Une Architecture à Base de Composants pour la Gestion de la Qualité de Service dans les Systèmes Embarqués Mobiles

Jean-Charles Tournier

Vendredi 1er Juillet 2005

Thèse effectuée sous la direction de

Jean-Philippe Babau, Stéphane Ubéda (CITI/INSA Lyon)

et

Vincent Olive (France Télécom R&D)

Plan

- Contexte
- Problématique
- Proposition d'architecture : Qinna
- Expérimentations et analyses
- Conclusions et perspectives

Contexte - Systèmes embarqués mobiles

- Nombreuses contraintes
  - Économiques, ergonomiques, mobilités
- Ressources matérielles limitées et spécifiques
  - CPU, mémoire, batterie, réseau
- Applications de plus en plus complexes
  - Multimédia => QdS
- Systèmes dédiés
    - Matériel (SoC, DSP) et logiciel (OS, applications)
- Besoin d'ouverture et de flexibilité
  - Ajout/retrait/modification dynamique de services ou d'applications





Contexte - Systèmes embarqués mobiles

- **Gestion dynamique et sûre de la QdS de niveau ressources**

1. Générique : non liée à une politique ou un langage de spécification
2. Hétérogéne : gestion des contraintes de QdS hétérogènes
3. Dynamique : gestion dynamique de la QdS
4. Auto-configurable : détermination des niveaux de QdS requis
5. Confiante : s'assurer du niveau de QdS consommé
6. Réutilisable : réutilisation des mécanismes de gestion de QdS

Contexte - Programmation par composants

- Développement par assemblage de briques logicielles
- Bénéfices
  - Réutilisabilité
  - Partitionnement
  - Adaptation
- Différents domaines d'applications
  - Applications distribuées
    - EJB, CCM, .NET, DCOM, OSGi
  - Systèmes embarqués
    - PECOS, VEST, Koala, Fractal/Think, Rubus

Contexte - Programmation par composants

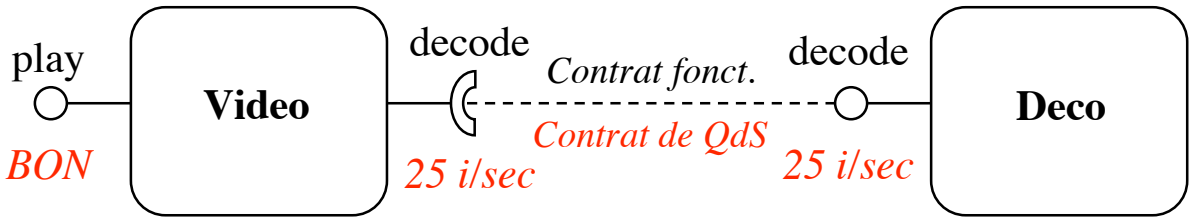
- Gestion de la QdS
  - Généricité
    - Politiques de QdS prédéfinies
      - EJB : persistance, Koala : mémoire, VEST : ordonnancement TR
  - Hétérogénéité
    - Pas prise en compte
  - Dynamicité
    - Pour les QdS autres que de niveau ressources (persistance, sécurité, etc.)
      - EJB, CCM, .NET
  - Auto-configuration
    - Pas prise en compte
  - Confiance
    - Pour les gestions de QdS statique
      - Koala, PECOS, VEST
  - Réutilisation
    - Pour les QdS autres que de niveau ressources
      - EJB, CCM

Problématique

- **Gérer à l'exécution la QdS de niveau ressources des systèmes embarqués ouverts à base de composants**
  - Générique, dynamique, hétérogénéité, confiance, auto-configurabilité, réutilisabilité

Prise en compte de la QdS dans les modèles à composants

- Modèle de composants simple
  - Interfaces requises et fournies
    - Uniques points de communication
  - Propriétés configurables
  - Composition de composants => contrat
  - Contrat ***[BJPW99]***
    - Fonctionnel, Pre/post conditions, Synchronisation, **QdS**



Principes de gestion de contrats [ACH98]

- Spécification
  - Niveau de Performance
  - Niveau d'importance
  - Politique d'adaptation
- Initialisation
  - Mapping
  - Test d'admission
  - Réservation
- Gestion
  - Observation
  - Maintenabilité
  - Adaptation

