

Architecture Objet pour la Qualité de Service ARTO (Adaptable Real Time Objet)

Jose-Lino Contreras , Jean-Louis Sourrouille

[jcontrer | sou] @if.insa-lyon.fr

Laboratoire L3i - INSA de Lyon

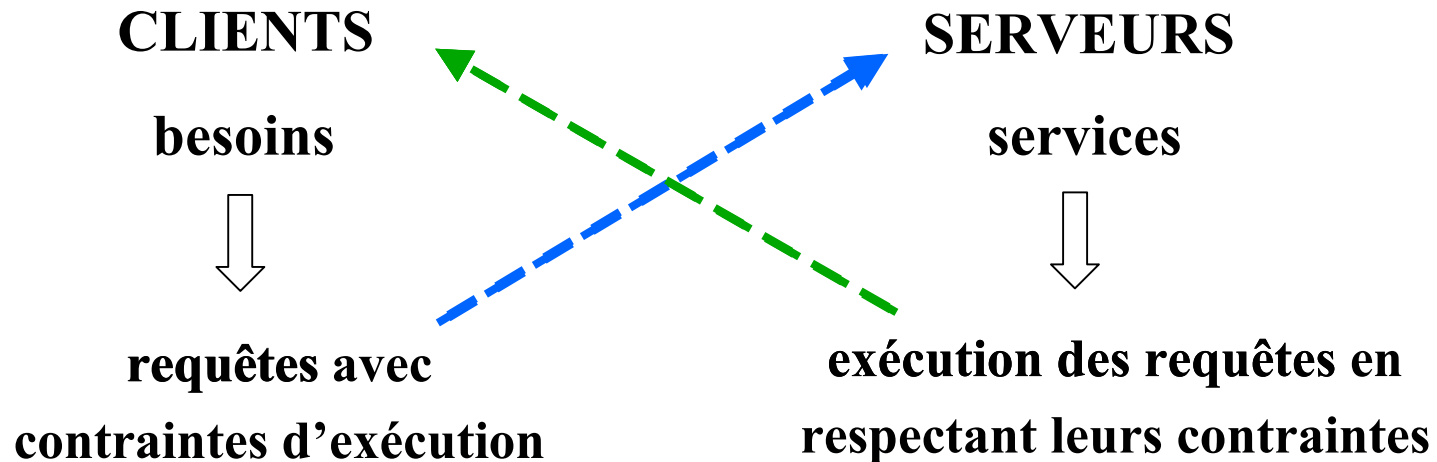


- Introduction
- Qualité de Service
- ARTO
- Conclusions

- **Modèle Général d'Objet Actif pour la Qualité de Service**
 - ☞ Fournir la meilleure QoS possible
- **ARTO: Objet Actif Adaptable pour applications temps réel**
Contexte d'exécution variable et inconnu a priori
Activités critiques et non critiques
 - ☞ critiques : respect des contraintes temporelles
 - ☞ non critiques : degrés de liberté, meilleur effort**Aspect temporels : un des aspects de la QoS**
- **Comment?**
 - ☞ comportement flexible
 - ☞ adaptation au contexte d'exécution
 - ☞ **respect des principes de l'approche objet !**

Clients, Serveurs, Contraintes, QoS ...

