Architecture Distribuée

Cours n°4

A.Saval

Objectifs du cours “Architectures distribuées”

- Compréhension des motivations
- Compréhension de la logique de conception d'une architecture distribuée
- Maîtrise des principaux modèles
- Aperçu des problèmes posés
- Aperçu de quelques frameworks existants

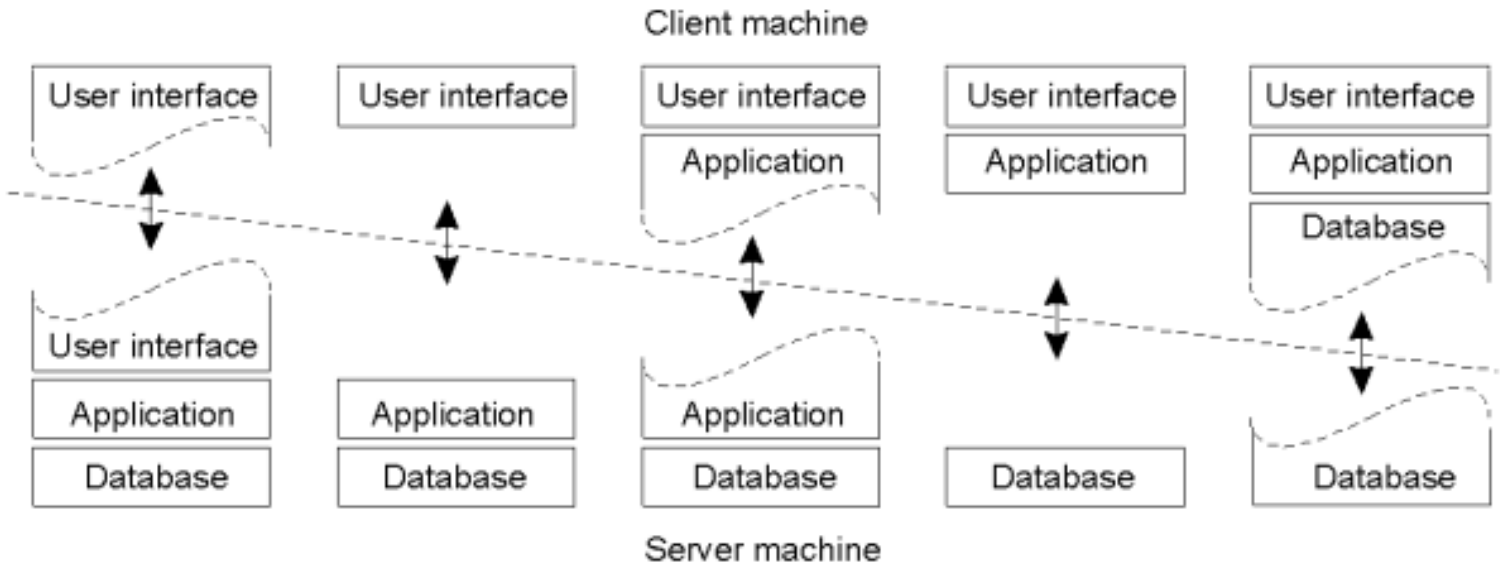
Aperçu du cours

- Introduction
- Problème de conception d'architecture
- Architecture logique & matérielle
- Système distribué
- Modèles d'architecture
  - Client/serveur
  - 3-tiers
  - N-tiers
  - P2P
  - Virtualisation

MODELES D'ARCHITECTURE

Modèle: 3-tiers

Répartition des couches de fontionnalités entre client / serveur



Séparation des fonctionnalités: communication

Séparation des fontionnalités:

- Besoin de définition de ces fonctionnalités
- Définition des moyens de communication

- Formalisation de :
  - La mise à disposition de fonctionnalités
  - L'accord sur des moyens de communication

Definition: Service

Spécification des fonctionnalités sémantiques d'une entité.

Qualité de service:

Caractéristiques non fonctionnelles d'un service (précision, disponibilité ...)

Interface de service:

Entrées et sorties compatibles avec le service.

- Paramètres et résultats
- Règles de séquence d'appel du service

Definition: Protocole

Ensemble de règles qui décrivent la façon dont se déroulent les échanges et les interactions entre les objets.