Qinna - Principes généraux

- Séparation des préoccupations
  - Fonctionnelles et de QdS
- Séparation des mécanismes et politiques de gestion de QdS
  - Une politique définit quand une action doit être menée
  - Un mécanisme définit comment réaliser l'action
- Identification dynamique des composants
  - Identification des composants à l'exécution
- Tout est composant
  - Du niveau applicatif au niveau ressources en passant par les niveaux services et système d'exploitation

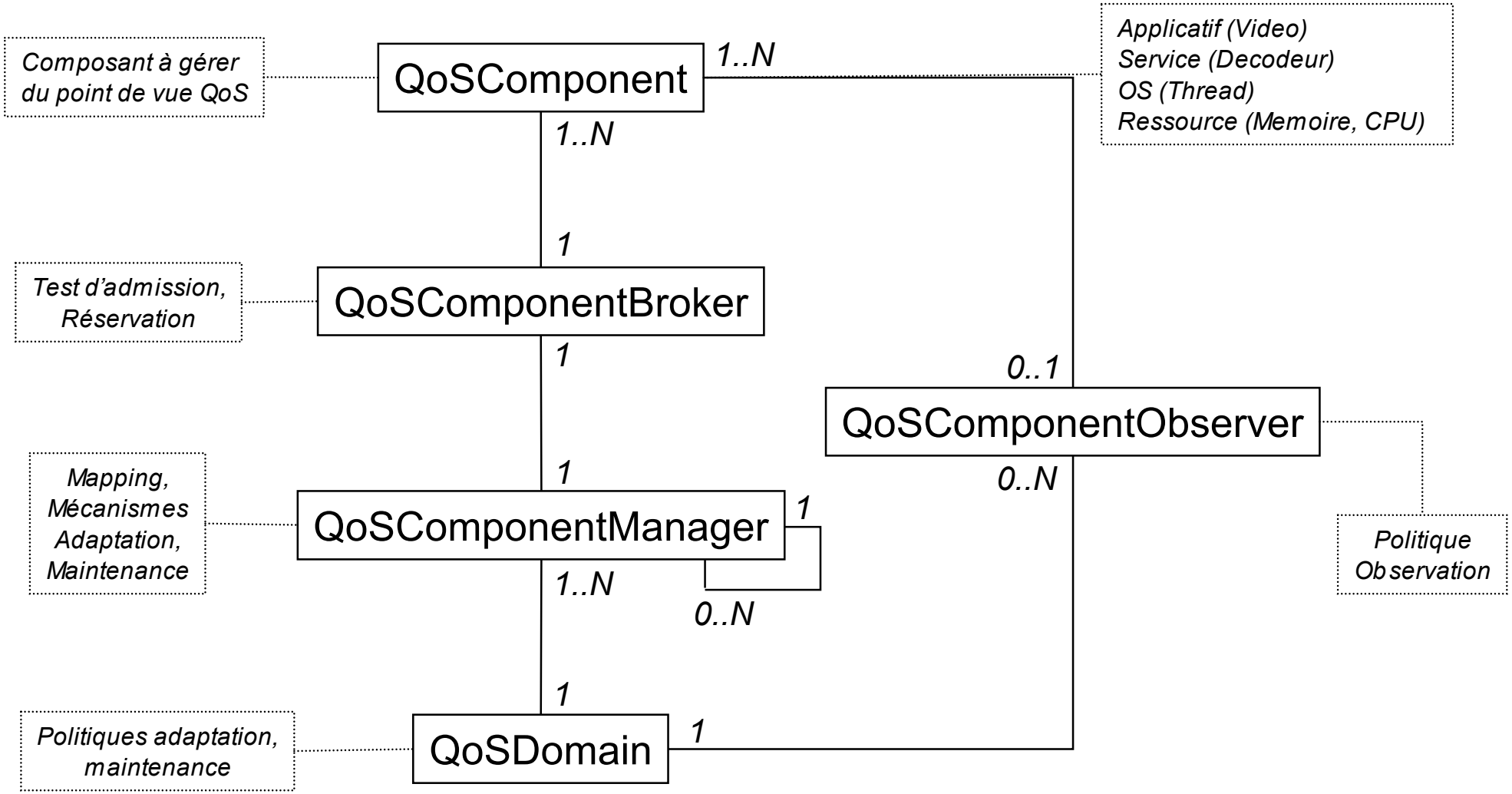
Qinna - Résumé des objectifs

- Systèmes à composants
- Contrats de QdS
- Principes de gestion de contrats de QdS
  - Spécification, initialisation et la gestion
- Principes de génie logicielle
  - Séparation des préoccupations
  - Séparation des politiques et mécanismes

