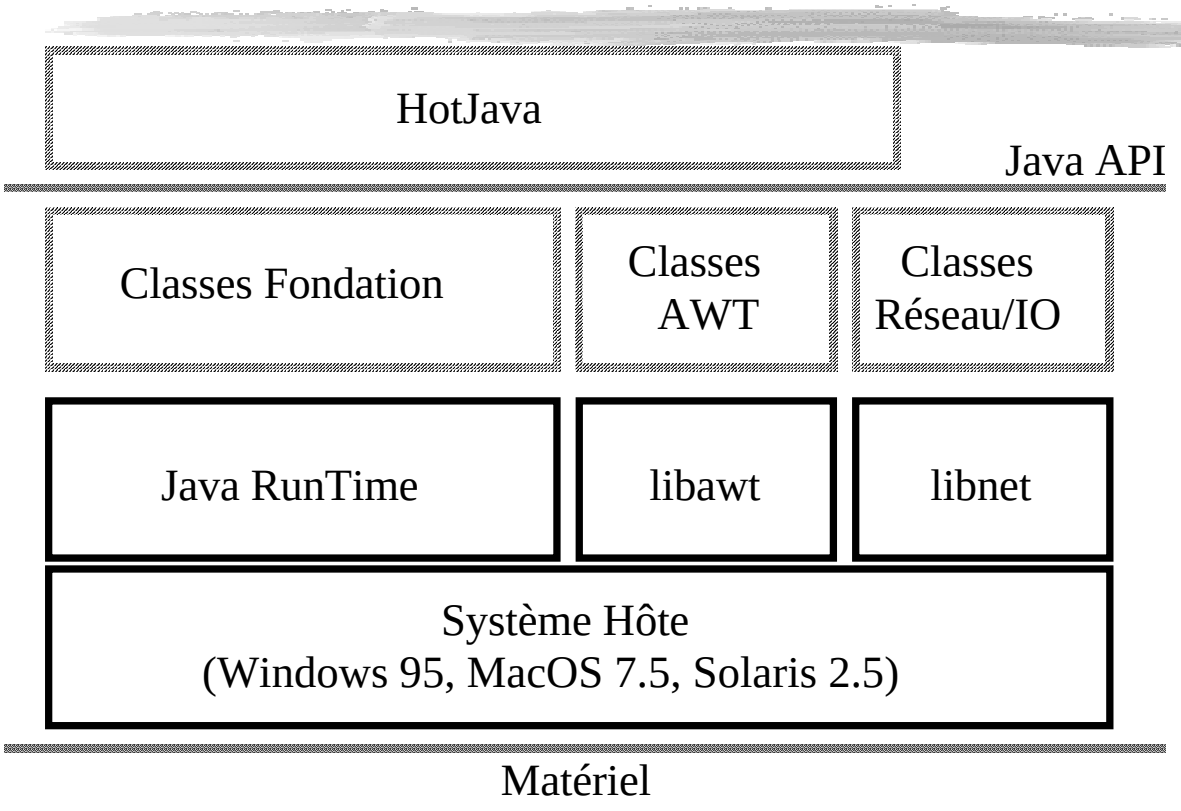
Système d'exploitation Petit et Efficace

- Cibles
  - Internet/Intranet Network Computer
  - assistant personnel (PDA), téléphone mobile,
  - console de jeux, STB,
  - électroménager, domotique, ...
- Performances
  - Mémoire: 4 Mo en ROM (1Mo de police)
    RAM: 512K pour OS +
    512K pour les pages ou les applets
  - Speed: CaffeineMark benchmark
- Concurrence
  - MicroSoft Windows CE, Symbian, PalmOS, ...

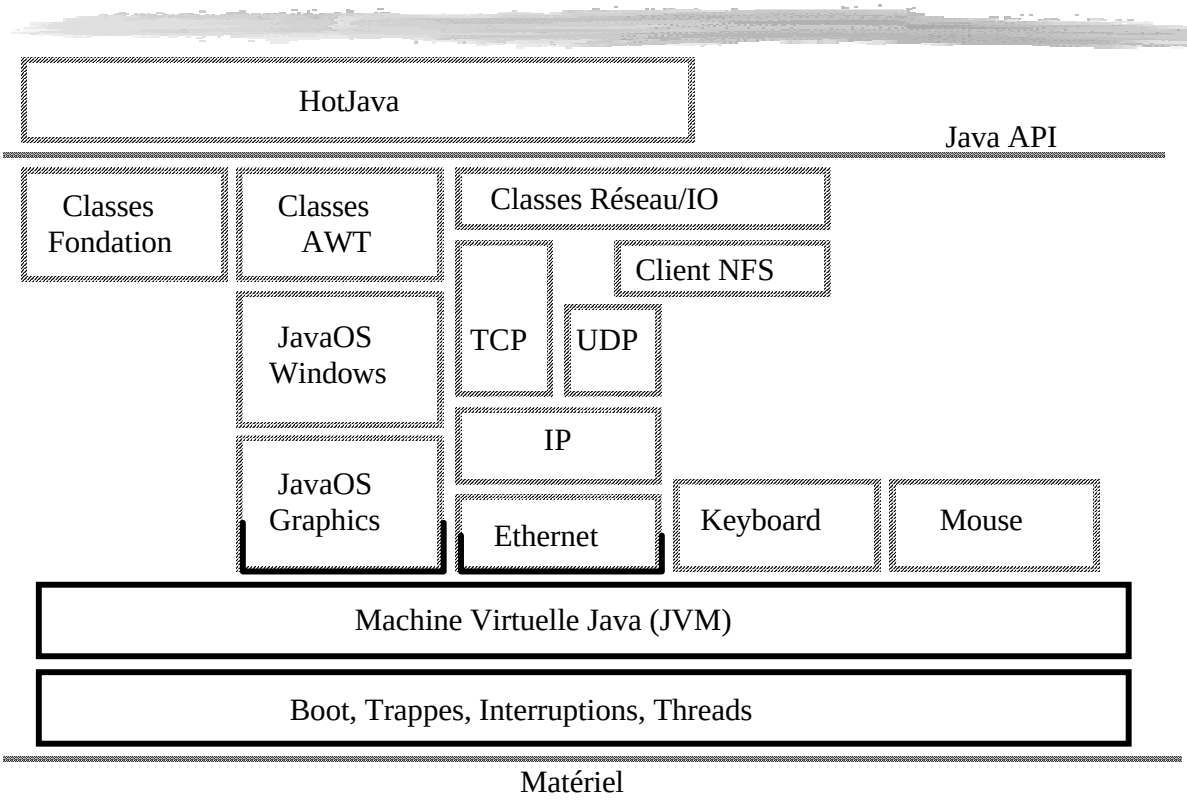
JavaOS

- Noyau Java *(basé sur la JVM)*
  - Threads
  - MMU seulement pour l'allocation contiguë
- Device Drivers
  - écrit en C et Java, JDDI
  - Network Protocol Suite
    - IP, ICMP, TCP, UDP, DNS (hostname) et NIS (login, passwd), Reverse ARP, DHCP (Terminal X / Diskless WS)
- Fenêtrage et Graphique
  - Tiny Abstract Windowing Toolkit