Ces régles définissent:

- Le format des messages
- Les différents états du protocole et pour chacun les messages reçus et envoyés
- Les contraintes de qualité du service

=> La définition d'un protocole n'implique pas la défintion d'un service

Exemple de service: annuaire de villes

- Le service fourni un ensemble de fonctionnalités de gestion des villes:
  - afficher les villes de l'annuaire
  - ajouter de nouvelles villes, les supprimer

- Interface:
  - l'ensemble de appels qui fournissent ces fonctionnalités.

- Qualité:
  - la vitesse de réponse, la duplication inutile des villes, la gestion de erreurs ...

- Règles:
  - l'enchaînement des appels (ajouter une ville, puis afficher l'ensemble des villes pour s'assurer que l'ajout a bien été effectué), la signification des arguments, les valeurs de retour

MODELES D'ARCHITECTURE

Communications dans les architectures distibuées

Communication

- En mémoire/persistant:
  - Ephémère (*Transient*): nécessite que le destinataire reçoive l'information au moment où elle est envoyée.
  - Persistent: quelque que soit l'état de l'expéditeur ou du destinataire.

- Synchronisation:
  - Asynchrone: celui qui appelle n'attend pas de réponse
  - Synchone: l'appelant est bloqué tant que sa demande n'a pas été acceptée
  - Par tampon (buffered): tolérance limitée entre les appels et les réponses

Communication

- Eléments échangés:
  - Discret: ensembles structurés d'information, constitue un tout
  - Flux (streaming): chaque élémént se suffit à lui même
- Connexion:
  - Mode connecté
  - Mode non connecté

Communication

- Fiabilité:
  - Correction des erreurs (aucune, intégrée au protocole, gérée par le service)
  - Reprise de la communication sur erreur
- Contraintes temporelles:
  - Relation temporelles entre les échanges
  - Synchrones: delais maximum d'attente
  - Isochrone: par intervalles de temps

Middleware

- Basé sur l'IP:
  - Connecter n'importe quel élément.
  - Gérer l'intéropérabilité au niveau UDP, TCP/IP.

- Répond à différents besoins:
  - Faire transiter des informations sur la couche de transport
  - Pas de gestion des standards

- Conséquences:
  - Pas de middleware émergeant
  - Multiplication des middleware par domaine d'application

Middleware

- Deux interfaces:
  - Application Middleware
  - Protocole réseau
- Cas d'utilisation 1:
  - Passage de messages entre applications
  - Dépend du protocole de l'application
- Cas d'utilisation 2:
  - Définition d'un protocole sur la couche middleware
  - Partage basé sur le standard du middleware

Remote Procedure Call

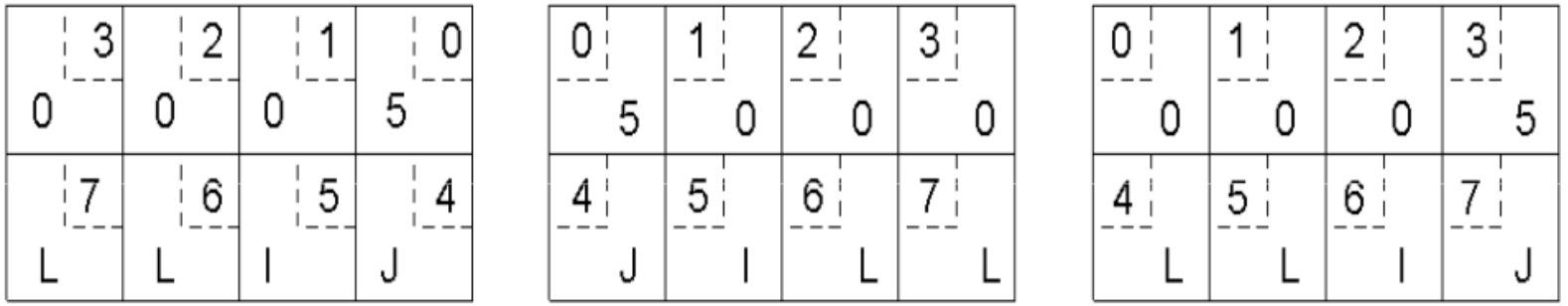
- Type Client/Server:

- Intérêts:
  - Transparence des accès
  - Portabilité
  - Diminue la dépendance au système

- Difficultés:
  - Passage à l'échelle
  - Fiabilité
  - Performance

Remote Procedure Call: Marshalling

Représentations des données d'un système à un autre:



(a) (b) (c)

(a) message sur Pentium
(b) reçu sur une machine SPARC
(c) l'inversion du message reçu ne suffit pas

Remote Procedure Call

- Prévoir les problèmes:
  - Décrire le comportement pour retrouver un état stable
- Possibilité de problèmes
  - Perte de requêtes, répliques
  - Problème serveur/client
- Solutions:
  - Timeout et forcer les demandes de réponses
  - Definir précisément les sémantiques des appels

Definitions: Proxy

- Se comporte de la même façon qu'un composant et implemente la même interface
- Propose des possibilités de filtrages ou de cache

Généralement un *proxy* se trouve du coté client, un *reverse proxy*, coté serveur.