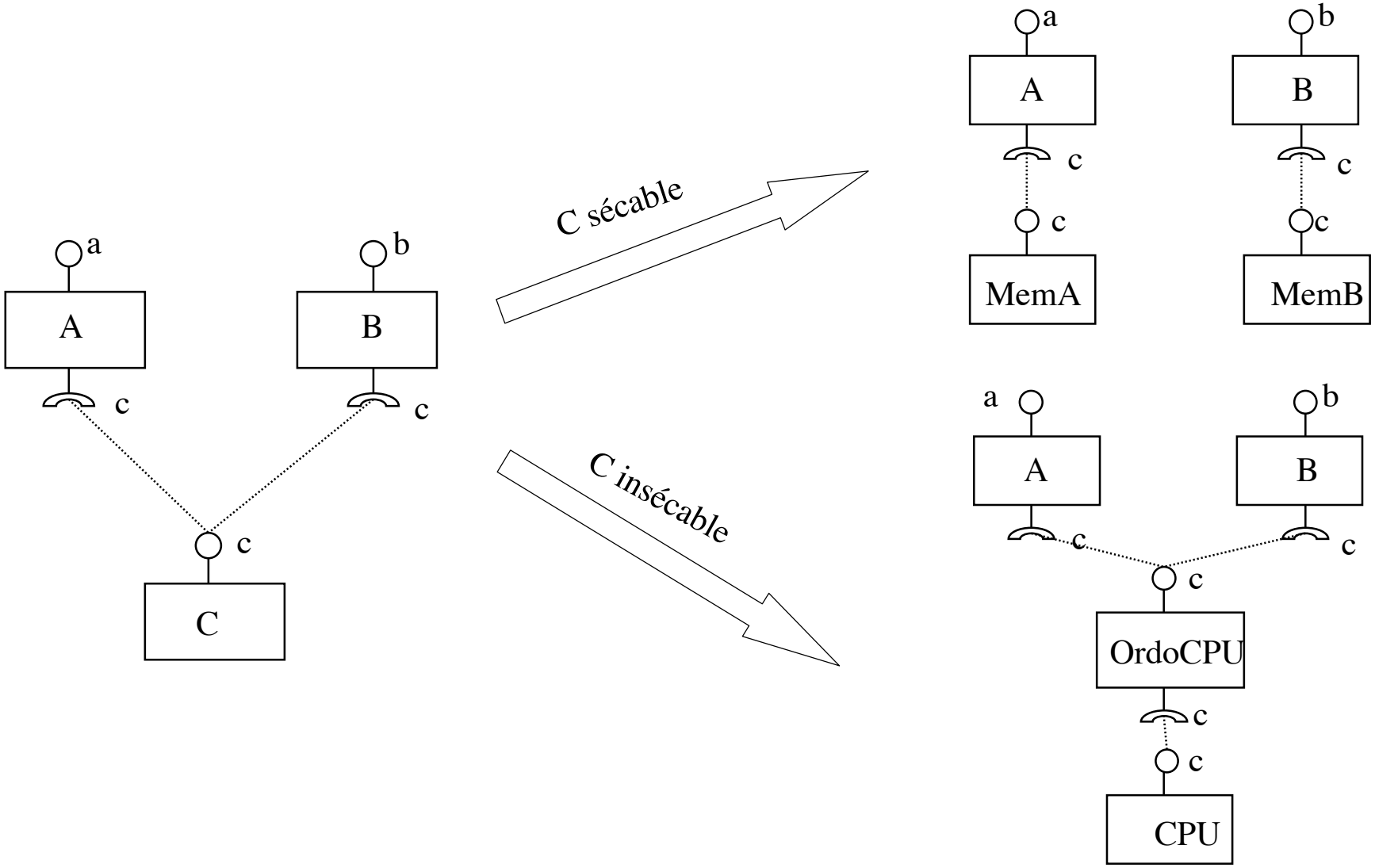
Qinna - Définition

- Qinna est une architecture définie par :
  - Des types de composants
    - Composants fonctionnels et de gestion de QdS
  - une API
    - Spécifier et gérer les contrats de QdS
    - Configuration des composants
  - Des types abstraits de données
  - Les comportements dynamiques
    - Établissement/annulation de contrat
    - Observation de contrat
    - Adaptation/maintenance de contrat