Java sur Système hôte



Java en Standalone



JavaOS

- Personal Java
  - JavaOS pour PDA,Téléphone intelligent,GSM,
  - Electronic consuming (Gizmo)
- Java PC
  - Recyclage des PC obsolètes en NC
    - i286, i386, … avec 4 Mo RAM
- Embedded Java

JavaOS :
un vrai Système d'Exploitation ?

- *NON*
  - pas de File System
  - pas de Memoire Virtuelle
  - pas d'Espace Adressage protégé
  - un seul langage de programmation : Java
- *OUI*
  - boot, login, drivers et protocoles reséau
  - applets concurrentes, milliers d'application

Les Network Computers (NC) - i

- Administration difficile des Postes PCs
  - Ownership Cost = 17000FF pour un poste PC et par an !! (source Gartner Group, 1997)
    - achat, maintenance, logiciel, formation, ...
- Roman Photo
  - Oracle et Sun attaquent Intel et MicroSoft
    - Java et les NCs *(9000 FF par poste et par an)*
  - Intel et MicroSoft ripostent
    - Net PC et Win NT 5

Les Network Computers (NC) - ii

- Définition
  - processeur adapté (Java Chips) , peu de RAM, modem/ethernet
- Application
  - Intranet / Extranet, Administration Simplifiée
- Produits
  - Oracle, Sun, IBM et les fabricants de terminaux X NCD
  - Compatibilité : NC Reference Profile

SetTopBox

- le « NC dans tous les foyers»
  - mariage télématique et télévision
    - GUI pilotée par une télécommande IR
  - Hardware
    - démodulateur satellite/cable, décompression hard MPEG2 (download), modem RTC/ADSL (upload), lecteur(s) de carte à puce (carte abonnée, carte bancaire, …)
- Télévision du futur
  - Pay per View *(donc Carte à Puce)*
  - Video à la demande *(donc Serveur de VOD)*
  - …

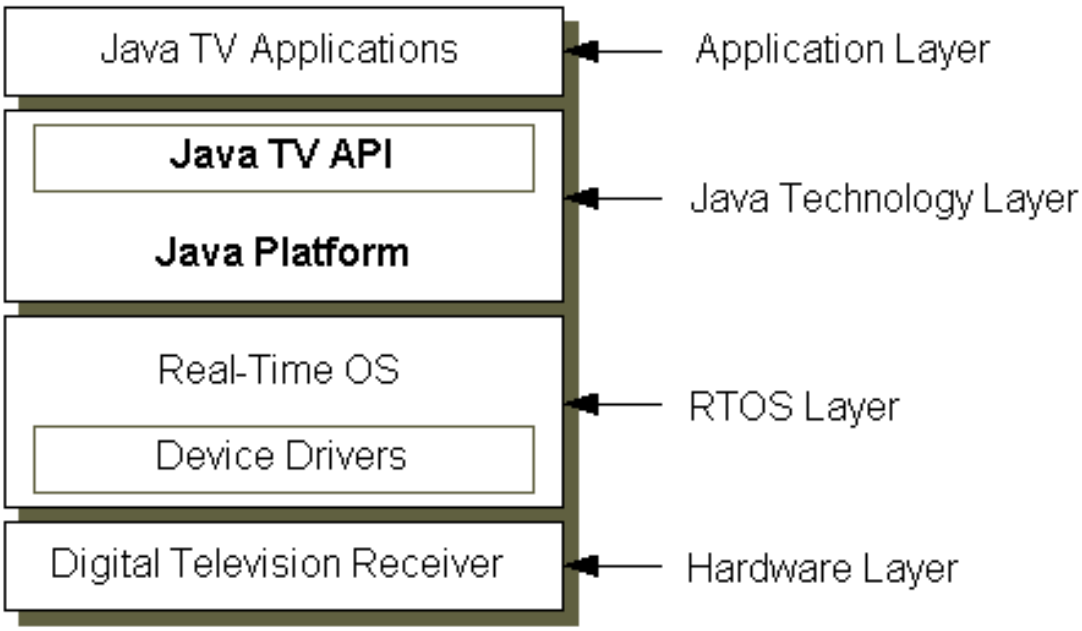
Les produits Java

JavaTV

Objectifs

- La TV devient numérique et interactive
- APIs Java pour STB
  - extension de la plate-forme java pour gérer certaines fonctionnalités spécifiques aux STB et aux télévisions.
  - reprend les travaux de normalisation de DAVICS et de DVB
  - FS de type Carrousel sur MPEG2/TS, Télécommande …
- Xlet
  - La STB télécharge et exécute des Xlets
  - Les Xlets sont les applications
    - Grille de programme, Sélection des canaux
    - Service interactive (TV achat, …)
- Acteurs
  - Sun, Canal + Technologies, OpenTV, ...