4

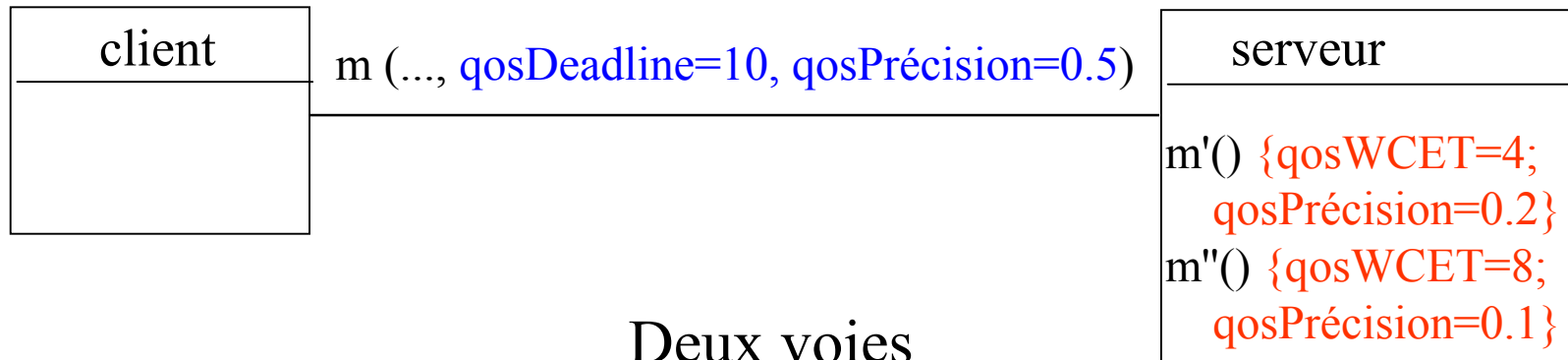


« *QoS ~ degré de satisfaction des besoins des Clients* »

Objectif : réalisation des services demandés
+
respect de contraintes d'exécution

UML + Caractéristiques de QoS...

Contraintes d'exécution \Rightarrow date d'échéance, précision, gigue...



Deux voies

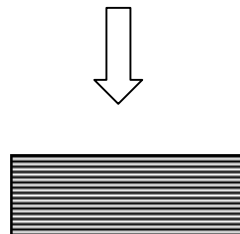
statique

dynamique

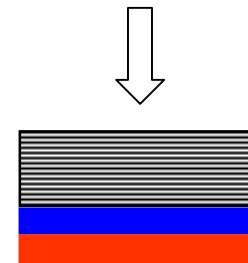
Assurer $QoS_D \leq QoS_F$

Satisfaire $QoS_D \leq QoS_F$

Optimiser QoS_F



Application

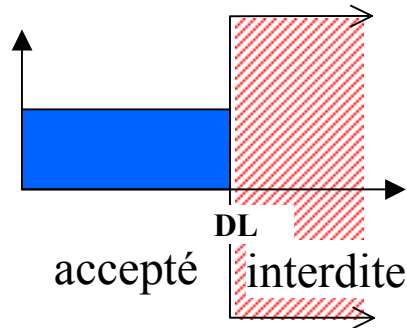


UML + Caractéristiques de QoS...

Voie statique



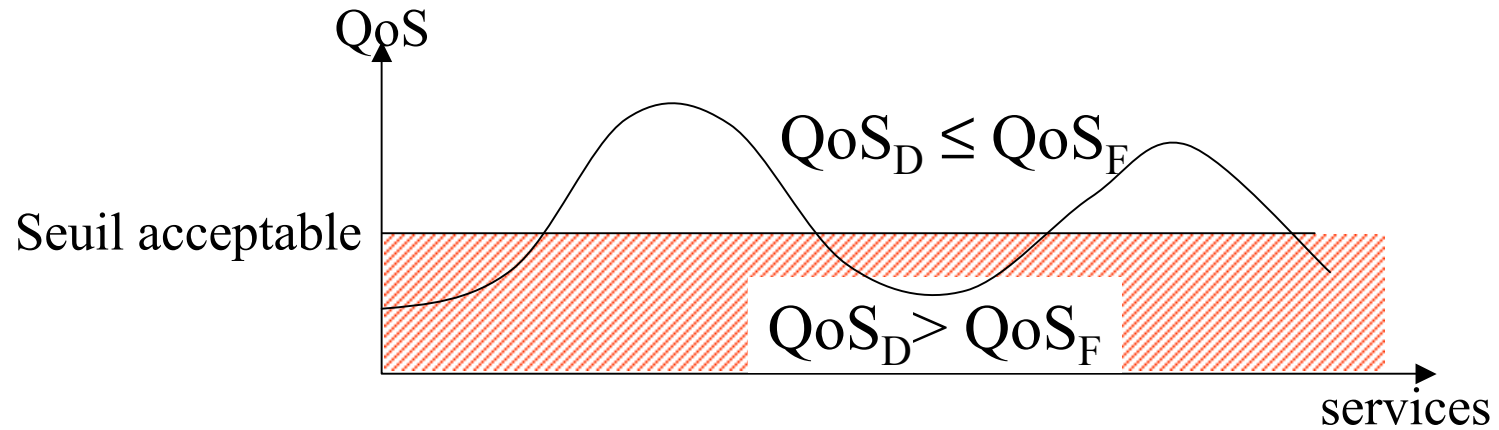
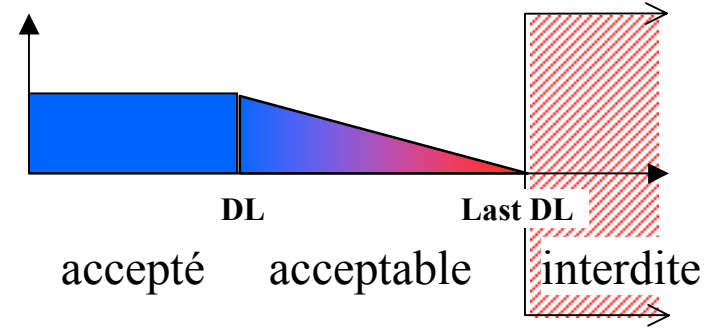
Tout ou rien ...
(généralement)