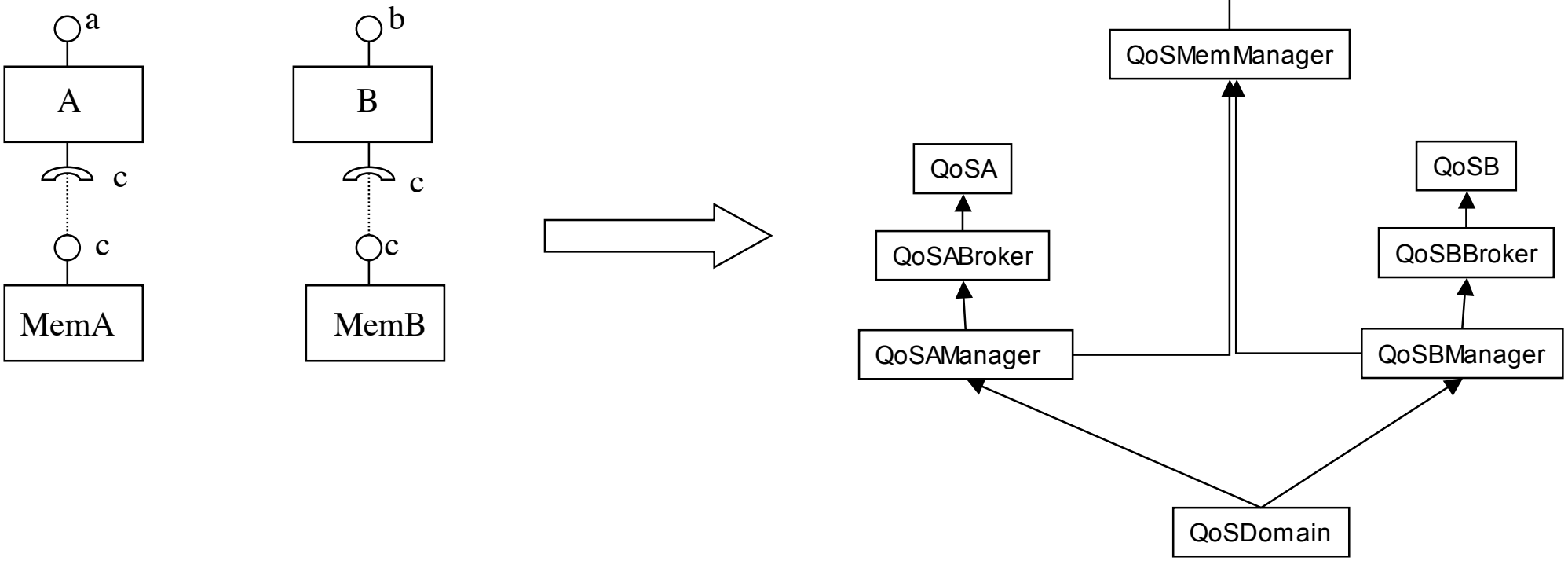
Qinna - Vue globale



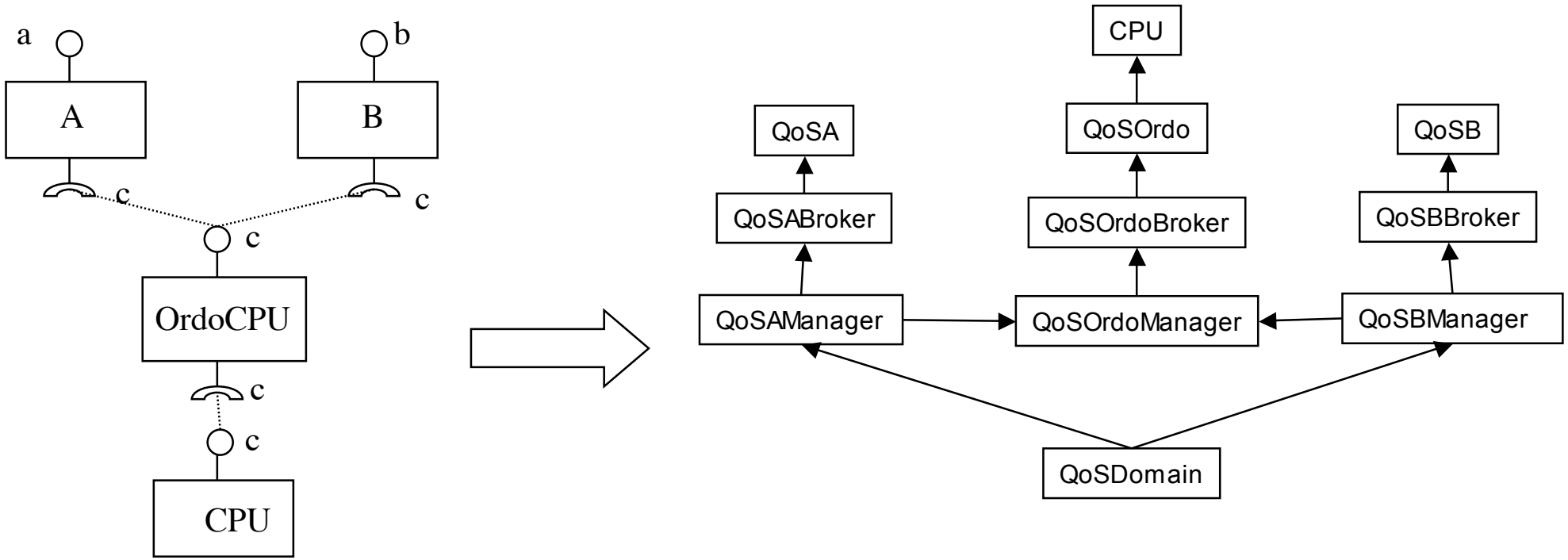
Intégration de l'architecture Qinna



Composant sécable



Composant insécable



Analyses et expérimentations

Qinna vs Généricité

- Réponses
  - Définition à base de type de composant (boîte vide)
  - API générique
- Expérimentations
  - Systèmes TR (EDF, régisseur, TR dur/relâché, etc.) et multimédia
    - Publications à WACERTS/RTSS 2004 et SAC 2005
  - Nécessité de disposer de plusieurs modes de fonctionnement
  - Découpage en sous contrats => expression de QdS sur chaque sous contrat
- Conclusions
  - Adaptée aux applications ayant plusieurs modes de fonctionnement
    - Adaptation dynamique due à l'évolution du système et/ou de l'environnement
  - Permet la structuration des systèmes
    - Identification claire des différentes opérations de gestion de QdS