Utilisation

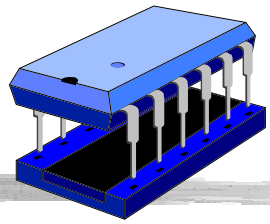


Les produits Java

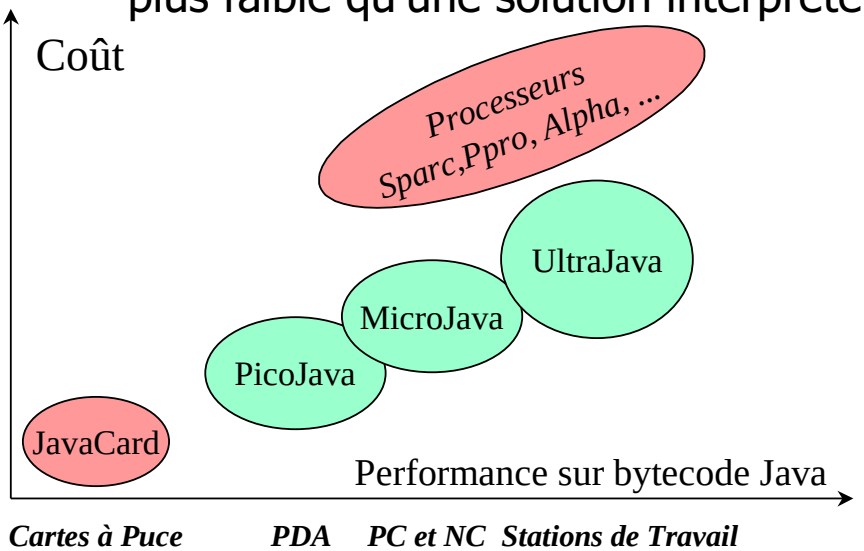
Chips et Benchmarks

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

JavaChip
la JVM dans le Silicium



rapport coût / performance
plus faible qu'une solution interprété



S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Benchmarks pour Java

- CaffeineMark
  - Evaluer les performances des JVMs ou des JavaChip
- Exemple
  - MicroJava701/200 13332 CaM

Les produits Java

RMI

Remote Method Invocation

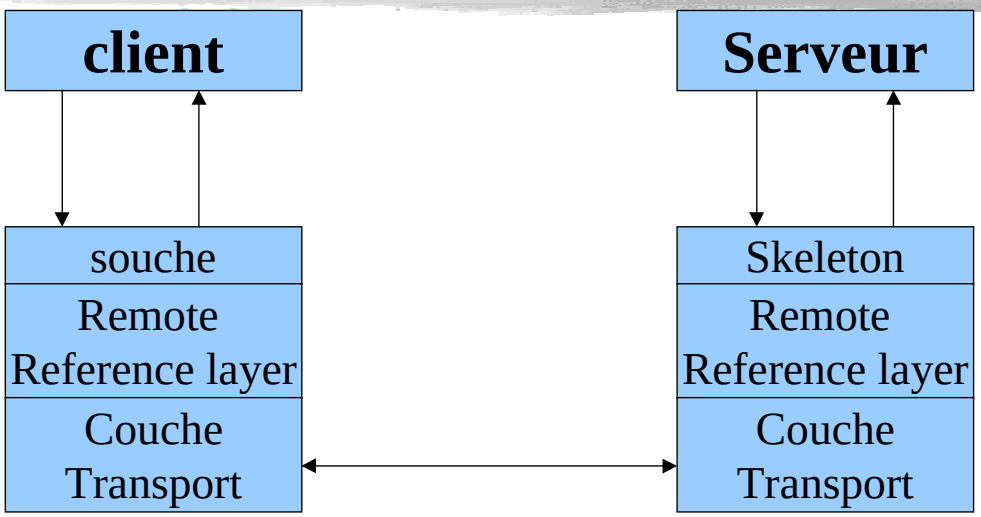
Objectifs

- Manipuler des objets sur des machines distantes de manière similaire aux objets sur la machine locale
- Permet à un objet local d 'invoquer des méthodes sur un objet distant (nécessite 2 JVM)
- Corba allégé :
  - Avantage : plus simple à utiliser
  - inconvénient : tout JAVA (client et serveurs)
- *Core* API (1.1)
  - Gratuit (contrairement à Corba)

Fonctionnement

- RMI propose :
  - Un GC distribué
  - la gestion des représentants locaux d 'objets distants, et leur activation
  - la liaison avec la couche transport et la gestion des sockets
  - de conserver la syntaxe d 'invocation des objets locaux pour les objets distants

Architecture



La souche contient les représentants locaux de références d ’objets distants
Une invocation de méthode distante est traitée par la RRL
Les objets distants sont repérés par des références gérées par la couche skeleton

Programmer avec RMI...

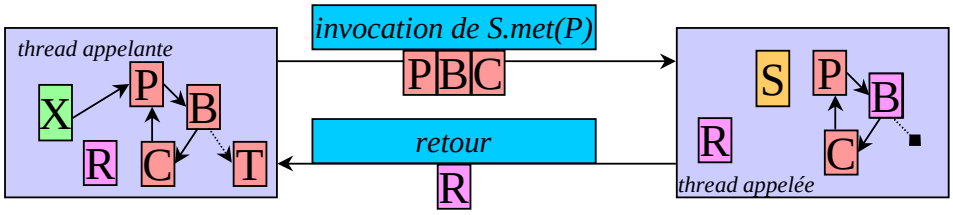
- Pour créer une application avec RMI :
  - 1- définir les interfaces pour les classes distantes.
  - 2- Créer et compiler les implémentations de ces classes.
  - 3- Créer les classes pour la souche et le skeleton à l'aide de la commande rmic.
  - 4- Créer et compiler une application serveur.
  - 5- lancer le rmiregister et lancer l'application serveur.
  - 6- Créer et compiler un programme client qui accède à des objets distants du serveur.
  - 7- lancer ce client.

Java RMI
Remote Method Invocation

- Invocation d'une méthode
  sur un objet situé sur un autre site
  - en général, un objet du serveur pour une applet
  - le serveur est implanté par un objet
    - `java.rmi.server.UnicastRemoteObjet, ...`
  - le service est enregistré auprès d 'un « annuaire »
    - `java.rmi.Naming`
    - `rmi://monhote/mon service`
- Un paramêtre est un objet de la thread appelante
  - sérialisation de l 'objet
    - (envoi d'une image de l'objet vers l'appelé)

Java RMI
la Sérialisation

- opération d'empaquetage (.ser) des paramètres et du résultat (de type objet)
  - fermeture transitive du graphe d'objets Java
  - méthodes de objets envoyés
- Utilisation
  - Persistance, RMI, JavaSpace
  - Plateforme d'agents mobiles (Aglets, Odyssee, Voyager, ...)



Les produits Java

JavaIDL

De Java à CORBA

- Java IDL
  (http://www.javasoft.com/products/jdk/idl/index.html)
  - Le compilateur idltojava
    - Permet de générer les souches et squelettes pour travailler avec n'importe quel ORB
  - Le JDK 1.2 inclut une API CORBA et un ORB (org.omg.CORBA)
    - Permet à une application Java d 'invoquer des objets CORBA distants via IIOP
  - Un service de nommage est aussi disponible (org.omg.CosNaming)

Les produits Java

Java Transaction Service et OTS

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Transactions ?

