GT MOB

Architecture Logicielle pour l'Adaptation Dynamique d'Applications sur Périphériques Contraints

Philippe ROOSE

(23 janvier 2009)

LIUPPA

IUT de Bayonne-Pays Basque

Plan

- Applications sensibles au contexte
- Descriptif des activités menées - historique
  - Qualité de Service
  - Méthode de Conception
  - Outils d'adaptation permettant la prise en compte du contexte
- Évolutions vers l'embarqué
  - Cas des capteurs
  - Généralisation à tout type de périphérique « contraint »

Partie 1

Constat

- L'évolution des applications a mis en évidence le besoin d'informations autres *(heure, localisation, préférences utilisateurs, historique interactions, etc.)* que celles uniquement liées aux traitements.

- Elles ne sont pas forcément explicitement identifiées.

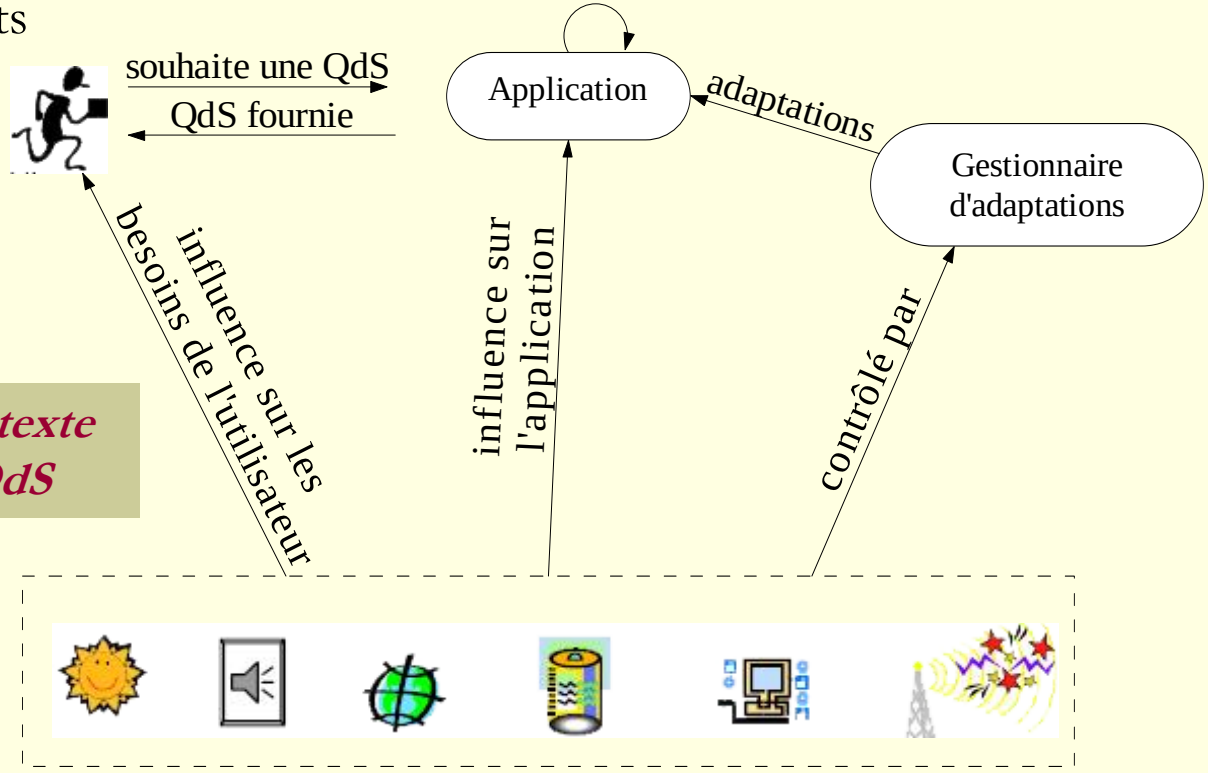
C'est le contexte de l'application [Liebermann, 2000], [David, 2005]

Conséquence

- Concevoir des applications :
  - Sensibles au contexte
  - Adaptant leurs traitements

  en conséquence

Les adaptations ont pour but d'améliorer la QdS

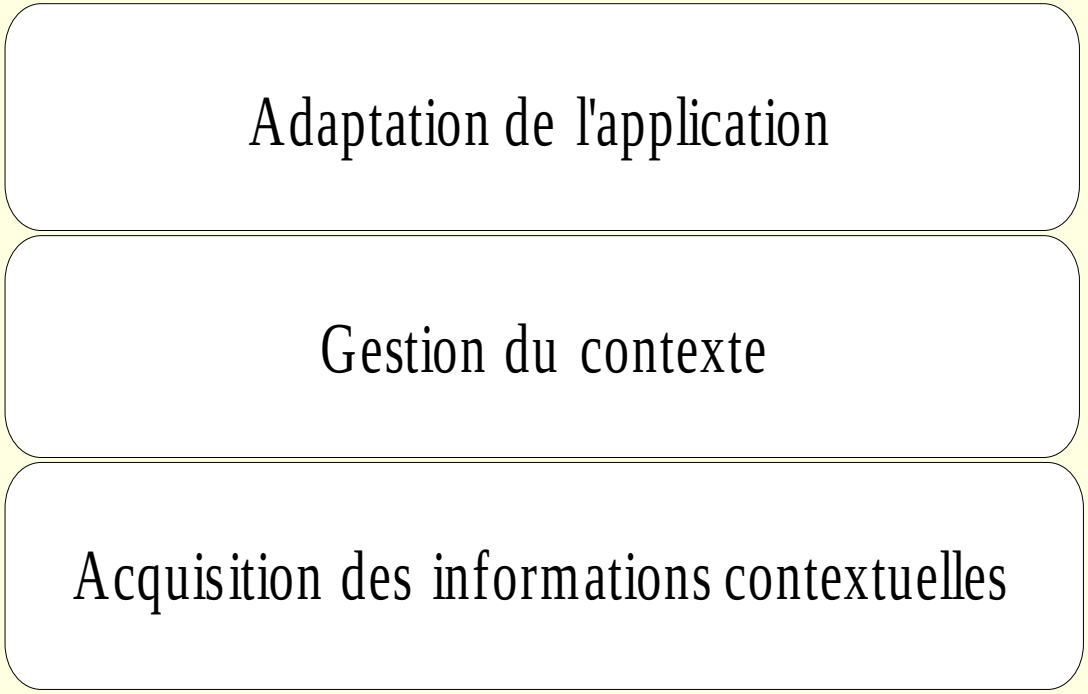


Les modifications du contexte peuvent influer sur la QdS

Répondre au mieux aux sollicitations

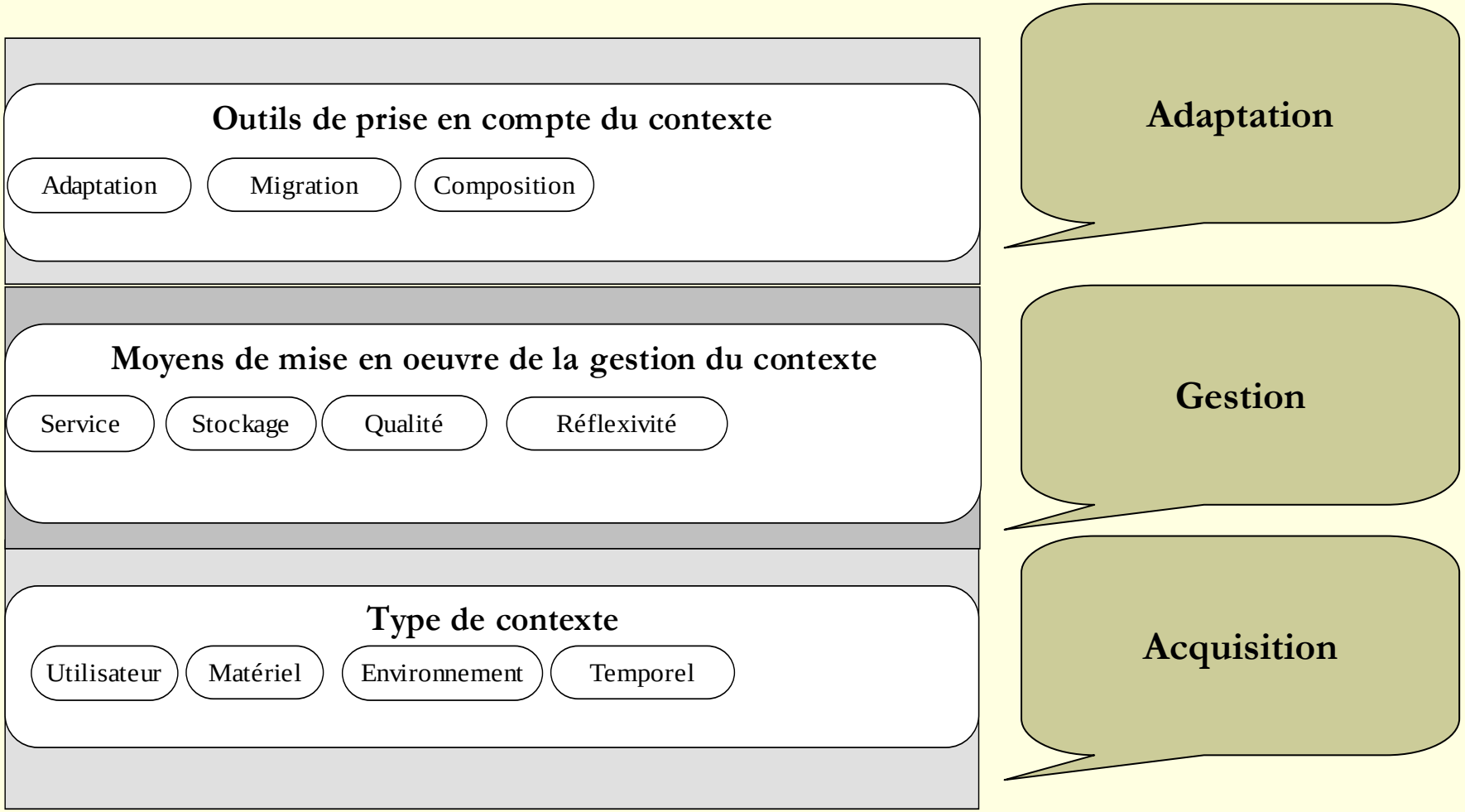
Plan d'action

- Principe architectural des applications sensibles au contexte [Tigli, 2006]