Definitions: Stub

- Historiquement il s'agit d'une implémentation vide d'un interface
- Pour RPC/RMI:
  - Stub client: équivalent d'un proxy
  - Stub serveur: gère les appels du stub client et les échanges de messages

Exemple: un server proxy HTTP

Definitions

Object Wrapper:

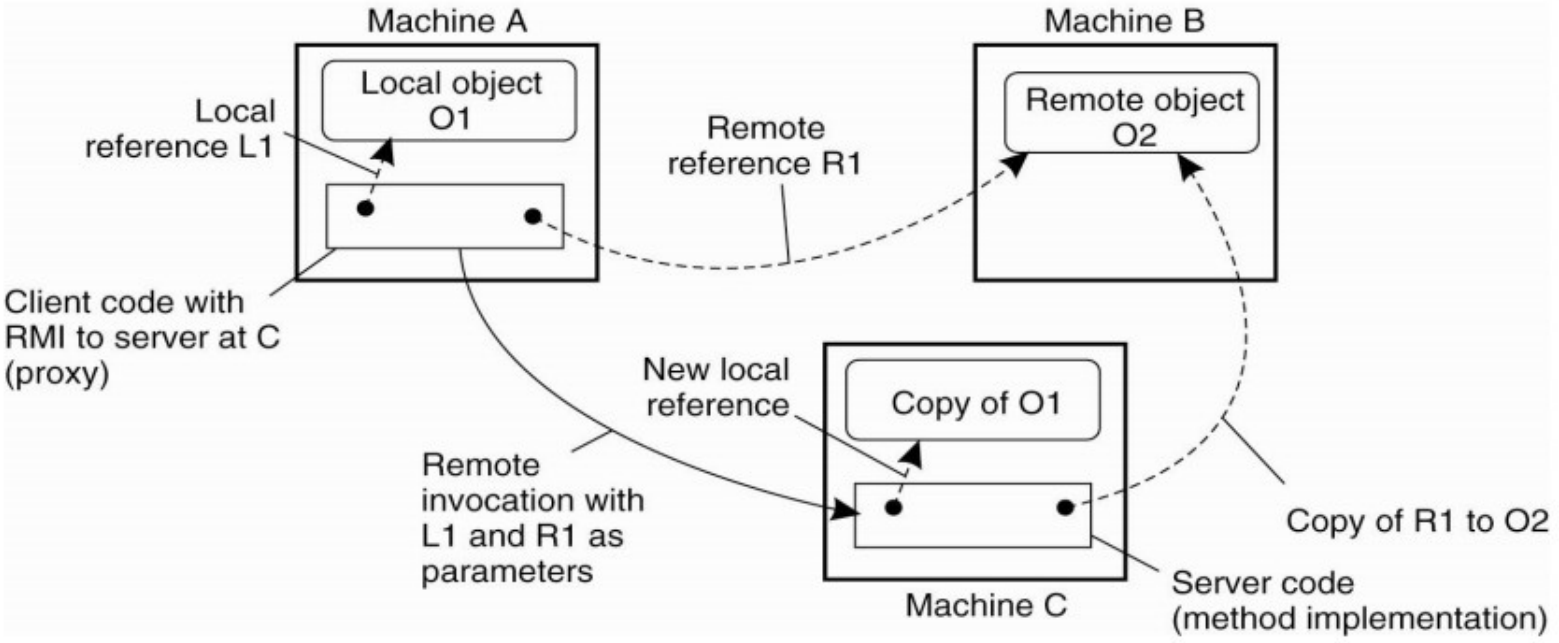
- Un composant qui gère la transition des objets et des appels entre les interfaces
    - Convertir l'objet d'un type en un autre

Broker:

- Un composant qui gère les messages et liens entre les références et les objets.