- Suite d 'actions comprises entre
  - Begin_Transaction, qui marque le début d 'une transaction
  - et Commit_Transaction, ou Abort_Transaction, qui termine une transaction
- Exemple Typique :

```
Begin_Transaction
        CpteA+=100;
        CpteB-=100;
Commit_Transaction
```

- cette suite d 'actions répond aux propriétés ACID:
  - Atomicité : Tout ou rien
  - Cohérence : Données sont laissées dans un état cohérent
  - Isolation : Modifications pas visibles pendant la transaction
  - Durabilité : Les actions validées ne peuvent pas être perdues

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Validation d 'une Transaction

- Centralisé

  l'application et les données sont sur la même machine

  ⇨ Panne facile à traiter

  Validation à 1 phase

- Distribué

  l'application et les données sont sur 2 à N machines

  ⇨ Panne (partielle) difficile à traiter

  Validation à 2 phases (2PC : Two Phases Commit)

Transactions et objets répartis - OTS de l 'OMG

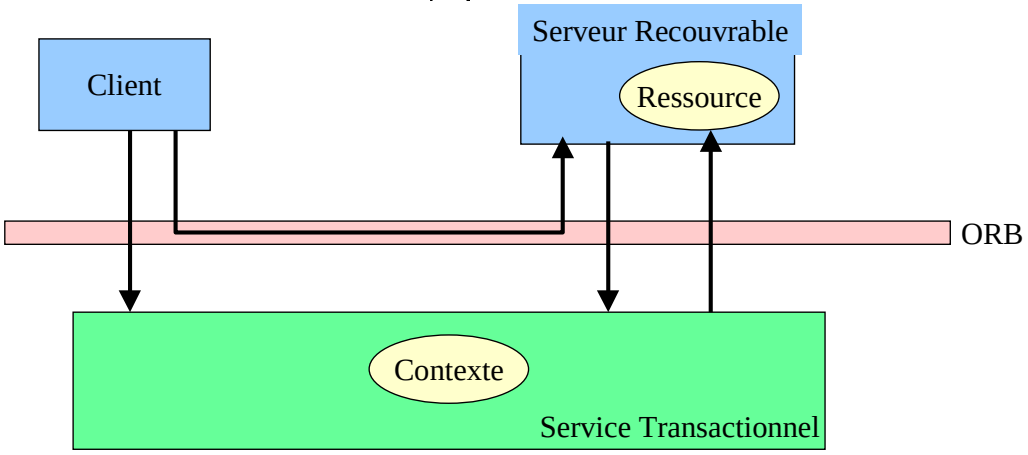
- La spécification décrit un ensemble d 'objets pour gérer des transactions
- Un objet transactionnel est un objet qui hérite d 'un objet défini par l 'OTS : TransactionalObject

```
interface bank.CosTransaction.TransactionalObject
{
...
void credit(in float amt)
...
}
```

- OTS fournit :
  - Les mécanismes d 'enregistrement et de gestion du contexte transactionnel
  - La validation à une ou à deux phases
  - Le modèle de transactions emboîtées

Architecture

- 3 éléments :
  - Le programme client qui commence et termine la transaction
  - Les serveurs dont les données sont affectées par la transaction
  - Le service transactionnel, qui coordonne la transaction



Les interfaces de communication

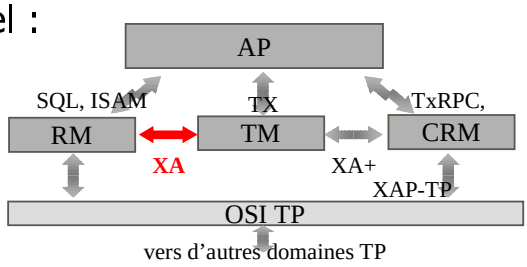
- L 'interface *TransactionFactory* : permet de créer une transaction auprès du service transactionnel
- L 'interface *Control*: permet à l 'application d 'interagir avec l 'OTS de manière explicite
- L 'interface *Terminator* : est utilisée pour demander la validation ou l 'annulation d 'une transaction
- L 'interface *Coordinator* : permet la coordination de la transaction entre les composants distribués
- L 'interface *Resource* : supporte le protocole de validation à 2 phases pour valider les modifications sur les données
- l 'interface *RecoveryCoordinator* : est utilisée par la ressource pour redémarrer après une panne
- L 'interface *Current* : permet un gestion transparente des transactions

JTS: une API pour OTS

- Sun fournit un mapping Java de OTS
  - 2 modules :
    - org.omg.CosTransactions
      - Qui offre les interfaces de gestion et de contrôle des transactions
    - org.omg.CosTSPortability
      - Qui offre un mécanisme d 'identification mutuelle entre un ORB et un OTS (pour faciliter le portage d 'un OTS sur un ORB

JTA : Java Transaction API

- 2 interfaces :
  - Mapping Java de l 'interface XA de X/OPEN (package javax.transaction.xa)
    - Rappel :



  - Une interface de démarcation de transaction (javax.transaction.UserTransaction)
    - Begin, Commit, Rollback, GetStatus, ...