GT MOB

Domaine de recherche : le contexte



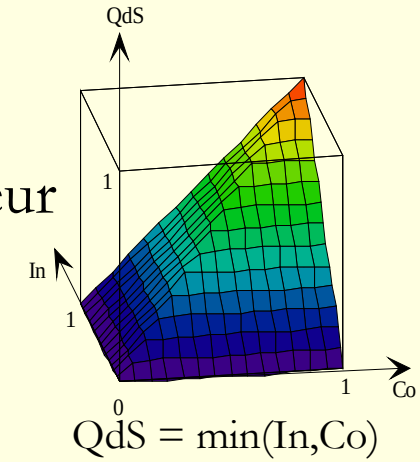
GT MOB

Définition générale de la QdS

- Pas de définition unique
- Standard X.902 de l'UIT
  - « *ensemble d'exigences concernant le comportement collectif d'un ou plusieurs objets* »
- Notion habituellement utilisée dans les réseaux
- Depuis la démocratisation d'Internet, l'évaluation de la QdS n'est plus basée uniquement sur des critères réseaux et matériels

Qualité de service

- Concept d'utilité : « *satisfaction qu'une personne retire de la consommation d'un bien ou d'un service* » [Parkin, 1992].
- Nous nous intéressons aux causes de la satisfaction :
  - Critère intrinsèque : le service ne donne pas satisfaction
    - *résolution, nombre de couleurs, etc.*
  - Critère contextuel : le service ne donne pas satisfaction dans le contexte actuel
    - *img/sec*
- Représentation : courbes d'indifférence
  - **0** : service rendu rédhibitoire/ **1**: attendu pour l'utilisateur



QdS = min(In,Co)

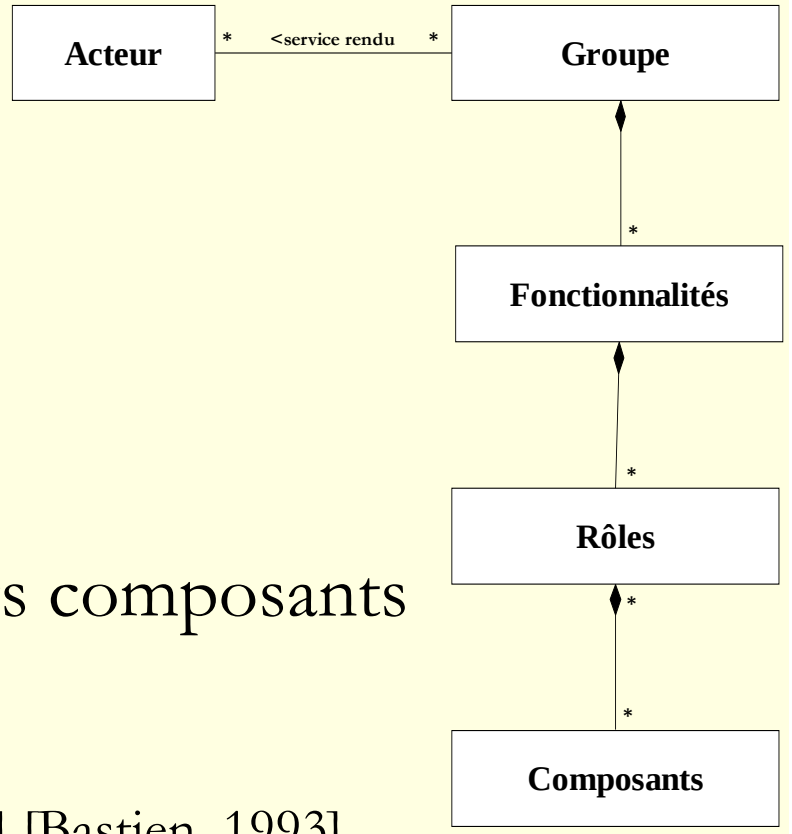
Verrous

- Modélisation pour l'adaptation
- Evaluation de la QdS

Thèse de S. Laplace

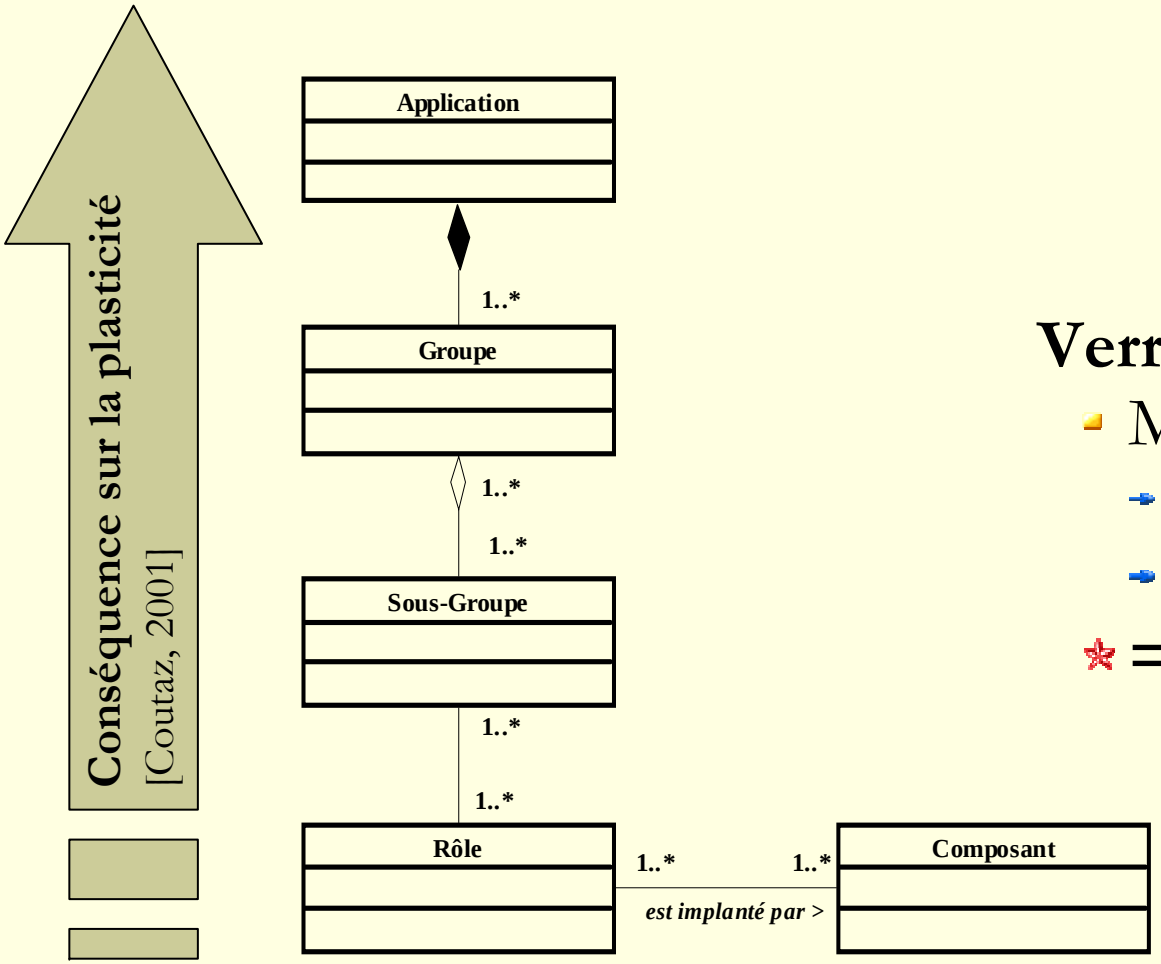
Structuration des applications

- Groupes
  - Service rendu à un acteur
- Sous-groupes
  - Fonctionnalité
- Rôles
  - Peuvent être joués par plusieurs composants
- Composants
  - Logiciels/Matériels [Francke, 1991] [Bastien, 1993]
  - Unitaires ou composites *(vertical/horizontal)*



Structuration des applications

GT MOB



Verrous

- Modélisation d'applications
  - Reconfigurables
  - QdS
- **=> Implantation**

Méthode de conception de l'application

GT MOB

Les graphes représentent l'architecture de l'application

- **Graphe des flots de contrôle**
  - Spécifications fonctionnelles fournies par le concepteur
- **Graphe fonctionnel**
  - Modèle fonctionnel de l'application
  - Détail des assemblages en rôle atomiques
- **Graphe de configuration**
  - Possibilités d'assemblages avec les composants
  - Composition (composants, connecteurs) et l'implantation d'une application