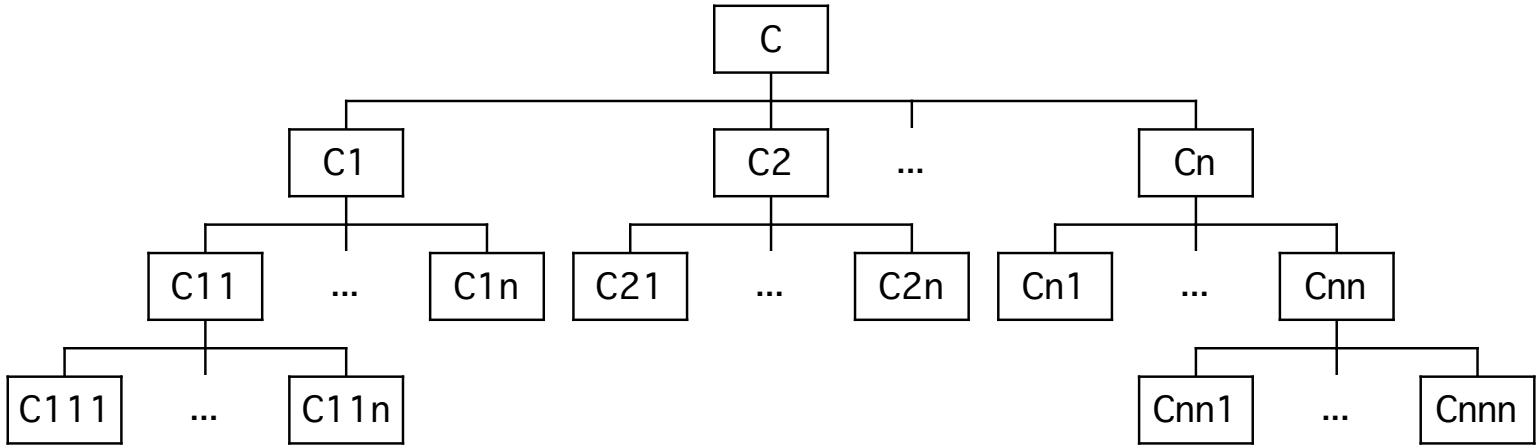
Qinna vs Dynamicité - 1/3

- Réponses
  - Contrats de QdS inter composants
  - Intégration des principes de gestion de contrats de QdS (spécification, initialisation, observation, adaptation et maintenance)

- Expérimentations
  - Evolution du système logiciel
    - Arrivée/départ de composants
    - Profil variable de QdS
    - Relation d'ordre des niveaux d'importance
  - Evolution des capacités des ressources
    - Batterie, Réseau, CPU

Qinna vs Dynamicité - 2/3

- Couts
  - Granularité des composants



  - Parcours exhaustif de l'arbre implicite
  - Pire cas d'initialisation (echec du N^{ème} sous contrat)
    - Etablissement, puis annulation des N-1 contrats

Qinna vs Dynamicité - 3/3

- Coûts (suite)
  - Nb de niveaux de QdS des composants (Q)
  - Nb de contrats en cours de gestion (M)

    => Nb maximal d'opérations: **P=2N.Q.M**
  - Efficacité dépendante de l'implémentation
    - Par ex: choix des premiers nœuds à explorer de l'arbre

- Conclusions
  - Réponse adaptée aux systèmes visés
    - Nombre limité de composants
    - Ressources matérielles et logicielles connues

Qinna vs Hétérogénéité

- Réponse
  - Opérateur ***q2 translate(T_QoS q1)*** définit par les QoSComponentManagers
    - Traduction d'un niveau de QdS
- Experimentations
  - Ordonnancement de tâches apériodique sur un ordonnanceur périodique
  - Ordonnancement de tâches TR relâchées sur ordonnanceur TR dur
  - Effectue un changement sémantique
  - Coût évalué en termes d'opérations à effectuer et/ou de gaspillage de ressources
- Conclusion
  - Nécessité de prévoir les différentes formes d'hétérogénéité pour limiter les coûts

Qinna vs Confiance

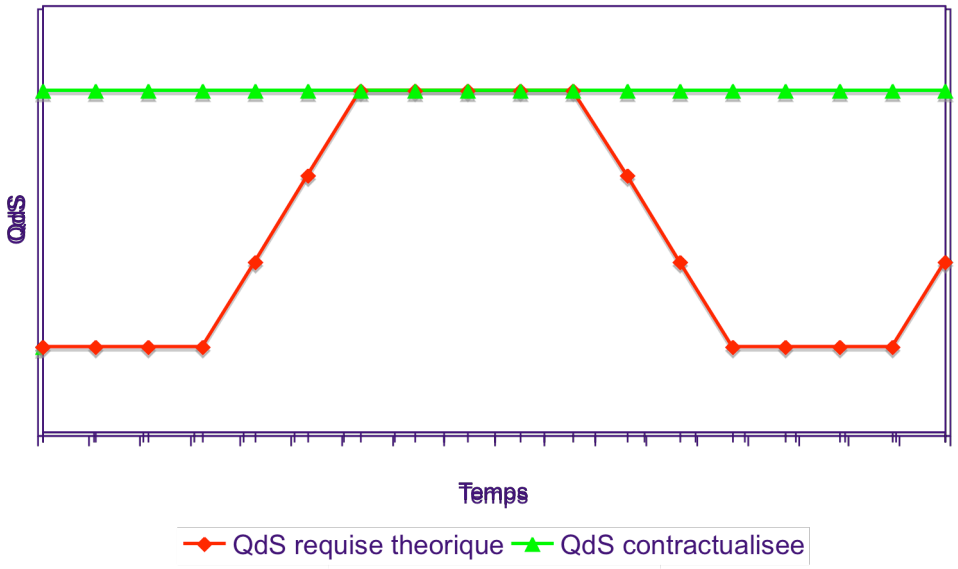
- Réponses
  - Contrôle des niveaux de QdS fournis par les QoSComponents
- Expérimentations
  - Réalisation des contrôles des niveaux de QdS pour les composants ressources (mémoire, sockets, threads)
  - Impossibilité de fournir une QdS supérieure à celle contractualisée
- Conclusion
  - Difficulté d'implémenter les contrôles des QoSComponents pour les composants autres que ressources