Les produits Java

EJB - Enterprise JavaBeans

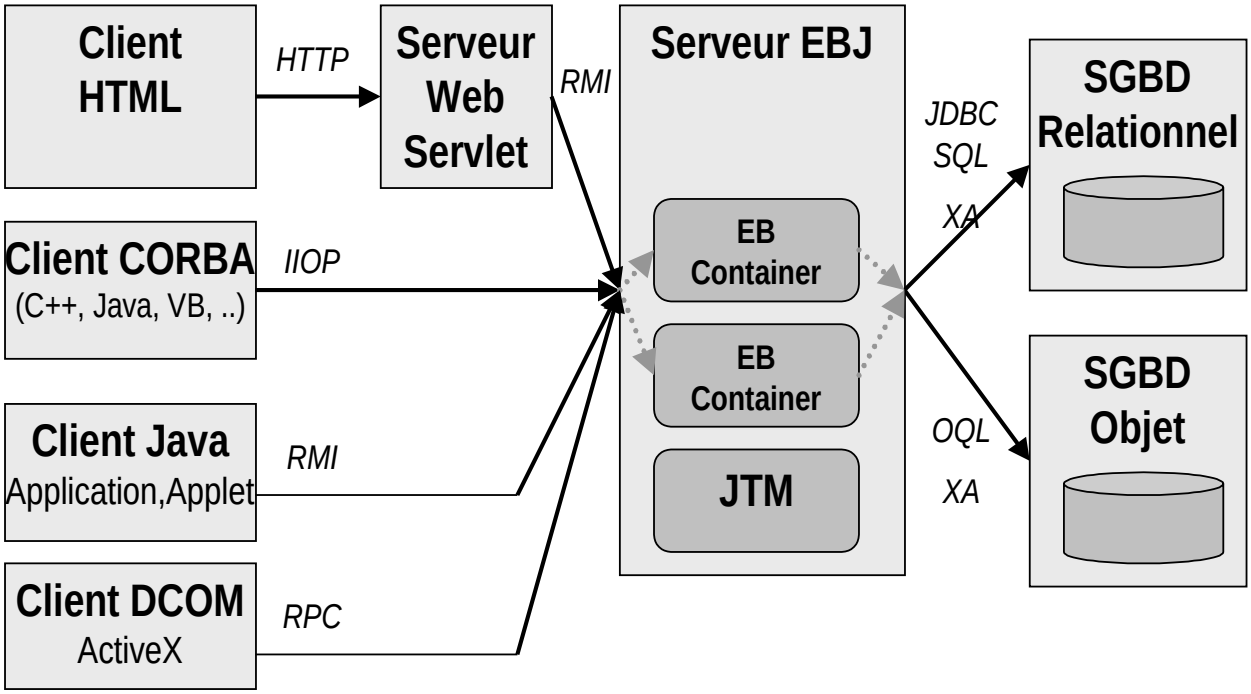
S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Objectifs

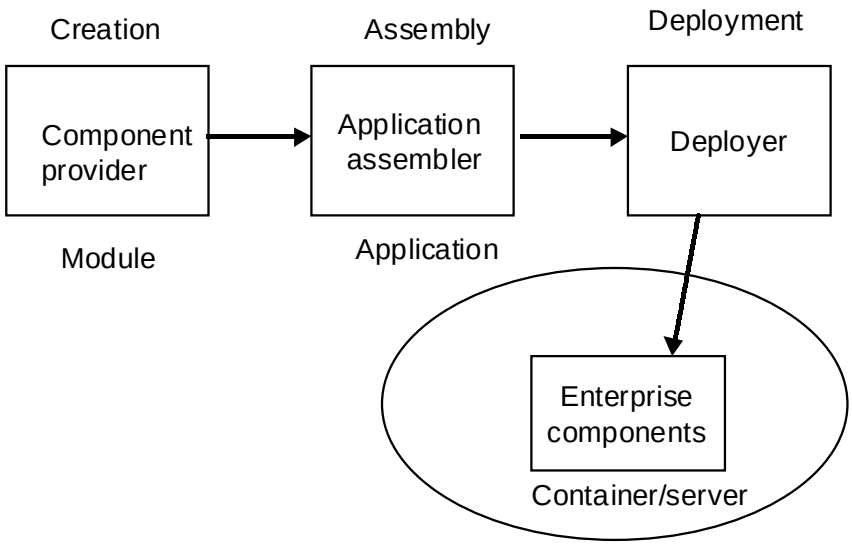
- Appliquer le modèle de programmation par composants aux systèmes distribués
  - But : simplifier la création des applications distribuées
- Applique la plupart des concepts étudiés :
  - Java Beans pour l 'utilisation de composants
  - JDBC pour la persistance dans des BD Relationnelles
  - RMI ou Corba (javaIDL) pour la distribution
  - JTS pour la gestion de transactions distribuées
- Solution :
  - Diviser en problèmes distincts la réalisation

S. Lecomte, D. Donsez - UVHC/ISTV - 1997-2000

Architecture de Serveur EBJ



Le processus de développement



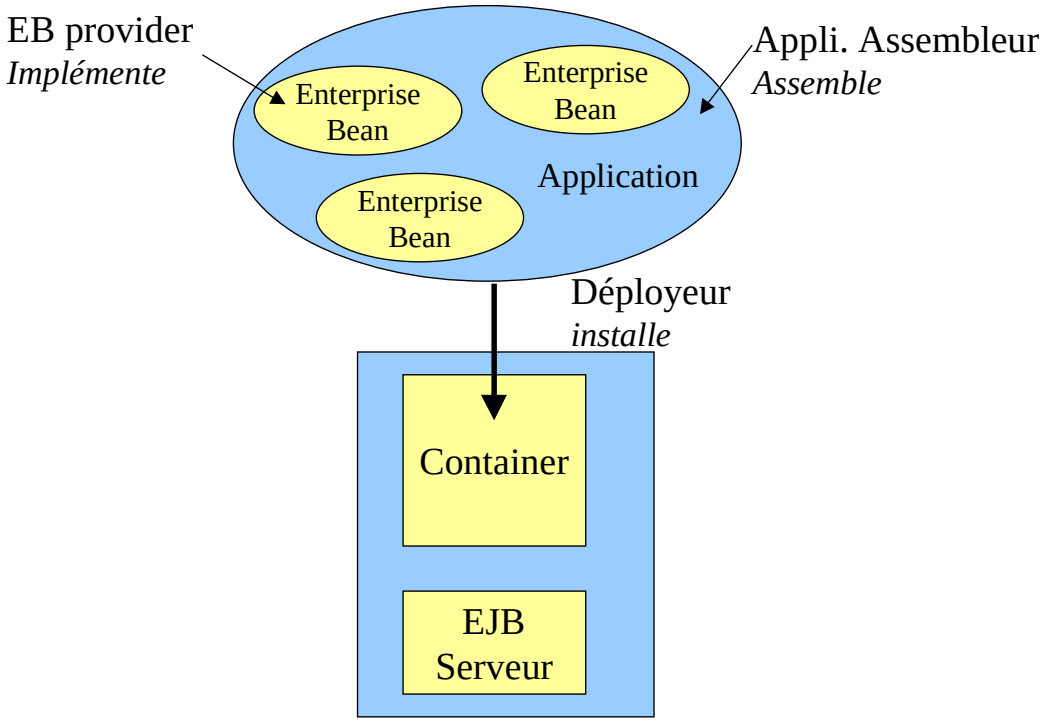
Différents rôles...