Automatisable

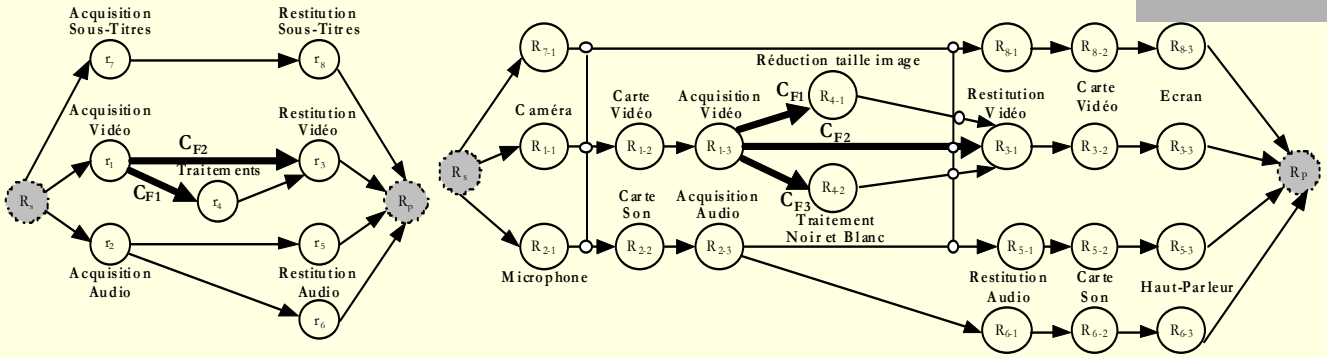
- Taxonomie Composants

Automatique

Thèses de S. Laplace/E. Bouix

Méthode de conception

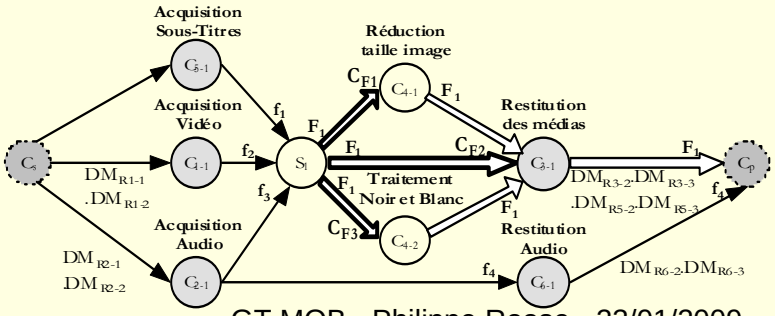
Graphe des flots de contrôle, Graphe fonctionnel, Graphe de configuration



Méthode de Conception

Règles de Tranformation

Graphe de Transition



Manque la dimension réseau

GT MOB

Représentation du déploiement

- **Graphe de transition**
  - Ajout des dépendances matérielles à partir du diagramme de configurations
  - Ajout des opérateurs d'interconnexion (*ex. duplication de flux*)
  - Expression de l'architecture
  - Composants, Connecteurs, Opérateurs

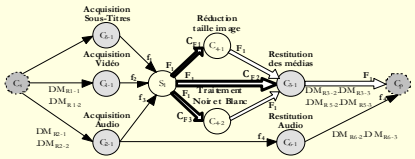
Automatique

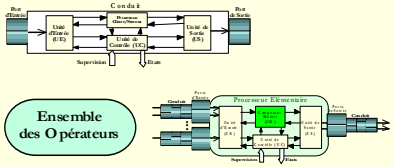
-Informations de déploiement

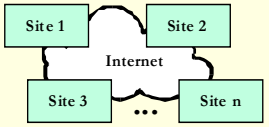
- **Graphe d'implantation**
  - Transformation du graphe de transition
  - Ajoute dimension réseau
  - Fournit une vue directement implantable.

Représentation du déploiement

Graphe de transition, Graphe d'implantation







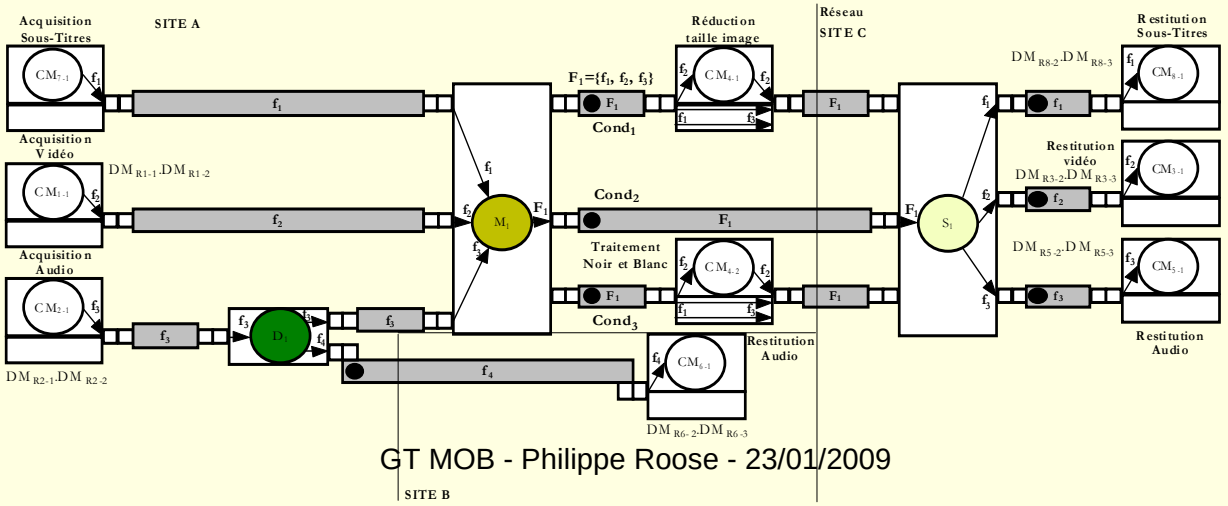
Graphe de Transition

Modèle de Composants Osagaia
Modèle de Connecteurs Korrontea

Informations de Déploiement

Règles de Tranformation

Graphe d'Implantation



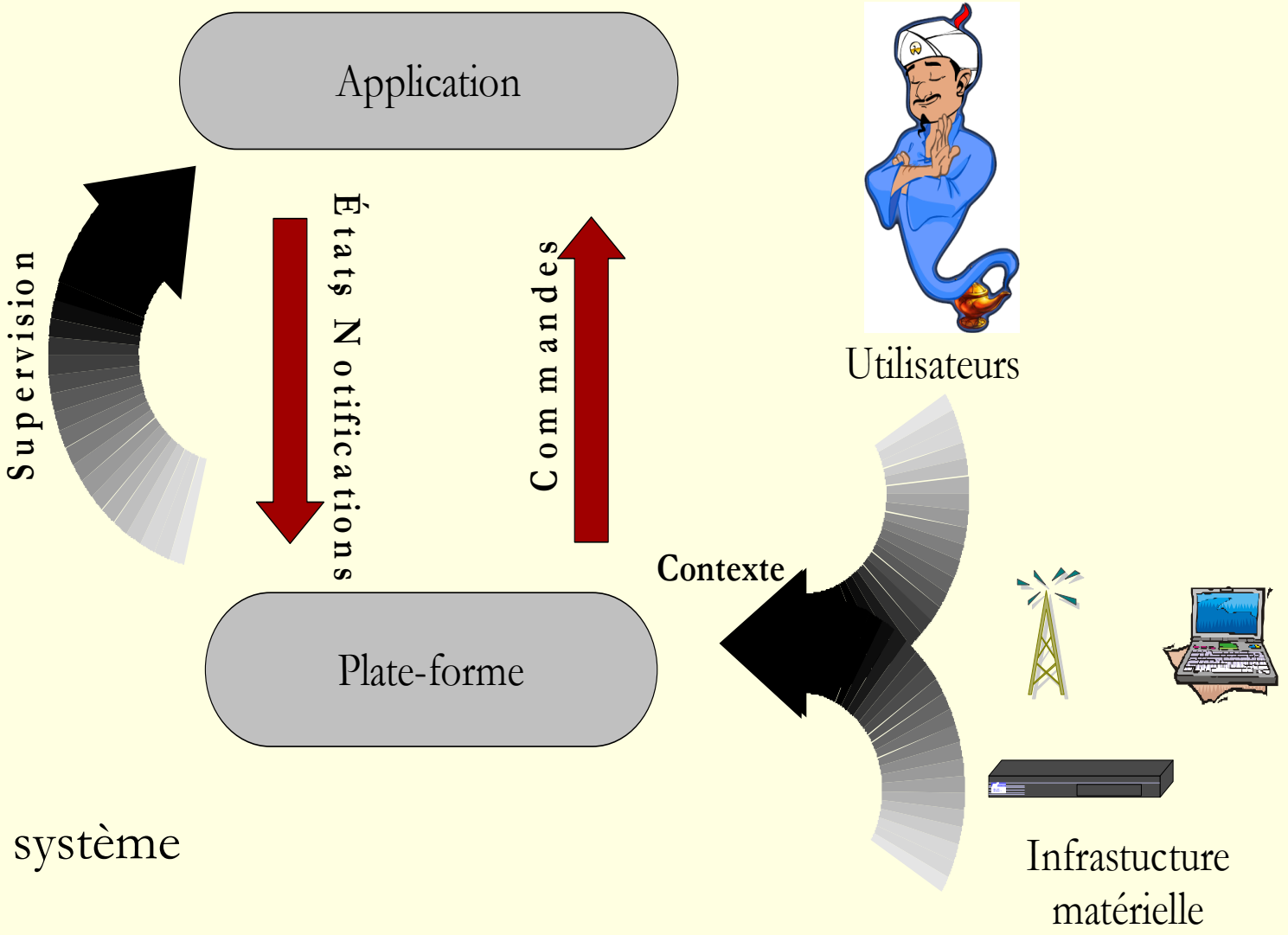
GT MOB

Évaluation de la QdS

GT MOB

- **Graphe d'évaluation**
  - Représente les informations nécessaires à l'évaluation de la QdS.
  - Intègre des vecteurs de QdS
    - résolution image, im/sec, fréquence, débit, etc.
  - Utilisé par l'heuristique de reconfiguration (sur la PF)