Voie dynamique



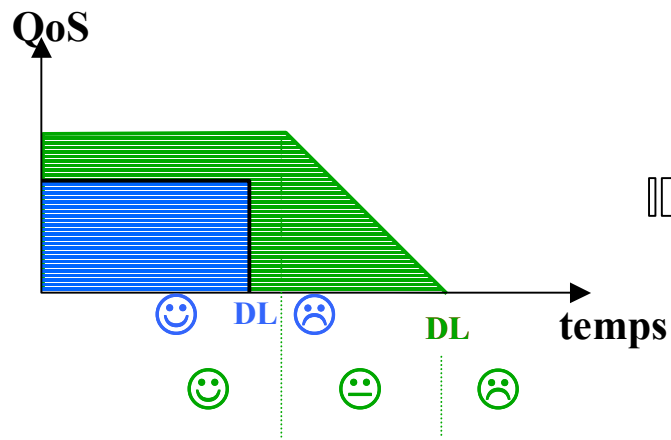
Best effort ...



QoS = f (critères)

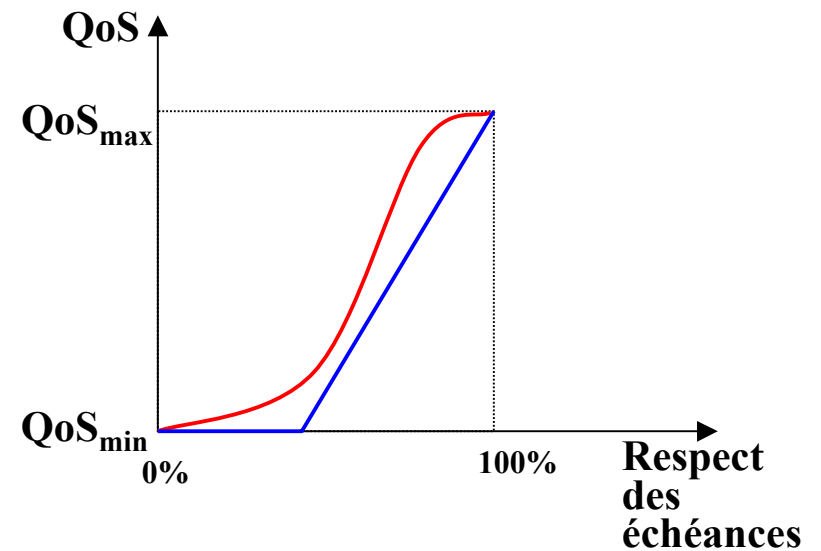
Exemple : C1 = respect des échéances

QoS = f(date service fourni)



Caractéristique de QoS_D pour l'échéance : date

QoS = f(nombre de respects des échéances)