- Enterprise Bean provider
  - Spécialiste du domaine applicatif (ex : banque)
  - rôle : implémenter la tâche ans soucis de distribution, transaction sécurité, etc…
- l 'Assembleur d 'application :
  - fabrique l 'application en utilisant des blocs pré-fabriqués (Enterprise Beans, GUI Client, Applets…)
  - Il utilise uniquement les interfaces des EB, il n 'est pas concerné par l 'implémentation

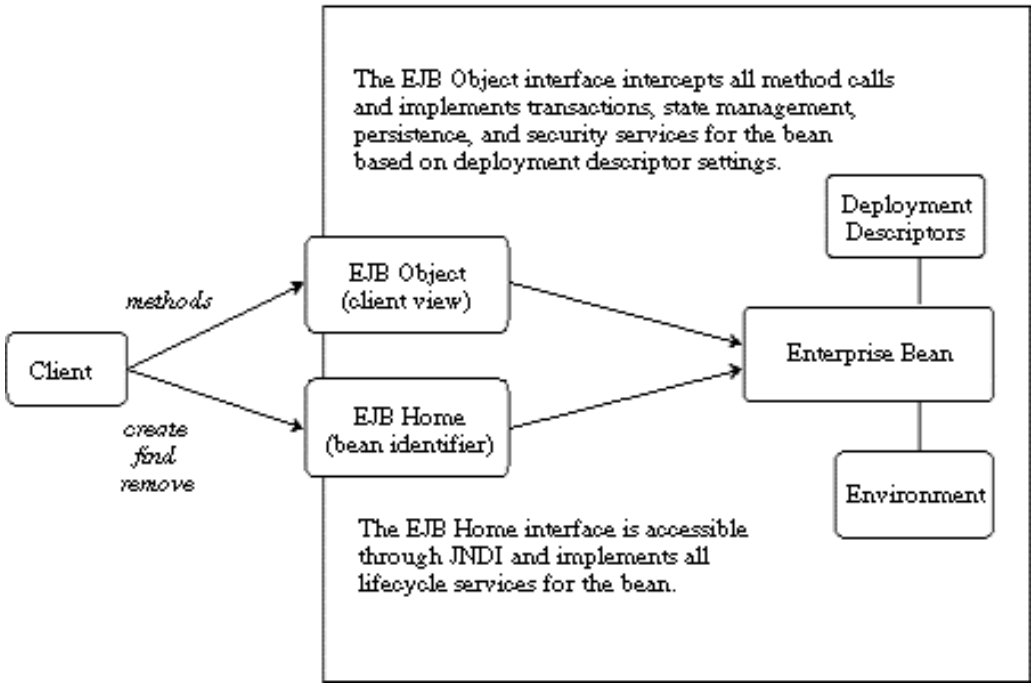
Différents rôles...

- Le déployeur :
  - installe l 'application en l 'adaptant à l 'environnement (Ex: politique de sécurité)
  - utilisation de descripteur de déploiement et intégration avec un logiciel de gestion des Ebs
- EJB Server Provider
  - Traite les problèmes liés à la distribution
- EJB Container Provider
  - Spécialiste des systèmes distribués, Transactions et de la sécurité
  - Container : colle entre les EB et le serveur d 'EB