Architecture Globale



Verrou

- Réflexivité du système

Outils d'adaptation permettant la prise en compte du contexte

Fusée à trois étages

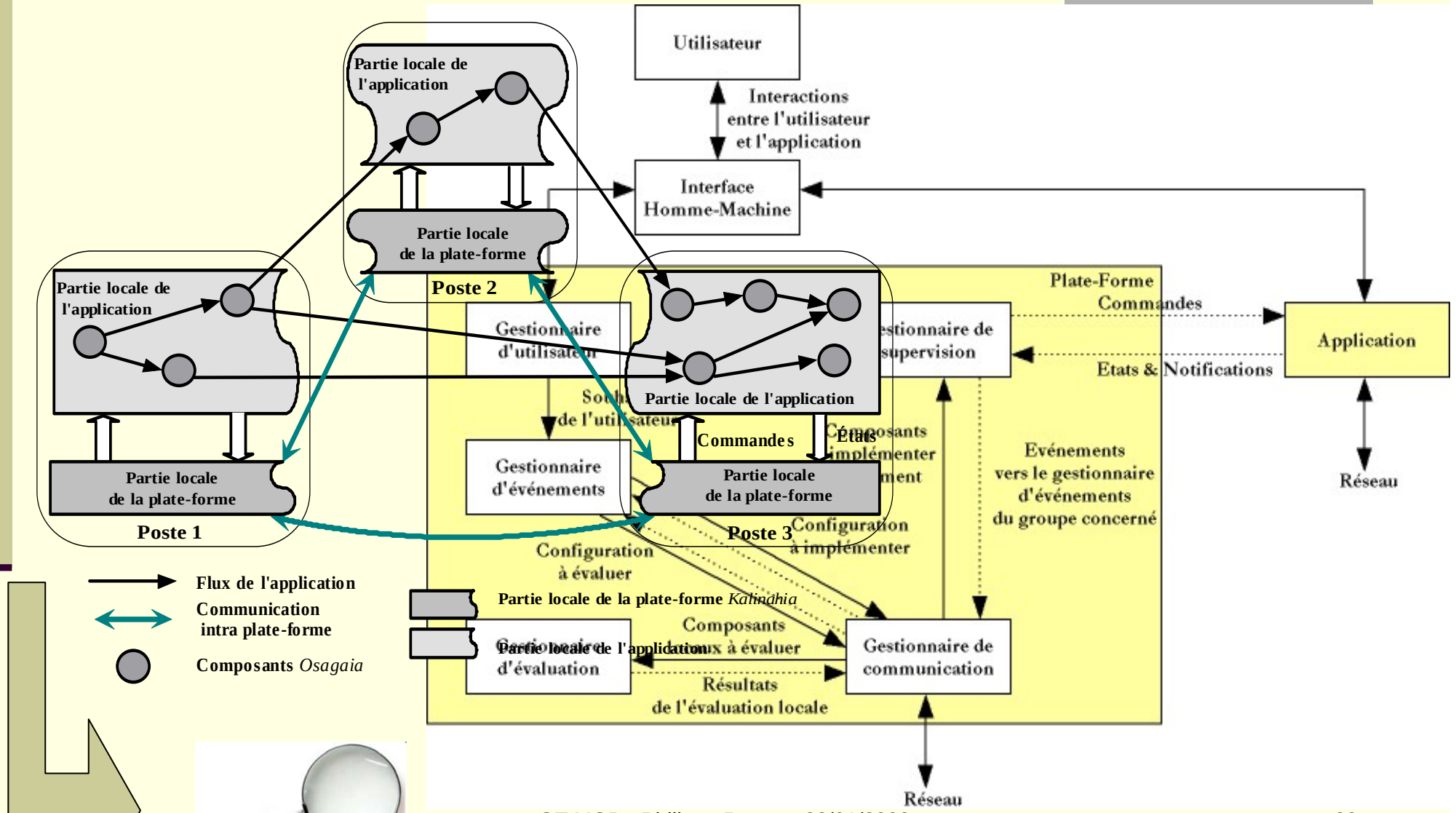
- **Kalinahia** : plateforme réflexive, implémente des heuristiques de reconfiguration, assure les (re-)déploiements/(re-) configurations des composants.
- **Osagaia** : modèle de composants supervisables permettant la réalisation d'assemblages dynamiques.
- **Korrontea** : modèle de connecteurs de première classe, supervisables, et capable de renseigner sur l'état des communications entre composants.

Thèses de S. Laplace/E. Bouix

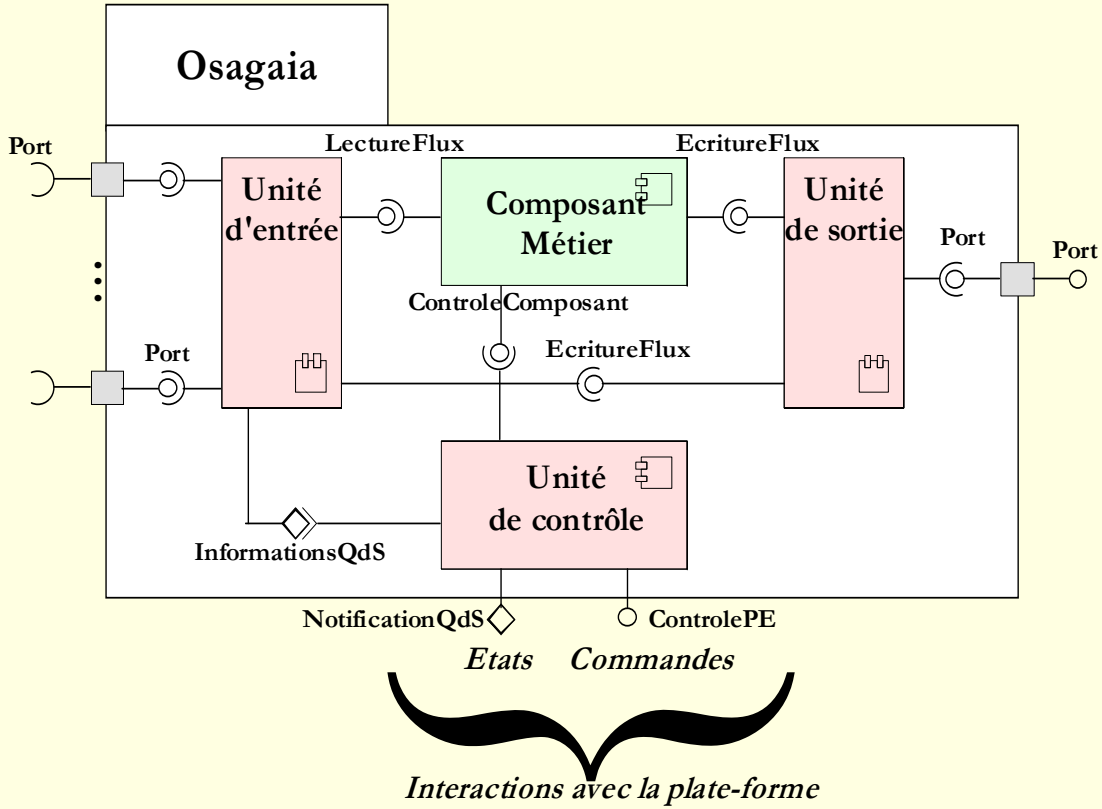
KALINAHIA : Plateforme d'exécution

- **Objectif** : Offrir une qualité de service acceptable aux utilisateurs
  - Meilleure qualité = NP-Complet
- **Moyens**
  - Modifier le comportement des composants logiciels (lorsqu'ils proposent différents niveaux de QdS) ;
  - Modifier la composition de l'application (par ajout, retrait, remplacement ou migration) des composants et par conséquent des flux de données.
- **Verrous**
  - Capture du contexte
  - Heuristique de choix de déploiement
  - (Re-)déploiement dynamique des composants