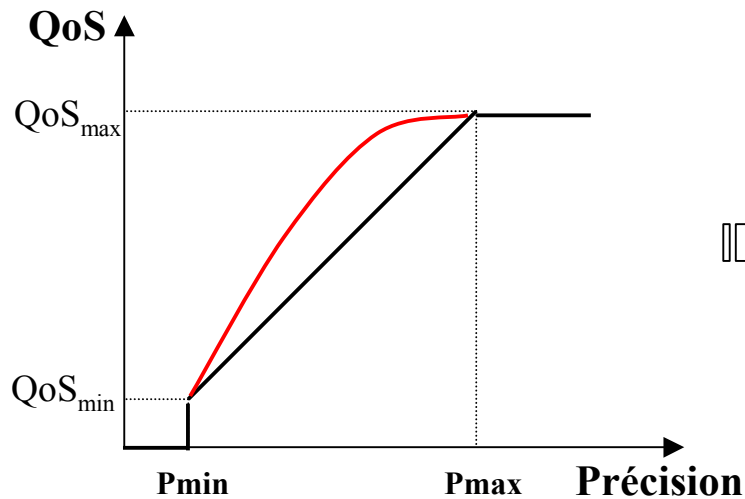


Critère de satisfaction de la QoS pour l'échéance : % de contraintes respectées

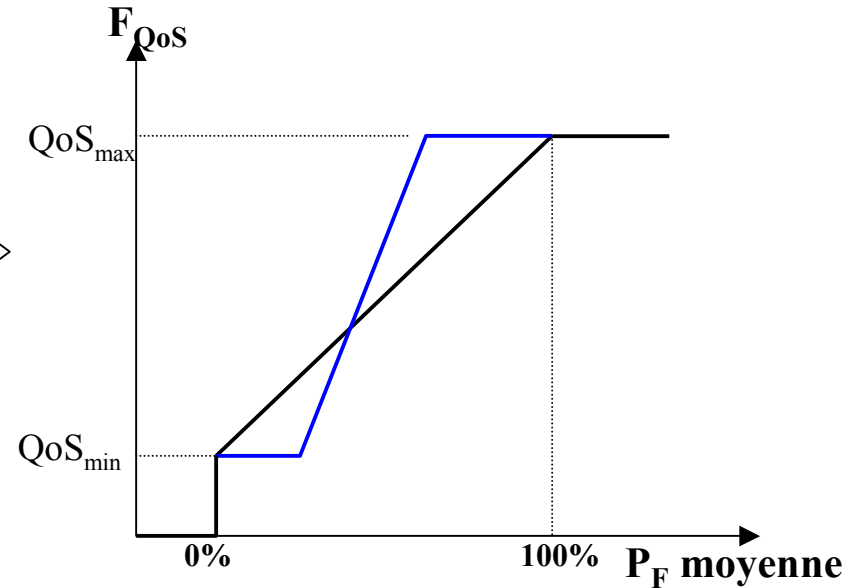
QoS = f (critères)

Exemple : C2 = Précision fournie, $P_F \uparrow \Rightarrow QoS \uparrow$; $P_F \downarrow \Rightarrow QoS \downarrow$

QoS = f(précision fournie)