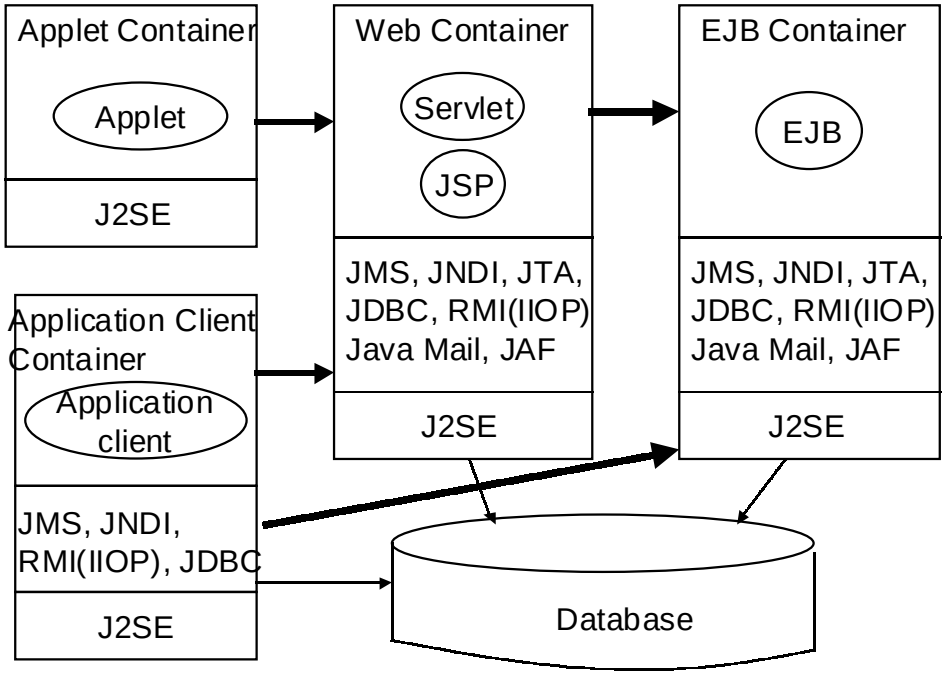
Récapitulatif



EJB Container



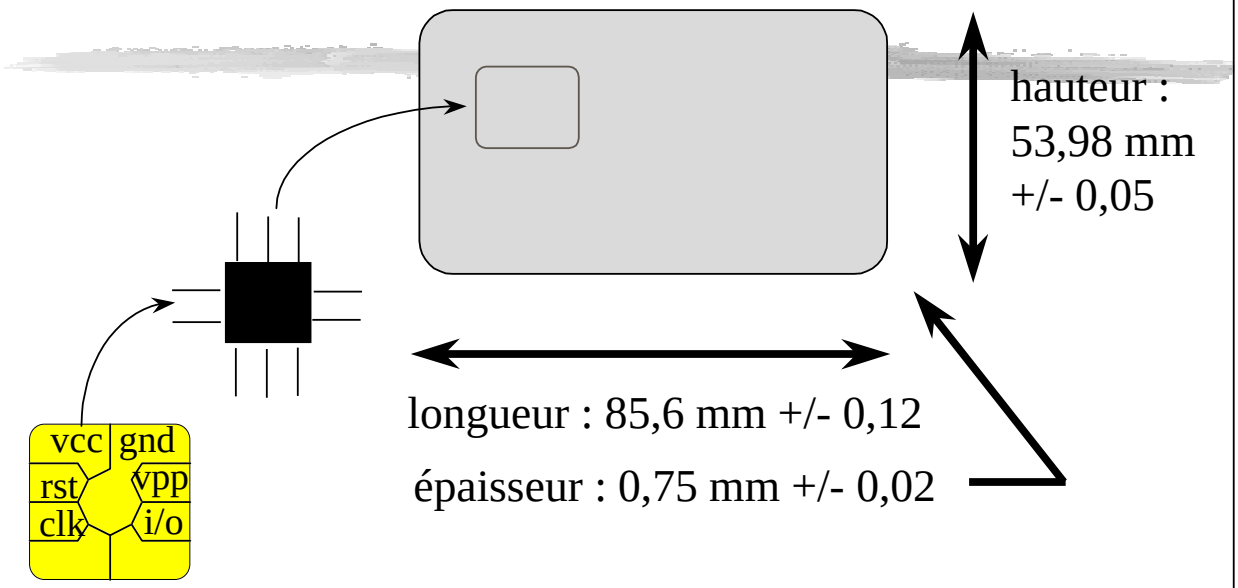
Vers J2EE
L 'Architecture



Les produits Java

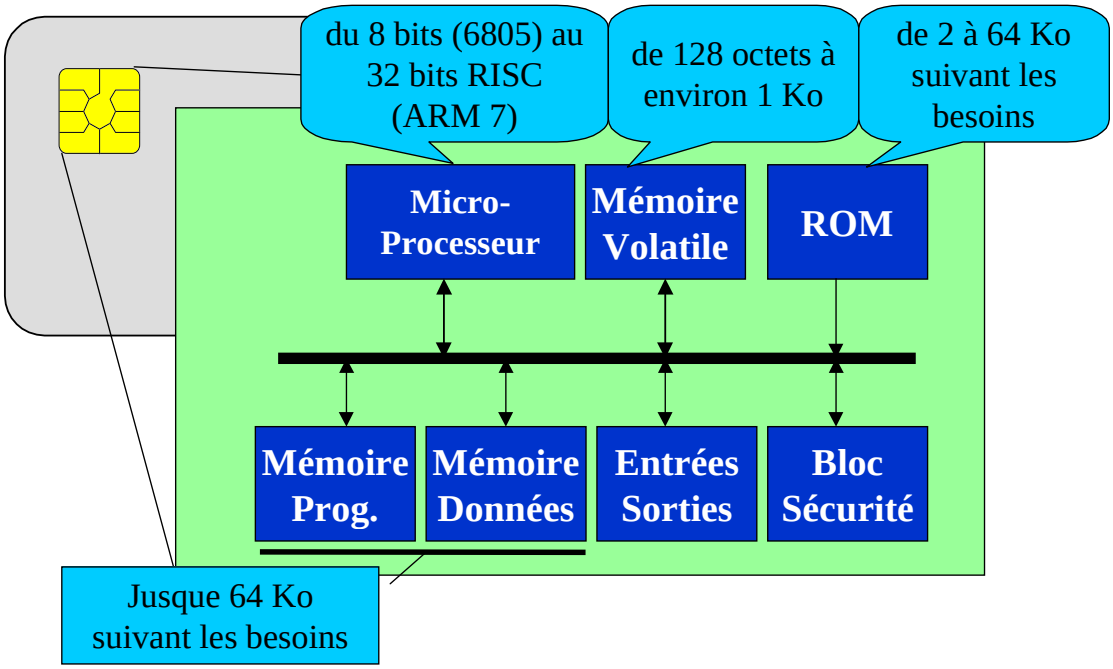
La JavaCard

Caractéristiques de la carte



- Norme ISO 7810 : Caractéristiques physiques (1ère partie)
- Norme ISO 7816-1 : Caractéristiques physiques (2ème partie)
- Norme ISO 7816-2 : Caractéristiques électriques

La carte à microprocesseur



JavaCard : Historique

- Carte '96 (CNIT-Paris)
  - Schlumberger présente la CyberFlex 1.0
  - Parallèlement d 'autres projets commun sur base de :
    - Langage C (Multos)
    - Langage Forth (Projet Gemplus)
  - Accord entre les principaux participants et création du javaCard Forum
- 1998 : le vrai départ
  - CyberFlex 2.0, GemXpresso, ...

Le JavaCard Forum

- Consortium de fabriquant :
  - Carte : Delarue, Gemplus, Oberthur, Schlumberger
  - informatique : IBM, SUN
  - Matériel : DEC, Motorola
  - Utilisateurs : banques
- But :
  - promouvoir la solution de la Javacard
  - Faire des choix communs (définition de standards)
- Solutions :
  - Un comité technique
  - Un comité « business »
  - plus d 'information : http://www.javacardforum.org

Une nouvelle génération : Les cartes multi-services

- But :
  - Permettre à plusieurs applications de coexister
  - Partager des données entre plusieurs applications (non redondance d 'information)
  - Possibilité de charger/décharger des applications en cours de vie de la carte
- Avantages
  - une seule carte pour l 'utilisateur
  - Possibilité d 'évolution des cartes
- Inconvénients
  - Peur des partages
  - Tailles réduite de la place / application

Besoins des développeurs

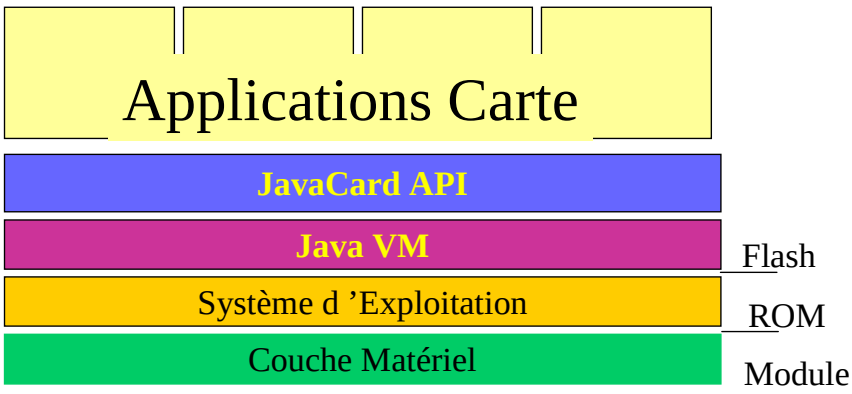
- Sortir la carte d 'une programmation «élitiste » :
  - Seuls les programmeurs de longue date savent correctement écrire une application carte
  - Solution : utiliser des langages de programmation courants en informatique (C, Java, Visual Basic)
- Permettre de tester des solutions :
  - portage sur carte plus rapide => possibilité de test