Modèle informel de KALINAHIA



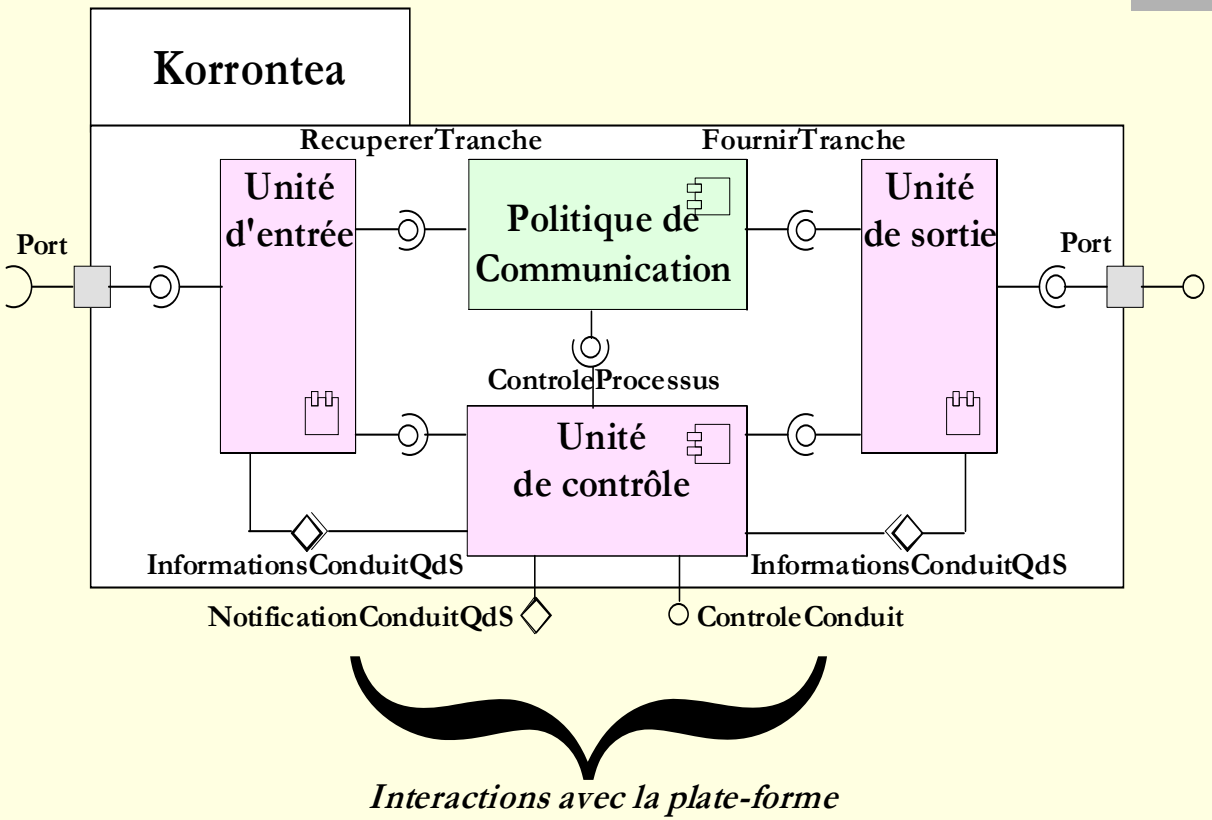
Modèle conceptuel d'OSAGAIA



Verrous

- Connexions/Déconnexions
- Supervisables
- Séparation des préoccupations Conteneur/CM

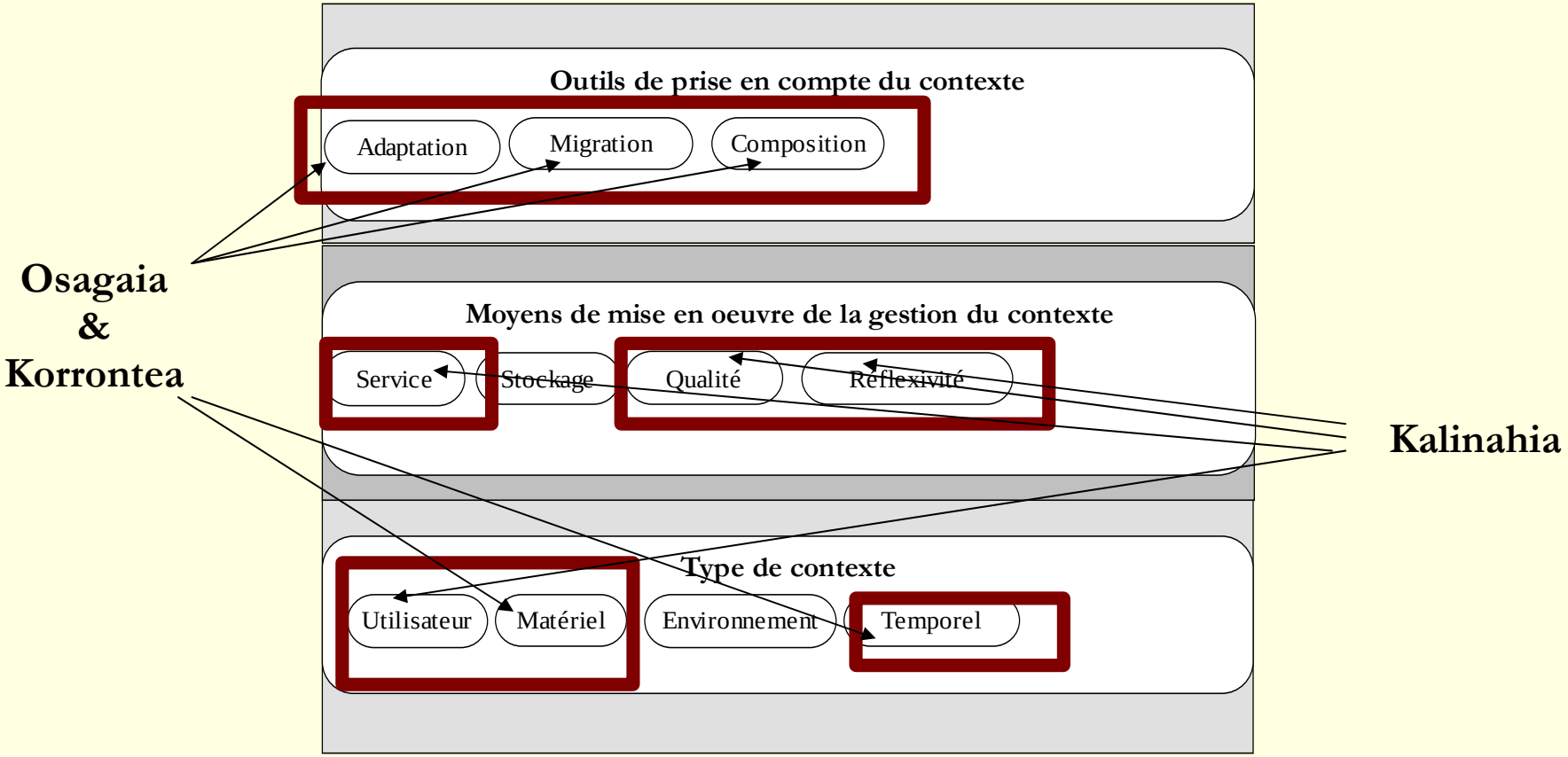
Modèle conceptuel de KORRONTEA



Verrous

- Respect des contraintes temporelles
  - *Politiques de synchronisation*
- Évolutivité des contraintes

Outils d'adaptation permettant la prise en compte du contexte



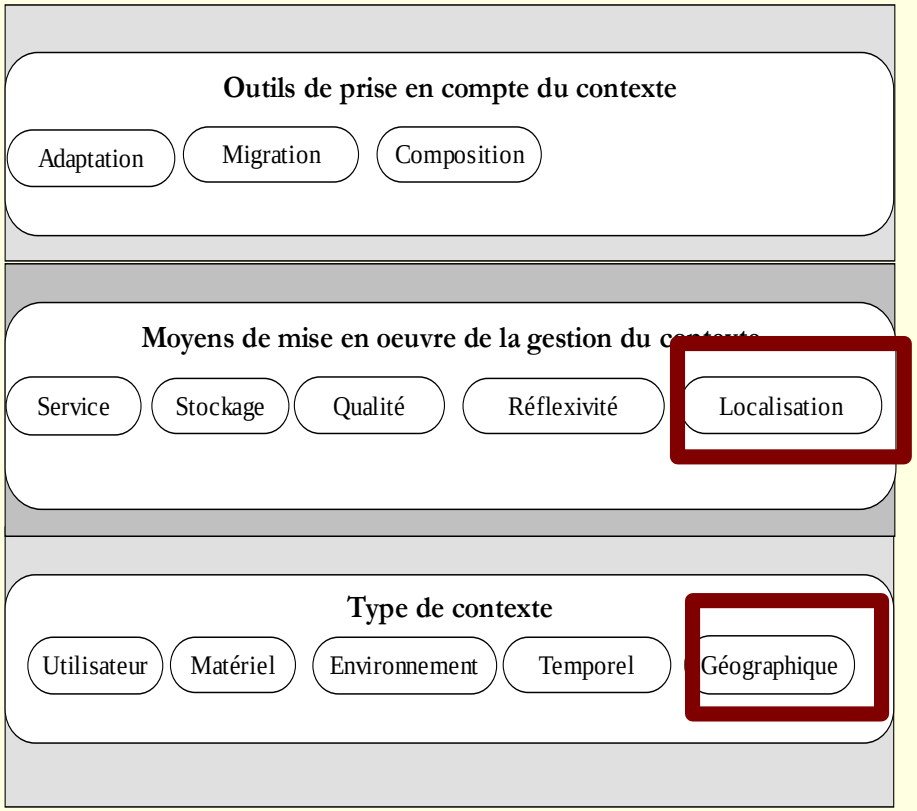
Partie 2

Évolution des applications

- Prise en compte du contexte physique
  - des composants (localisation, vitesse de déplacement, luminosité, etc.)
  - de l'environnement d'exécution (Capteur, Téléphone, PDA, PC)

- Ajout d'informations utiles pour :
  - Reconfigurations dynamiques
  - Adaptations plus pertinentes.

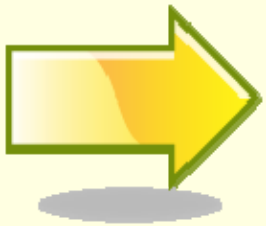
ANR TCAP JCJC 2006
Thèse de C. Louberry



Exemple d'application

- Surveillance :
  - Capteurs : infrarouge, température, etc.
  - Caméras
  - Composants logiciels de traitement : analyse d'images, etc.