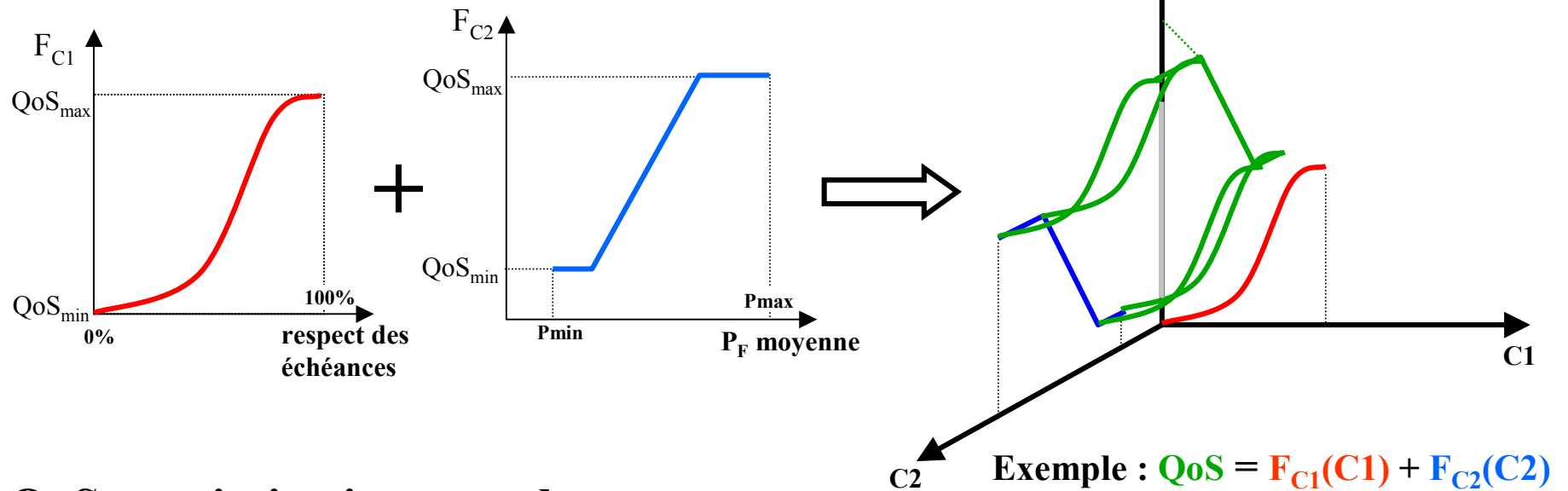
QoS = f(rapport entre P_F et $p_{min..pmax}$)



Caractéristique de QoS_D pour la précision spécifiée : $[P_{min..P_{max}}]$

Critère de satisfaction de la QoS pour la précision :
% moyen dans $[P_{min..P_{max}}]$

QoS = f (C1, C2, ...)



QoS : optimisation complexe ...

- requêtes futures inconnues
- optimisation dans un espace à n dimensions
- dépendance entre critères

☞ F_{C1} et F_{C2}

indépendantes : $QoS = F_{C1}(C1) \odot F_{C2}(C2)$

dépendantes : $QoS = F_{C1C2}(C1, C2)$

Satisfaction des contraintes :

1) respect des échéances

1.1) importance (en cas de faute)

2) critères

précision : [Pmin..Pmax]

qualité : valeur maximum

...

} dépendants

} indépendants

Calcul de la QoS :

$QoS_{é,i} = F_0(\text{échéance, importance})$

$QoS = QoS_{é,i} + \max (F_1(\text{critère}_1) + F_2(\text{critère}_2) + \dots)$

- ☞ F_0 considère tous les services, chacun avec sa durée minimale
- ☞ \max est évalué pour chaque service avant son exécution

Contrat implicite avec le Client

- Messages critiques : respect de l'échéance
- Messages non critiques
 - garantis : respect des échéances (sauf pour satisfaire un message critique)
 - moins importants : non exécutés en cas de surcharge

Degrés de liberté

- Messages garantis
- Règles d'exécution
 - min n/m exécutions, max n fautes sur m exécutions, ...
- Méthodes différentes
 - polymorphisme (plusieurs méthodes avec durée et propriétés différentes)
 - "task pair" (méthode normale + méthode d'exception)
 - valeurs estimées
 - ...