Besoins des utilisateurs

- Se simplifier la vie :
  - utiliser la même carte pour toutes les applications
    - Plusieurs applications sur une même carte
  - Pouvoir changer de prestataire sans avoir à recommander une carte
    - Chargement / déchargement d 'application

La JavaCard

- Carte basée sur un interpréteur de bytecode Java



- Nécessité de pré-compiler les applications :

Coté Terminal Carte à Puce

- OCF : OpenCard Framework
  - initiative IBM, maintenant www.opencard.org
- ensemble de classes et d 'interfaces Java pour écrire des terminaux accédant à des cartes à puce
- Notion de pilote pour chaque modèle de lecteur
- Cibles : PC, NC, GSM, STB et Décodeur TV
- Concurrent : PC/SC (MicroSoft ;-)

Les produits Java

Les éditions Java

Les éditions Java

- Objectif
  - définir la liste des packages disponibles dans une distribution Java2 en fonction de la plate-forme cible
- J2ME : *Java 2 Micro Edition*
  - système embarqué (embedded) ou nomade (mobile)
- J2SE : *Java 2 Standard Edition*
  - station cliente
- J2EE : *Java 2 Enterprise Edition*
  - serveur Web et EJB

Les produits Java

Les Outils de Développement

Objectifs

- Documentation
- Deboggage
- Précompilateur
- Retro-Compilation
- Ofuscateur
- Analyseur de Performance
- Test de Compatibilité
- Analyseurs Lexical et Grammatical

Documentation

- Javadoc (JDK)
  - Génération automatique de la documentation HTML
  - à partir des commentaires présents dans les `.java`
- Commentaires et Tags

```
/**
 * This is a <b>doc</b> comment.
 * @see java.lang.Object
 */
```

- Documentation Standard (HTML avec/sans frame)
  - hiérarchie des classes et des interfaces, liste des packages
  - résumé et détail d 'une classe, interface, méthode, propriété,...
- Documentation Customisée (RTF, XML, MIF, HTML, ...)
  - Doclet : classe Java chargée par Javadoc pour customiser le résultat de la génération
- Remarque : N 'oubliez pas d'ajouter la génération au Makefile

Deboggage

- Déboggage symbolique
  - option de javac : -g, -g:source,vars,lines
  - deboggage en ligne de commande dbx : jdb (JDK)
  - deboggage graphique (lié à des AGL)
- Trace
  - option de TRACE du programme
  - peut ralentir le .class avec les tests TRACE/¬TRACE
    - solution : utiliser un précompilateur

Précompilateur

- Insertion de directive de précompilation
  - comme dans cpp (phase 1 de CC) : #include, #define, #ifdef, ...
- Intérêt
  - 1 source .java -> plusieurs .java alternatifs
    - exemple d 'application: .java avec trace et .java sans trace
      - évite d 'avoir un .class ralenti par les tests de trace
- Outils : Mocha Source Obfuscator, ...
- Exemple

```
private void myfunction(int x, int y){
//#ifdef TRACE
System.out.println("hello world, I am in myfunction " + x + " " +y);
//#endif
```

  - donne (si TRACE n'est pas défini)

```
private void myfunction(int x, int y){
//#ifdef TRACE
/// System.out.println("hello world, I am in myfunction " + x + " " +y);
//#endif
```

Rétro-Compilation

- Décompilation du bytecode
  d 'un `.class` Java en un source `.java`
  - Partielle/Totale
- Objectif de l ' « attaquant »
  - le nom d 'un bean, d 'une classe, des méthodes, les commentaires des traces, les informations de deboggage… ont une signification
  - retrouver les algorithmes d 'un composant métier, modifier le source (par exemple pour supprimer le code de vérification de la licence) , le détourner, le pirater ...
- Risque pour le développeur
  - Perte des droits d 'auteur, Manque à gagner, ...
- Outils : JDK javap, WingDis, NMI 's Java Code Viewer ...

La parade: l ' Ofuscateur

- But : Eviter l'interprétation d 'un bytecode
  par la rétro-compilation
  - Solution : rendre inintelligible le source avant distribution
- Outils
  - Mocha Source Obfuscator, Jmangle ...
- Méthodes appliquées par ces outils
  - brouillage du nom des classes, des méthodes, des propriétés, et les variables par renommage (a0001, ...)
  - mélanger les propriétés d 'accès (public, private, ...)
  - supprimer l 'information de deboggage
- Remarque : supprimer vos traces
- Attention : ne facilite pas la maintenance si vous manglez les exceptions
  « Erreur a238 : envoyez ce message à notre service de maintenance debug@mycomp.com »

Installateur

- Installation d 'une application
  - copie des classes ou d 'archives JAR
  - vérifie la présence d 'une JVM (JRE)
    - de sa version
    - de ses extensions standards (Swing,JCE,XML,JMF, …)
  - élabore la procédure de désinstallation en fonction de l 'installation
- Outils
  - Win32 : Installer for Java, InstallShield …
  - Unix : ...

Optimisation des Performances

- Interpréteur de Bytecode
- Compilateur Natif (statique)
  - .class en .c en .s en .exe
- Compilateur à la volée (dynamique)
  - Compilation JIT (Just-In-Time) de Symantec
- Optimiseur HotSpot™
  - analyse de la taille des tableaux et vecteurs
  - garbage collector
  - « method inlining »
    - avec vérification au chargement (dynamique) d 'une classe

Test de Compatibilité JavaCheck

- Outil de test de compatibilité
  - d 'une application ou d 'une applet
- avec une plateforme Java
  - PersonalJava version x.y, ...
- La description de la plateforme et de ses périphériques est dans un fichier .spc

Analyseurs Lexical et Grammatical

- Permet de produire un « parser » en java à partir d 'une grammaire et d 'actions en java
- Outils
  - Analyseurs Grammaticaux (LALR)
    - BYACC/Java
      - yacc de Berkeley avec des actions C/C++ ou Java
    - Jacc, JavaCup, ANTLR, QJJ, ...
  - Analyseurs Lexicaux
    - JavaLex