Qinna vs Auto-configurabilité - 1/2

- Réponses
  - Attributs **reliable/unreliable** des contrats de QdS
  - Valeurs ***default*** des niveaux de QdS requis
  - Mise en place d'observateurs
    - Observations périodiques ou événementielles
  - Mise en place des politiques et mécanismes de maintenance

- Expérimentations
  - Détermination de niveau de QdS requise
    - Constant
    - Variable
  - Publication à ICSE/CBSE 2005

Qinna vs Autoconfigurabilité - 2/2



- Stratégies
  - Contractualisation au niveau max
    - –Gaspillage de ressources
    - –Niveau max inconnu
  - Contractualisation au niveau min
    - –Violation de contrat
      - –Retard si observation périodique
      - –Opérations de maintenance d'un pas *P*

- Conclusions
  - Fortement lié au profil de QdS
  - Compromis entre nb d'opérations de maintenance, gaspillage de ressources, et retard
  - Impacte sur la mise en place des politiques de QdS par la création de retard

Qinna vs Réutilisabilité

- Réponses
  - Architecture à base de composants
  - Séparation des préoccupations fonctionnelles et de QdS
  - Séparation des politiques et des mécanismes de QdS
  - API pour la configuration des composants (par ex. la relation d'ordre du QoSDomain)

- Expérimentations
  - Bibliothèque de composants Qinna
  - Réutilisation du même QoSDomain pour plusieurs expérimentations
  - Réutilisation des QoSComponentBrokers pour des admissions simples

- Conclusions
  - Réutilisation liée aux objectifs d'hétérogénéité et d'auto-configurabilité

Conclusions

- Contexte
  - Systèmes embarqués mobiles à composants
- Qinna
  - Architecture à base de composants de gestion de QdS des systèmes ouverts à composants
  - Définition à l'aide de composants
    - Structuration claire de l'architecture
    - Identification de chaque opération de gestion de QdS et de son coût
  - Contrats de QdS
    - Expliciter les relations de QdS entre les composants
    - Enveloppe de QdS pour une meilleure maîtrise de la QdS (cf. temps réel)
    - Compromis entre un gaspillage de ressources et un nombre d'opérations à effectuer