Remote Method Invocation

- Proche de RPC
- Points de divergence:
  - Un objet représente un contexte (**état**) ainsi que ses opérations
  - Séparation entre interface et l'impémentation
  - La réference aux objets est transparente



Communication par messages

- Comparaison avec RPC/RMI:
  - Synchrone dans le temps
  - Synchrone dans l'espace
  - Synchrone dans les fonctionnalités

- Besoin de communication avec un couplage moins fort

Système de files d'attente

- Caractéristiques:
  - Assurance d'une réponse
  - Persistance
  - Asynchrone
  - Interface simple:

Put	Ajoute un message à la file d'attente
Get	Attend tant que la file d'attente est vide et retire le premier message
Poll	Vérifie s'il y a des messages dans la file d'attente. Non bloquant.
Notify	Ajoute une référence à appeler quand un nouveau message arrive dans une file d'attente

Communication par flux

- Contraintes temporelles fortes entre les données
- Gestion du temps réel
- Problème de qualité de service

MODELES D'ARCHITECTURE
modèle n-*tiers*

Problèmes de communication dans les systèmes ?

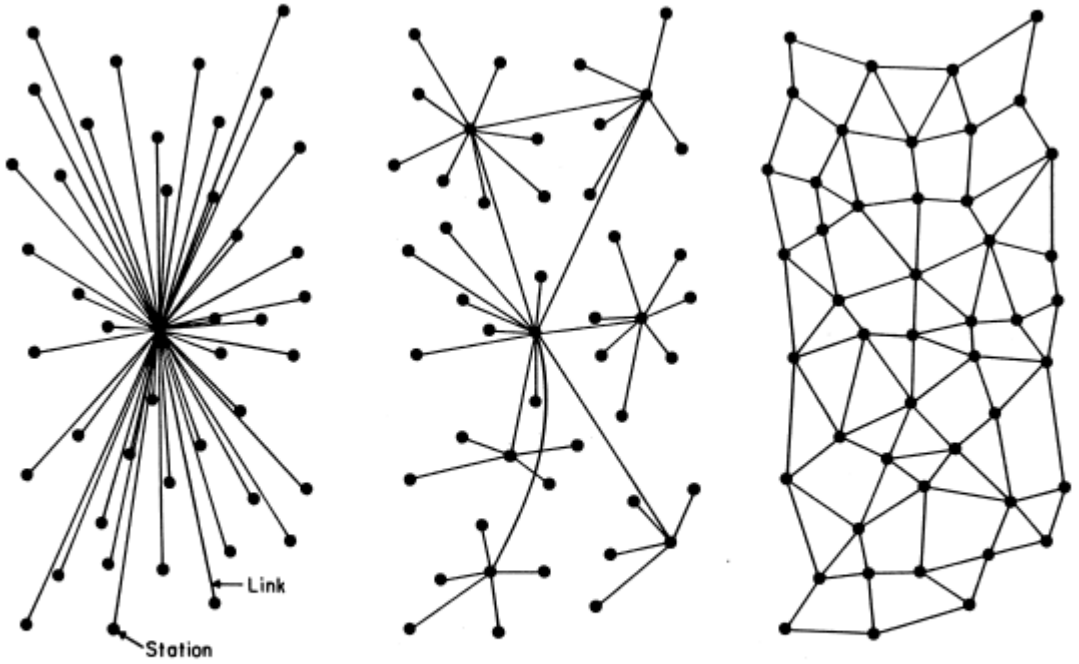


FIG. 1 — Centralized, Decentralized and Distributed Networks

Problèmes de communication dans les systèmes ?

Problème de décision:

- Est ce qu'il existe un ou plusieurs services capables de répondre à ma requête ?

- Quels sont les avantages/inconvénients de chacuns ?

- Quelles sont mes contraintes entre termes de resources ?

Références

- Software Architecture: IEEE Standard 1471-2000
- P. Kruchten, Architectural Blueprints—The “4+1” View Model of Software Architecture, IEEE Software 12 (6), Nov. 1995, pp42-50
- Tanenbaum & van Steen, Distributed Systems, Principles and Paradigms, seconde édition
- Architecture of Distributed Systems, cours de Johan Lukkien, 2011
- Architectural Patterns Revisited – A Pattern Language, Paris Avgeriou & Uwe Zdun, 2005
- Software Architecture, Foundations, Theory, and Practice, R.N. Taylor, N. Medvidovic, E.M. Dashofy, Wiley & Sons, 2009
- Software Architecture in Practice, Second Edition, L. Bass, P. Clements, R. Kazman, SEI Series in Software Engineering, Addison-Wesley, 2003