Logiciel de détection de mouvement



Détecte et localise la présence d'intrus

- Approche zone dangereuse : Caméra + logiciel capture vidéo
  ➔ affiche la vidéo et suit l'intrus
- Améliorer la qualité du service rendu

Problématique

- L'informatique ubiquitaire :
  - **Problématique** : gérer un matériel/logiciel de plus en plus **hétérogène** et **mobile**
- Emergence des capteurs sans-fil ces dernières années
- Nombreux défis dans les domaines des réseaux et des architectures logicielles
- Optimisation des ressources :
  - **Hôte** : énergie (batterie), capacité de calcul, etc.
  - **Réseaux** : congestion, agrégation de données, etc.

Problématique

- Utilisation des capteurs
  - principalement pour leurs fonctions propres de mesures de l'environnement
- Gestion du contexte
  - adapter des services en fonction du lieu, de l'heure, etc.
- Peu de travaux sur l'utilisation des capteurs comme supports d'application
- Peu de travaux sur la gestion du contexte comme outil de gestion de la QdS

Problématique

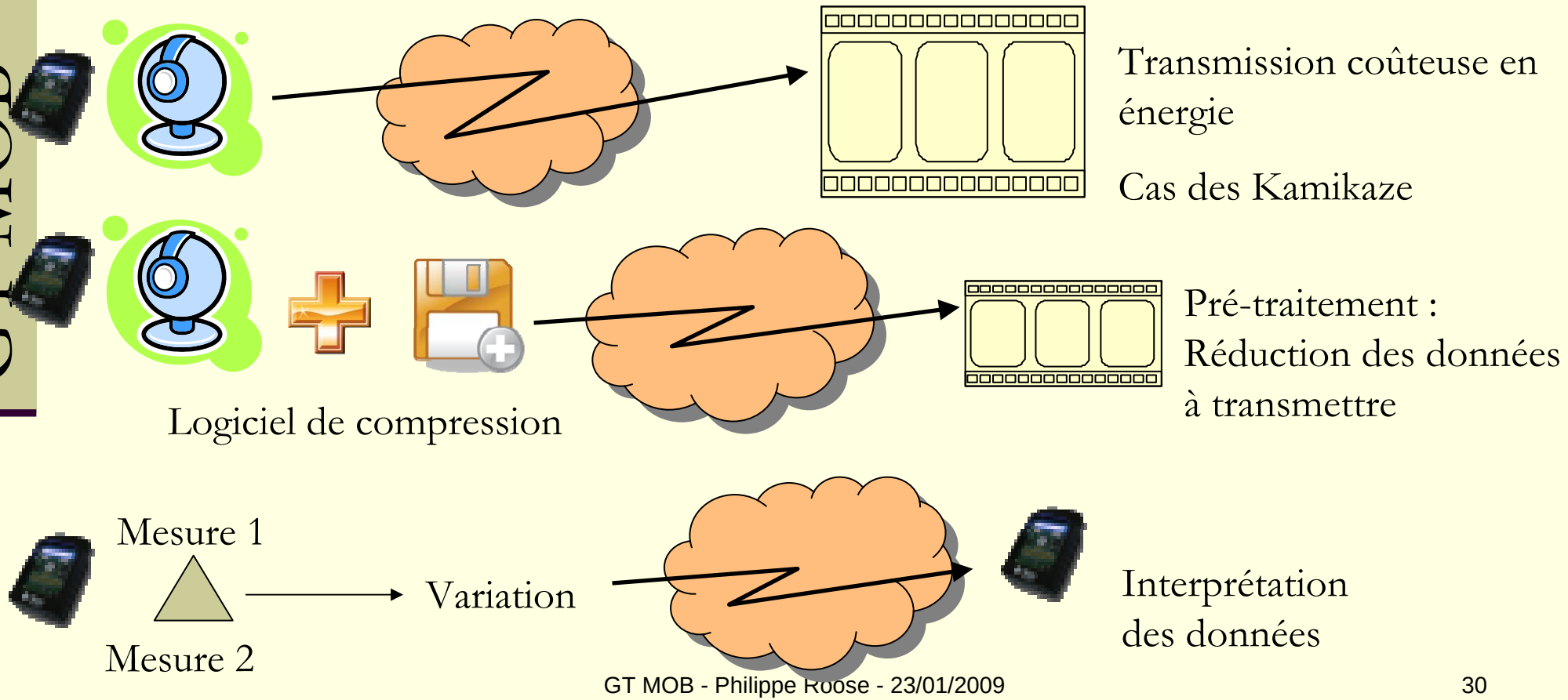
- Capteurs : dispositifs effectuant des mesures de l'environnement et transmettant de l'information, dotés d'une capacité de calcul et de mémoire.

➔ **Peuvent héberger des composants logiciels en relation ou non avec leur fonction**

Même fonctionnalité, plusieurs possibilités

➔Proposition de nouvelles configurations

➔Accroissement de l'offre de QdS



Proposition

- Utiliser les capteurs comme support de composants logiciels
  - Possibilité de minimiser les informations transmises (traitement local)
  - Possibilité de réaliser des traitements en cours de transferts (relai)
- Possibilité de gestion de ressources (délocaliser un composant sur un périphérique moins limité)
  - Possibilité de prise en compte du contexte
- Système sensible au contexte : adapte les informations ou les services selon les circonstances courantes d'utilisation

Objectif

- Proposer une plateforme de supervision sensible au contexte pour les applications distribuées.
  - Contexte : principale source d'information pour l'évaluation de la QdS

➔ Besoin d'identifier les informations contextuelles utiles à la plateforme

GT MOB

Revenons sur la notion de contexte

- Pas de définition unique
- Origine : Schilit et Teimer
  - « *Capacité d'une application et/ou d'un utilisateur mobile de découvrir et réagir aux changements de sa situation* »
- Extension de la définition de David et Ledoux
  - « *Le contexte d'exécution d'une application regroupe toutes les entités et situations externes qui influent sur la qualité de service/performance (quantitative et qualitative) telle que perçue par l'utilisateur* »
  - Perception également par le système
- Toute évolution du contexte peut provoquer une évolution de la qualité du service rendu

Définition générale de la QdS

- Trois niveaux de qualité de service
  - Infrastructure
  - Application
  - Utilisateur

Types de QdS

GT MOB

Au niveau infrastructure

- Garantir la continuité de service malgré les défaillances matérielles et réseaux et l'hétérogénéité des périphériques

Au niveau application

- Maximiser la durée de vie de l'application (pérénité)

Au niveau utilisateur

- Garantir le respect des contraintes de l'utilisateur
- Qualité de service : service adéquat

Types de contexte

- Une application fonctionne grâce à une infrastructure et est utilisée par des utilisateurs

➔ Contexte d'exécution

➔ Contexte d'utilisation

Contexte d'exécution

- Ensemble des paramètres permettant de caractériser le support d'exécution de l'application
  - L'hôte
  - Le réseau

Paramètres	Continuité	Pérennité
Niveau d'énergie		X
Mémoire utilisée	X	
Mémoire disponible	X	
Type d'hôte		X
Temps d'exécution	X	
Débit	X	X
Taux d'erreur	X	X
Taux de congestion	X	X
Longueur du chemin		X
Délai	X	

Contexte d'utilisation

- Ensemble des informations caractérisant l'application et la satisfaction de l'utilisateur face au service rendu
- Informations nécessaires afin de respecter les contraintes définies par l'utilisateur et le concepteur
- Exemple : diffuser des images en couleur lorsqu'un mouvement est détecté

Plateforme

Composition :

- Supervision
- Usine à Conteneur
- Usine à Connecteur
- Routage

Distribution de la PF sur les différents hôtes de l'application

Service Supervision

Service principal :

- Réception des informations de contexte
- Évaluation QdS de l'application
- Décision de reconfiguration
- Transmission de requête de reconfiguration aux autres services
  - Déploiement de conteneurs
  - Déploiement de connecteurs

Service Usine à Conteneur

- Création de conteneurs adaptés à l'hôte (PC, PDA, SmartPhone, Capteurs).

- Actions :
  - Réception requête service Supervision
  - Téléchargement/instanciation du composant métier
  - Encapsulation
  - Déploiement
  - Lancement
  - Arrêt
  - Suppression