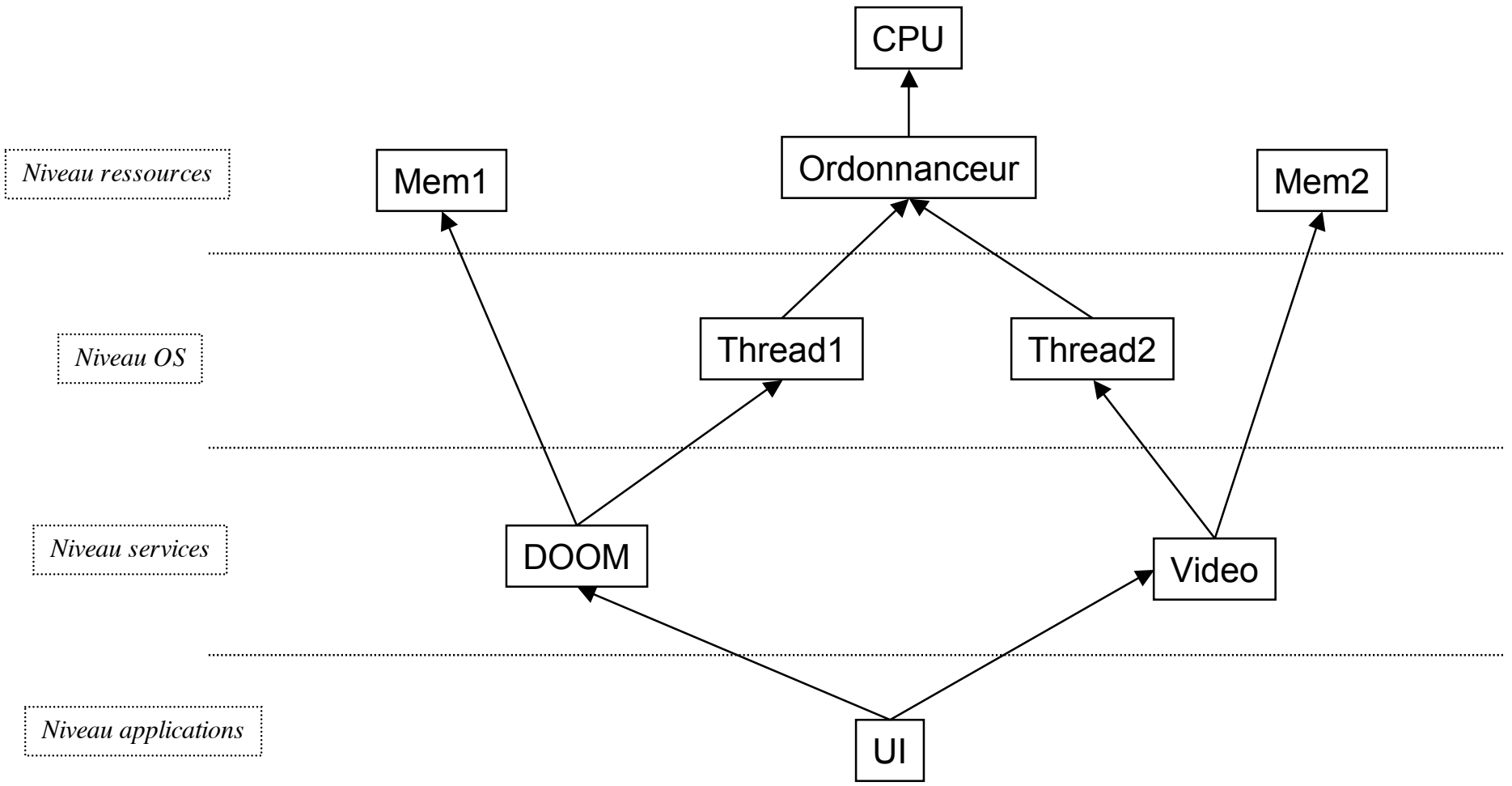
Perspectives

- Optimisations d'implémentation
  - Règles de parcours du graphe des sous contrats
  - Utilisation de méta-données
- Généralisation de l'approche
  - Autres propriétés non fonctionnelles (sécurité, FT, etc.)
  - Composition de propriétés non fonctionnelles
- Définition d'un méta-modèle de l'architecture
  - Génération automatique de l'architecture
  - Vérification des propriétés de l'architecture
- Application aux systèmes multiprocesseurs et distribués
  - Un seul QoSDomain vs coopération entre QoSDomain

Références

- **[BJPW 99]** Antoine Beugnard, Jean-Marc Jézéquel, Noël Plouzeau, and Damien Watkins. Making components contract aware. IEEE Computer, 32(7) :38–45, 1999.
- **[ACH 98]** Christina Aurrecoechea, Andrew T. Campbell, and Linda Hauw. A Survey of QoS Architectures. Multimedia Systems, 6(3) :138–151, 1998.

Exemple - 1/2


CPU
Ordonnanceur
Mem1
Mem2
Thread1
Thread2
DOOM
Video
UI
Niveau ressources
Niveau OS
Niveau services
Niveau applications

Exemple - 2/2

- Evaluations quantitatives
  - Taille totale: 749 ko
  - Taille Qinna: 11,5 ko => (1,5%)
  - Délai établissement QoSVidéo: 4, 08 ms
  - Délai établissement QoSDoom: 4,36 ms
    - Dont 1,08 ms de dégrad. vidéo