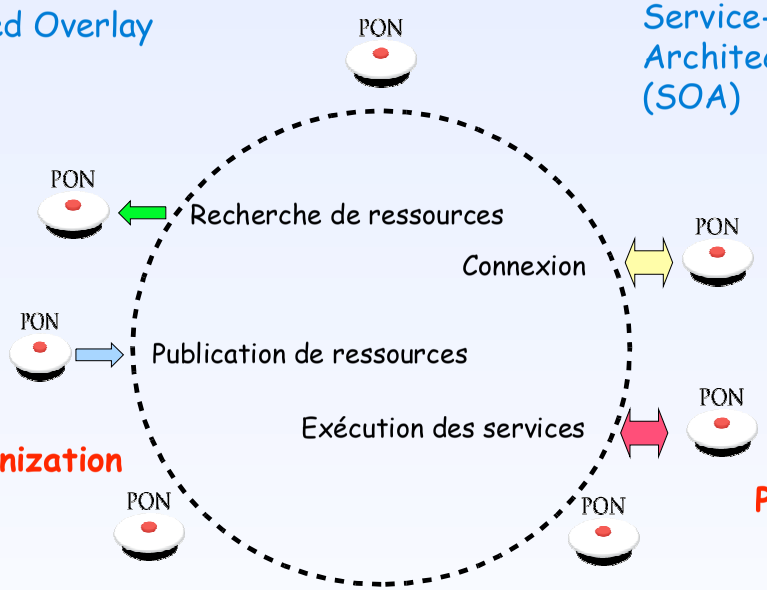
Programmable Overlay Network

Didier Parigot et Baptiste Boussemart

L'idée très simplement

Structured Overlay Network (SON)

Service-Oriented Architecture (SOA)



Self organization

Peer to Peer

<-protocol->

Histoire : Programmable Overlay Network

- Issu du modèle d'exécution de SmartTools (composants)
  - Version répartie en pair à pair de la Service-Oriented Architecture (SOA) de SmartTools
  - Décentralisé grâce à un réseau logique (Overlay Network)
- Les différences par rapport à SmartTools
  - Uniquement la partie exécution
  - Un petit logiciel (300k)
  - Prototype de recherche pour valider notre démarche

- Les concepts de SmartTools non utilisés
  - Fabrique logicielle avec phase de génération
  - Domain-specif Language, GUI client lourd

But : Architecture et Programmation Diffuses

- Objectifs
  - Dynamique, Auto organisée etc…
  - Simple
- Difficultés
  - Nouveau modèle de calcul : approche théorique.
  - Les technologies existantes et émergentes
    - SOA, IDM, Composants, Multi-Agent, WEB 2.0 etc…
- Overlay : En superposition aux techniques classiques.

The Future of The Internet

Interoperability
Scalability
Security

Composition
Planification
Orchestration
Mobility

Services

Overlay Application
SOA

Discovery
Invocation

Components

Resources

Extensible
Decentralization
Self organization

Overlay Network
Pub/Sub

Topology
Fault tolerance
Load Balancing
Reputation

Protocols

Underlay Network
Virtual Pipes

Internet of Things

Wireless
Ad-hoc Network
Mobile

Ma vision de l'ordinateur du futur
Overlay Computer
Superposition aux OS existants

Pile d'exécution
Code exécutable

Mémoire,
Editeur de lien

Bus + Device

L'existant et les Objectifs pour PON

- L'existant
  - SOA de SmartTools proche des Standards (SCA Service Component Architecture)
    - Pas d'orchestration à la Web-Services
  - Au dessus d'OSGi : SOA local à une JVM
    - intégration de SmartTools dans Eclipse
- Nos deux premiers objectifs à court terme pour PON sont :
  - Version répartie en pair à pair
    - Première expérience avec R-OSGi (remote OSGi)
  - Découverte/Recherche des composants en mode décentralisé
    - Par exemple sur des Structured Overlay Network (SON)

Utilisation d'un Structured Overlay Network

- Recherche/Publication des composants
  - Le monde des Structured Overlay Network (SON) parle en terme de ressources (fichiers, video...)
  - **Une ressource = un ensemble de services = un composant**
  - Publication des composants sur un SON
  - Recherche des composants à l'aide d'un SON
- Un SON : annuaire intelligent
  - Décentralisé en P2P, Divers techniques de topologie (Pastry, Chord...)
  - Réputation, Tolérance aux fautes, Groupe de communication (MultiCast)
  - Charge, Répartition etc...