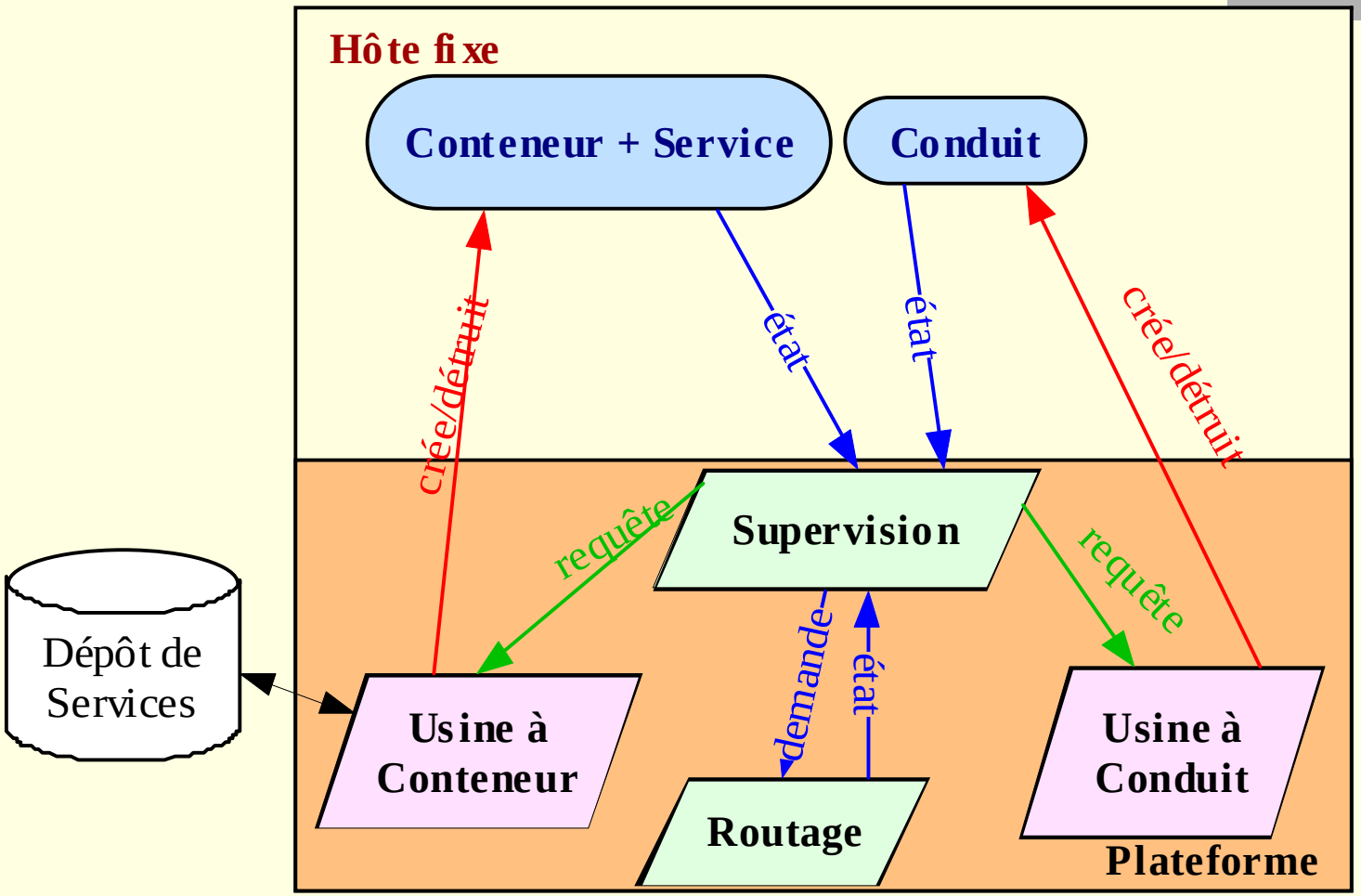
Service Usine à Connecteur

- Création des connecteurs
  - ➔Implémentation de la politique de communication choisie.
    - Synchronisation, Temps-réel, avec/sans pertes, etc.
- Actions :
  - Réception requêtes service Supervision
  - Déploiement selon l'hôte (CS, RPC, etc.)
  - Connexion/Déconnexion du conteneur de service (service Routage)
  - Suppression

Service Routage

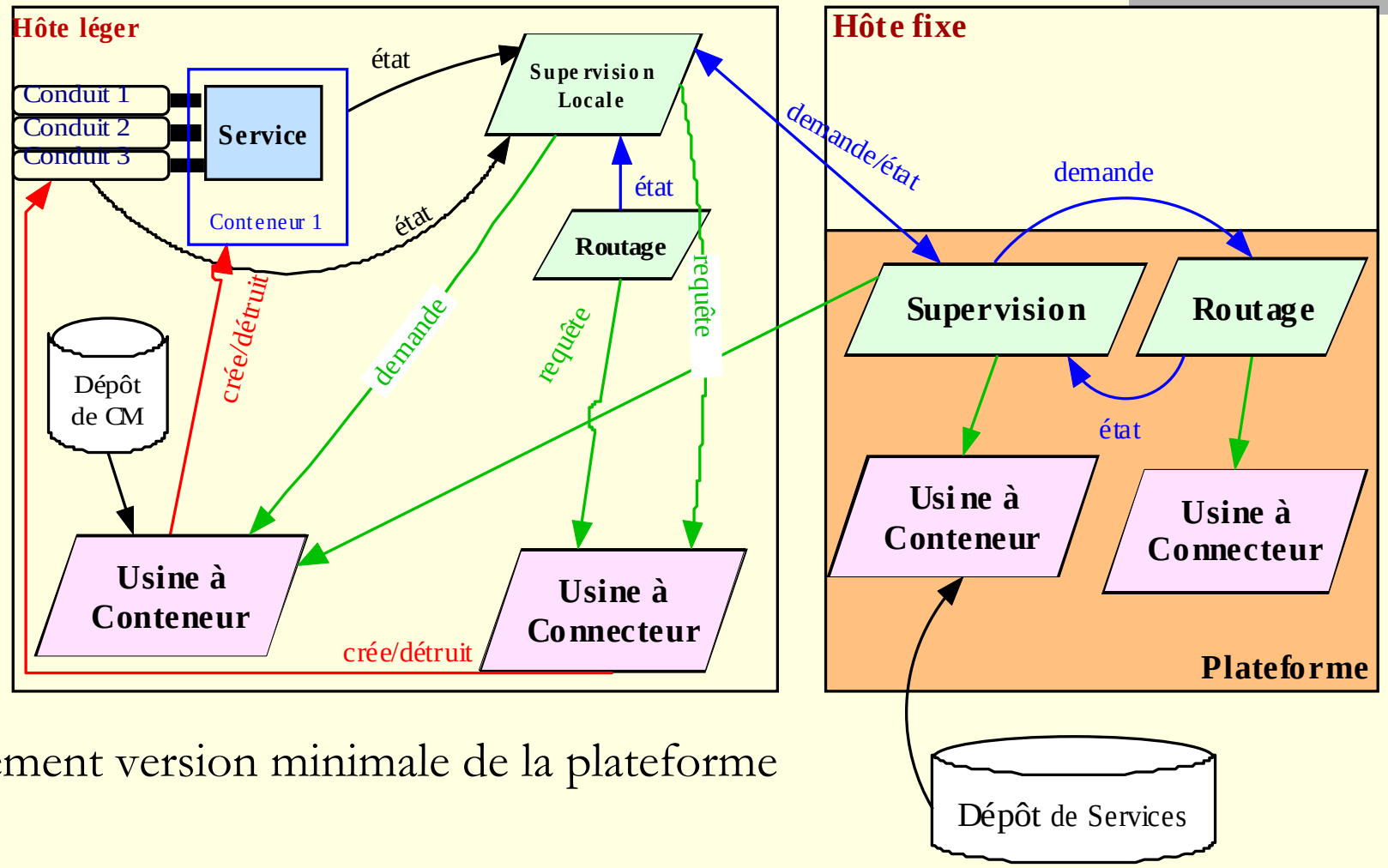
- Création et mise à jour d'une table de localisation pour atteindre les composants de l'application.
- Création des chemins d'accès aux hôtes.

Scénario Hôte fixe



Déploiement complet de la plateforme et des composants

Scénario Hôte léger



Déploiement version minimale de la plateforme

Conclusion sur la PF

- Gestion et reconfiguration dynamiques des applications distribuées en environnement contraint
- Capteur : nouveau support pour les fonctionnalités ➔ traiter les informations et minimiser les transferts
- PF : Collaboration de services
- Évaluation QdS et sensibilité au contexte
  - Assurer le service
  - Maximiser la durée de vie
  - Respecter les contraintes utilisateurs/applications

  } Informations contextuelles
- Ajout/suppression/déplacement de composants/connexions
- Déploiement selon contraintes (CDC/CLDC) et contexte fonctionnel

GT MOB

Perspectives sur la PF

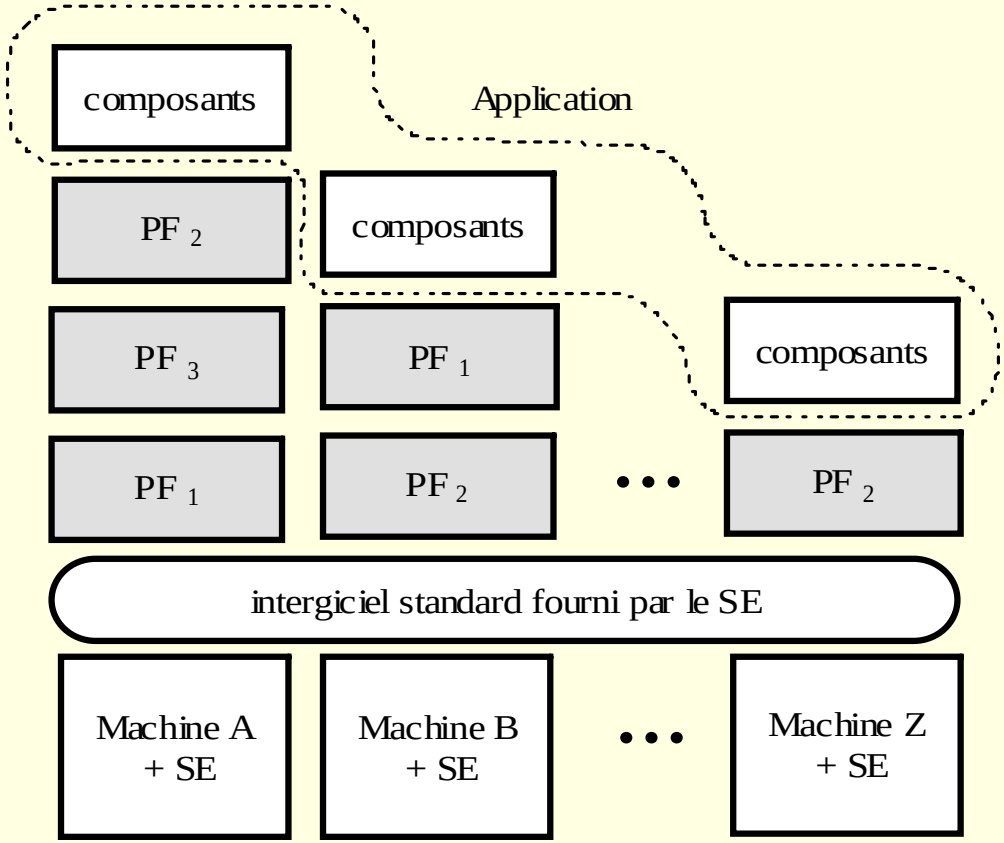
- Développement et déploiement de la PF sur capteurs, téléphones mobiles et PDA.
  - Caractériser les informations contextuelles nécessaires à la gestion de la QdS
  - Prototypage avec différents périphériques : capteurs, PDA, téléphones mobiles (différents modes de communication)
  - Valider le fonctionnement de la PF
  - Mesures de performance
- Étude de l'algorithme de décision de reconfiguration et de déploiement (heuristique)