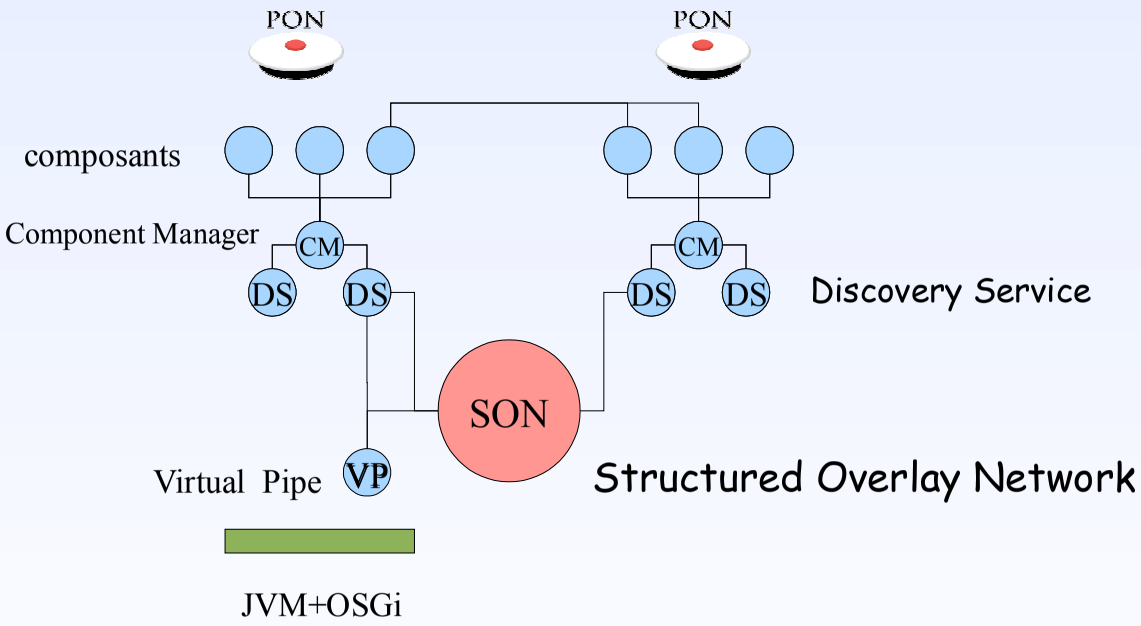
Communication pair à pair par service

- Modèle de communication en local entre composant
  - Composant = ensemble de services
  - Nom des services = tuyau de communication
  - Message asynchrone.
- Version répartie
  - Protocole de transport (TCP,UDP)
  - RMI, MOM, JMX, JXTA…
  - ESB (Entreprise Service Bus)….
- Même niveau d'abstraction que notre SOA (nom des services) et s'abstraire des protocoles (de transport) et des formats
- Solution (issu des concepts de JXTA)
  - Tuyau virtuel par composants ou services

Publication des composants sur un SON

- SOA en local : Gestionnaire de composants (CM)
  - Opération de connexion entre composants
  - Message asynchrone
- SOA + SON = publication et recherche des composants
- SON : `clé` → `valeur; put(clé,valeur); valeur` ← `get(clé).`
  - Nom du composant → tuyau virtuel
  - Type du composant → description des services
  - Autres informations pour la gestion/maintenance
- Gestionnaire de composant lié à des Discovery Services (DS)
  - DS local : version courant pour SmartTools
  - DS PON : en répartie au-dessus d'un SON.

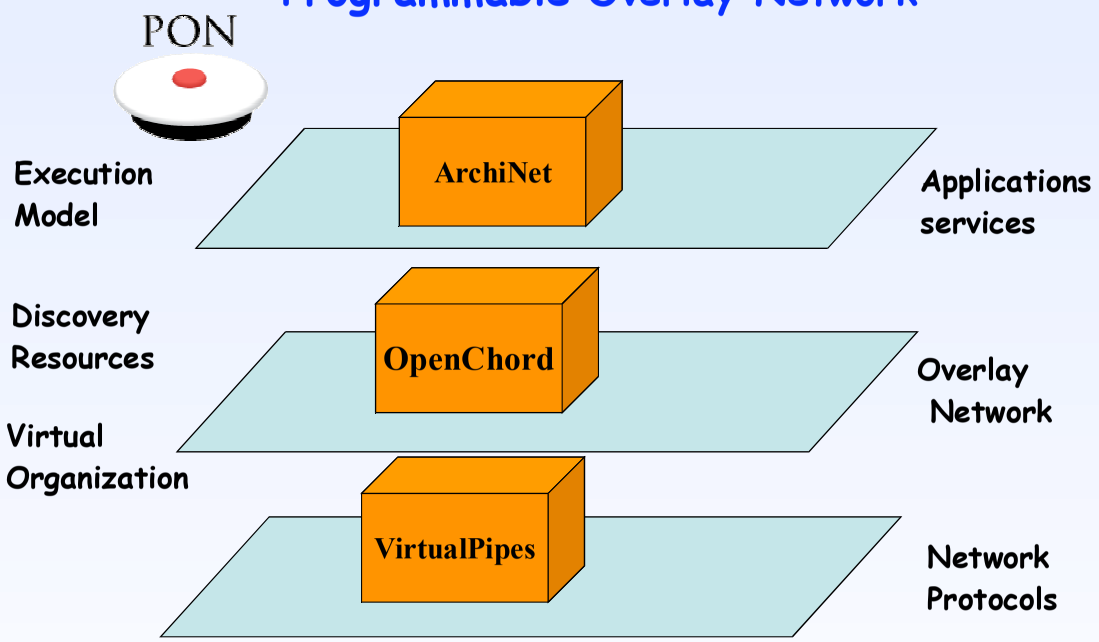
Version répartie : SOA + SON



Tuyau de communication = services

- Expérience avec R-OSGI :
  - Trop compliqué pour notre SOA
- Communication: Discovery Service (DS), première expérience point à point.
  - DS UDP, DS TCP.
  - Proxy fixe pour chaque type du protocole de transport
  - Multiplexage en fonction du type du protocole de transport
- Tuyau de communication: Nom unique UUID d'un tuyau
  - Pour une connexion TCP faire passer plusieurs tuyaux de communication
  - Publication des tuyaux sur le SON.
  - Utilisation de l'API NIO pour lecture/écriture non-bloquantes et réduit le nombre de thread (un seul thread par JVM/PON)

Programmable Overlay Network



Tuyau de communication : unicast

- Pour chaque composant un tuyau d'entrée (lecture)
  - Publié dans le SON : nom du composant → tuyau
- Connexion `connect (A,B)`, B à distance
  - Le DS de A recherche dans le SON le tuyau d'entrée (lecture) de B.
  - Le DS de A crée un proxy associé au tuyau d'entrée de B.
  - Le DS associe les sorties (écriture) de A à ce proxy B.
  - L'opération `connect` s'effectue aussi dans l'autre sens.
- Autres informations publiées sur le SON
  - Tuyau de service d'un DS pour effectuer les connexions dans le sens inverse.
- Communication en Unicast
  - Un composant connecté à trois composants alors trois envois du même message.

Tuyau de communication : Multicast

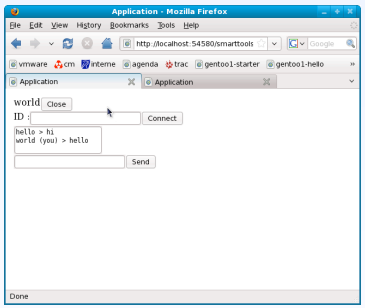
- Communication en MultiCast, écriture d'un seul message pour l'envoi d'un message vers une même JVM.
- Pour chaque service en sortie (écriture) d'un composant, publication d'un tuyau de sortie sur le SON.
- `connect (A,B)` et `connect (A,B1)`, B et B1 sur la même JVM
  - Le DS de B et B1 recherche sur le SON le tuyau de sortie (écriture) de A pour l'associé au service d'entrée (sortie) de B et B1
  - Le DS associe ce tuyau au service d'entrée (lecture) de B et B1
- Une seul écriture du message pour deux lectures