Perspectives sur la PF

- **Enjeux**
  - Périphériques contraints => PF légères
  - Mobilité et contexte environnemental => reconfiguration (PF intrusives)
- **Verrous**
  - Tendance à la complexité
  - Difficultés liées à la reconfiguration
  - Problème de stabilité

Perspectives sur la PF

- Plates-formes légères, spécialisées et cumulables
- On installe celles dont on a besoin

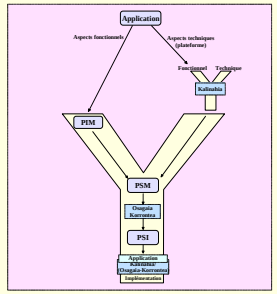


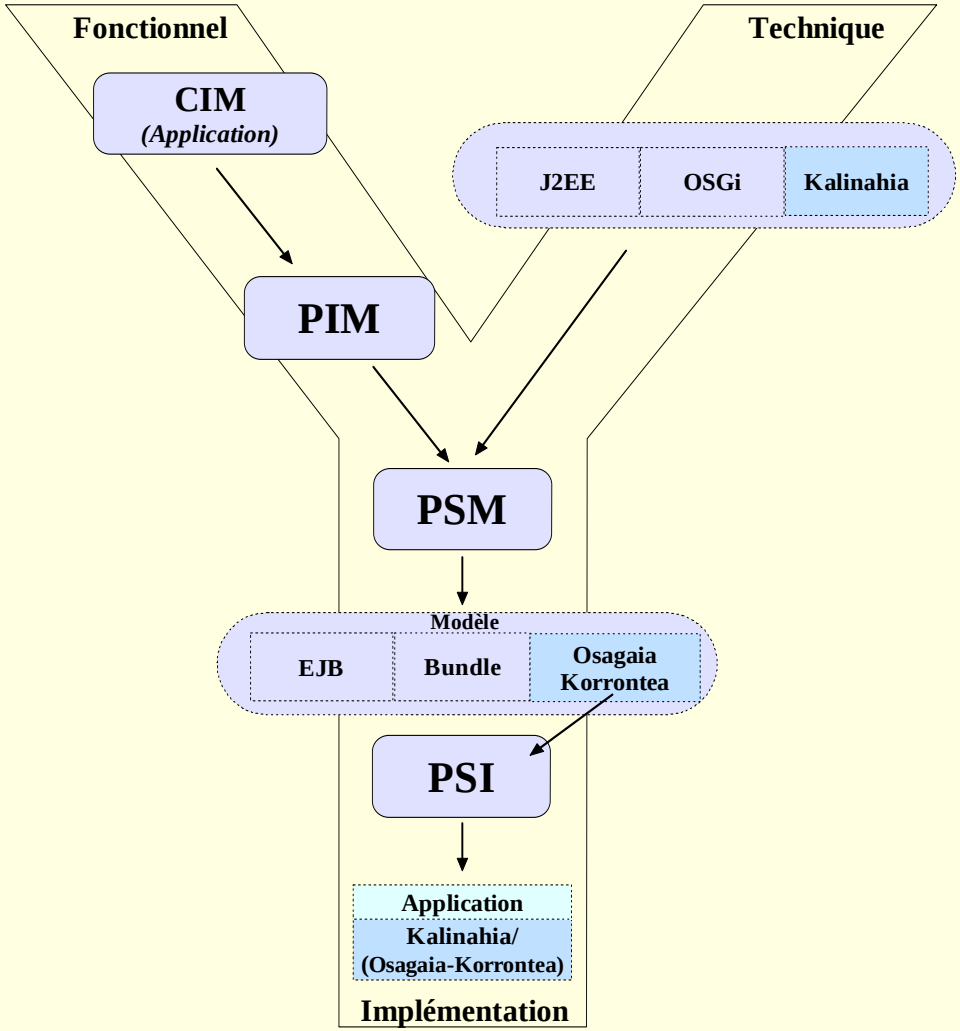
Empilement de plates-forme

En parallèle…

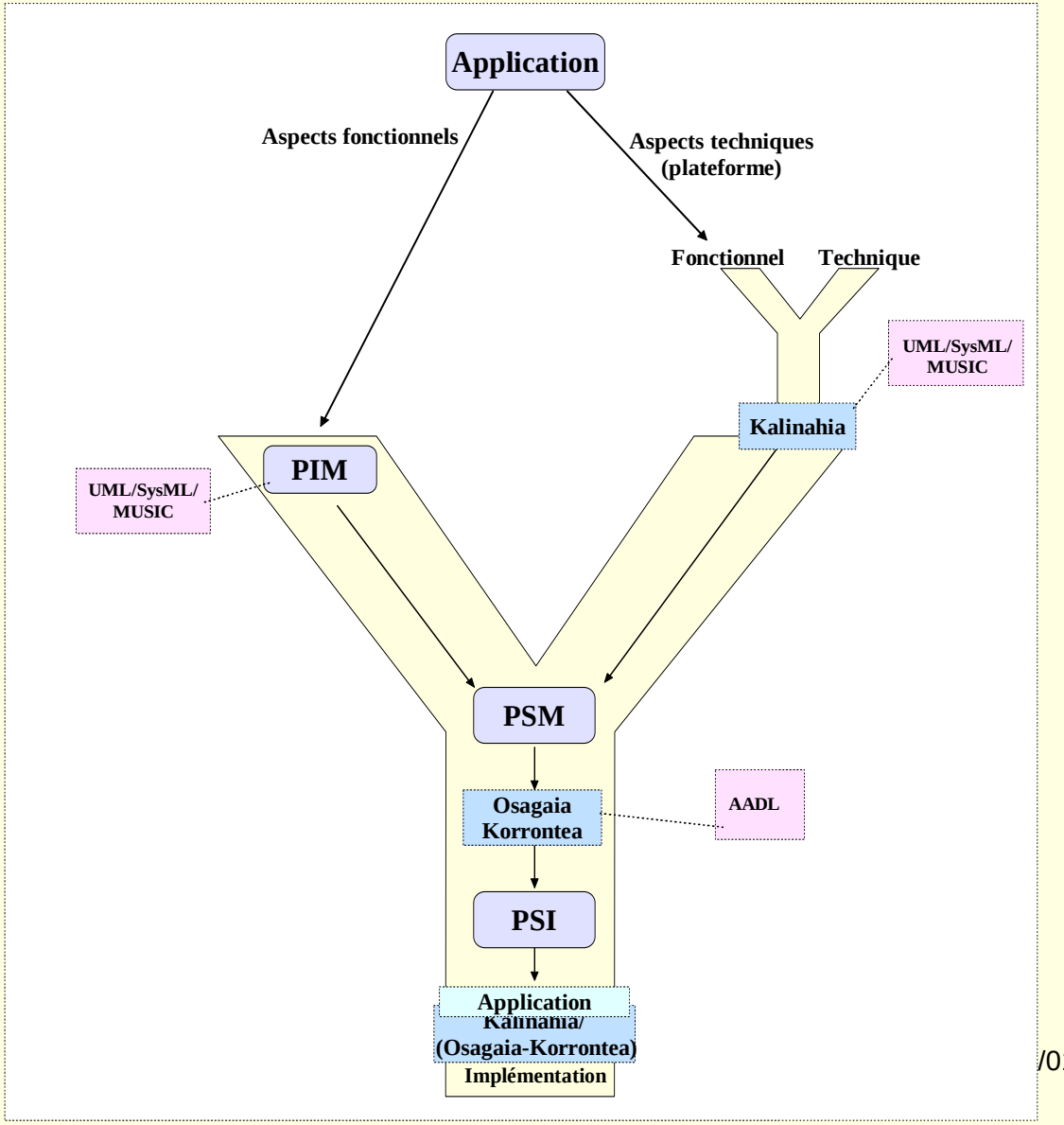
- Solutions envisageables
  - Approche plus généraliste/modèles
  - Modélisation à base d'UML
  - Automatisation de la démarche
  - Vérification des modèles

Scénario habituel





Scénario idéal



[Henzinger et Sifakis, 2006]

« l'influence directe de l'environnement sur le comportement du système fait que le logiciel et la plate-forme d'exécution ne peuvent plus être étudiés séparément »

Questions

Architecture Logicielle pour l'Adaptation Dynamique d'Applications sur Périphériques Contraints