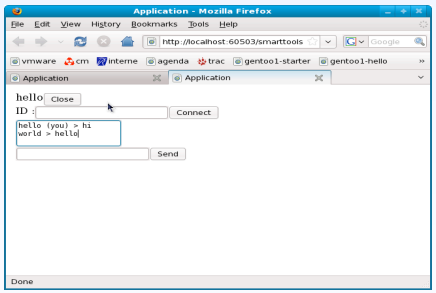
Distribution de PON (en construction)

- Distribution sous http://gforge.inria.fr/projects/smarttools/
  - PON (Programmable Overlay Net)
    - Les Bundles/Plugins de base pour PON.
    - Version Plugins ou Bundles
  - PON des petits exemples en version Standalone
    - Chat
    - Transport
- Le site de PON http://www-sop.inria.fr/lognet/pon/
  - Quelles informations sur l'installation (source plugins/eclipse)
- Sur notre serveur http://mirage/lognet/wiki/LogNet
  - Documentations purement techniques
- Développement en java: Linux,Windows, Nokia 800

Premier exemple naïf pour PON

- Exemple de « Chat » instantané (Hello Word pour PON)
- Les utilisateurs s'inscrivent (`put`) sur un SON et demande à être en communication avec quelqu'un (`get, connect`).
- Démonstration à télécharger http://gforge.inria.fr/frs/download.php/19603/chat-0.1.zip



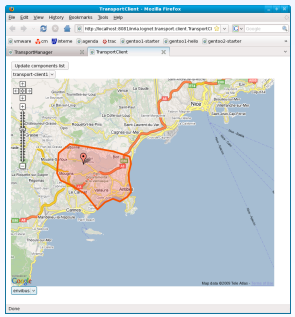


Interface en GWT pour les composants

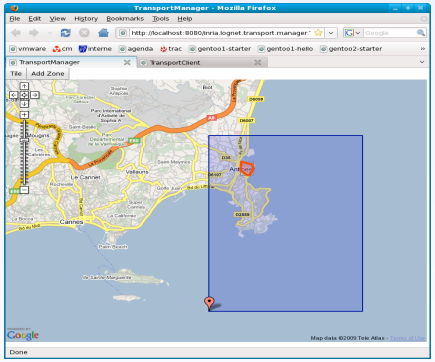
- Tous nos composants peuvent être munie d'une interface WEB
  - Accès à distance à nos composants (bundles sur un Nokia 800)
- Plate-forme OSGi bien instrumentée pour cela
  - Serveur Jetty etc…
- GWT: génération du Javascript à partir d'un code java
  - Programmer une page WEB comme en java
- GWT: communication entre le client et le serveur en RCP (google)

Exemple de localisation de Transports avec l'API Google Map

- Le prestataire de service publie pour une zone géographique leurs services dans un SON : `put()`
- l'utilisateur recherche en fonction de sa position les prestataires de transport dans la zone : `get()` et se connecte au service



Utilisateur



Prestataire

Exemple: Déploiement des PONs sur Cluster

- En cours de construction,
- Utilisation des possibilités
  - d'OSGi (installation à distance, lancement d'un bundle)
  - de PON (une SOA sur un SON)
  - Connexion entre les PONs avec le SON
- Lancement d'un ou des managers de déploiement reliés à un SON
- Un démon sur chaque machine qui lance une installation de PON minimal (client PON) et recherche le ou les managers disponibles (connexion à la PON)
- Le client PON va demander la configuration à lancer au manager
  - Fichier lu par le gestionnaire de composant.

Conclusion: Programmable Overlay Network

- PON est
  - un prototype de recherche pour valider notre démarche
  - un petit logiciel
  - très facile d'utilisation/installation

- Futur
  - Mieux utiliser les possibilités des SONs,
  - Instrumenter notre SOA par une coordination (work-flow),
  - Ouverture vers les protocoles de l'Internet des objets,
  - Déploiement d'un réseaux de PONs (sur un Cluster).

Conclusion: Questions

http://www-sop.inria.fr/lognet/pon/