- **Objet Actif**

- objet traditionnel

plus

- contrôleur

- capacités d'exécution locale

- communication par messages

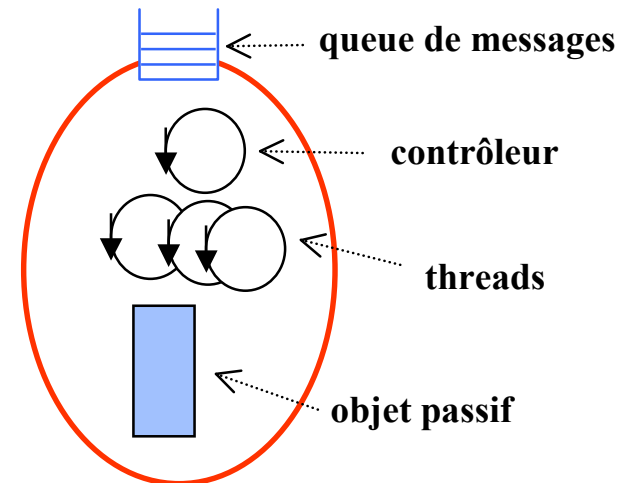
- queue de messages d'entrée

- décisions autonomes du traitement des messages

- **Concurrence**

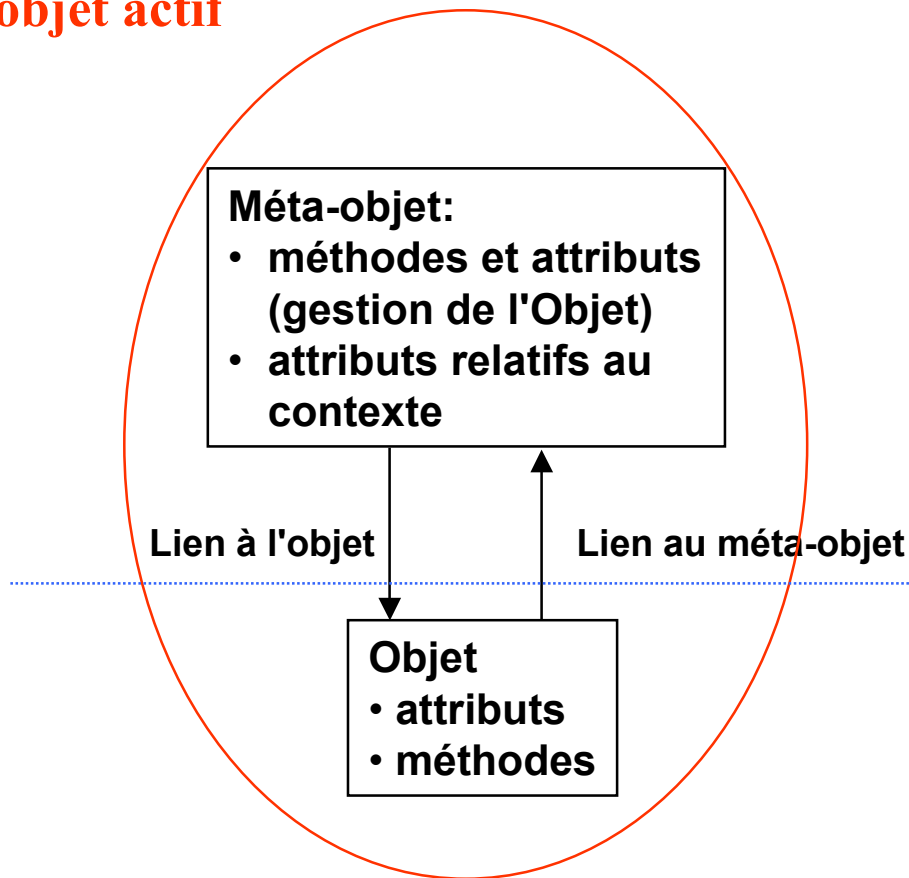
- messages asynchrones → inter-objet

- plusieurs threads par objet actif → intra-objet



ARTO = objet passif + contrôleur + queue de messages + thread(s)

objet actif



Niveau Méta : Méta Objets

☞ contexte d'exécution

META OBJET:

- ☞ Contrôle l'activité de l'Objet de Base
- ☞ Connait son Objet de Base
(état courant, méthodes, méthodes en exécution, messages en attente ...)
- ☞ Connait le contexte d'exécution
(charge du processeur, autres objets, historique ...)
- ☞ Applique les Critères de Décision

OBJET de BASE:

- ☞ Objet normal
- ☞ Solution "fonctionnelle" au problème de l'utilisateur

Niveau de Base : Objets de Base

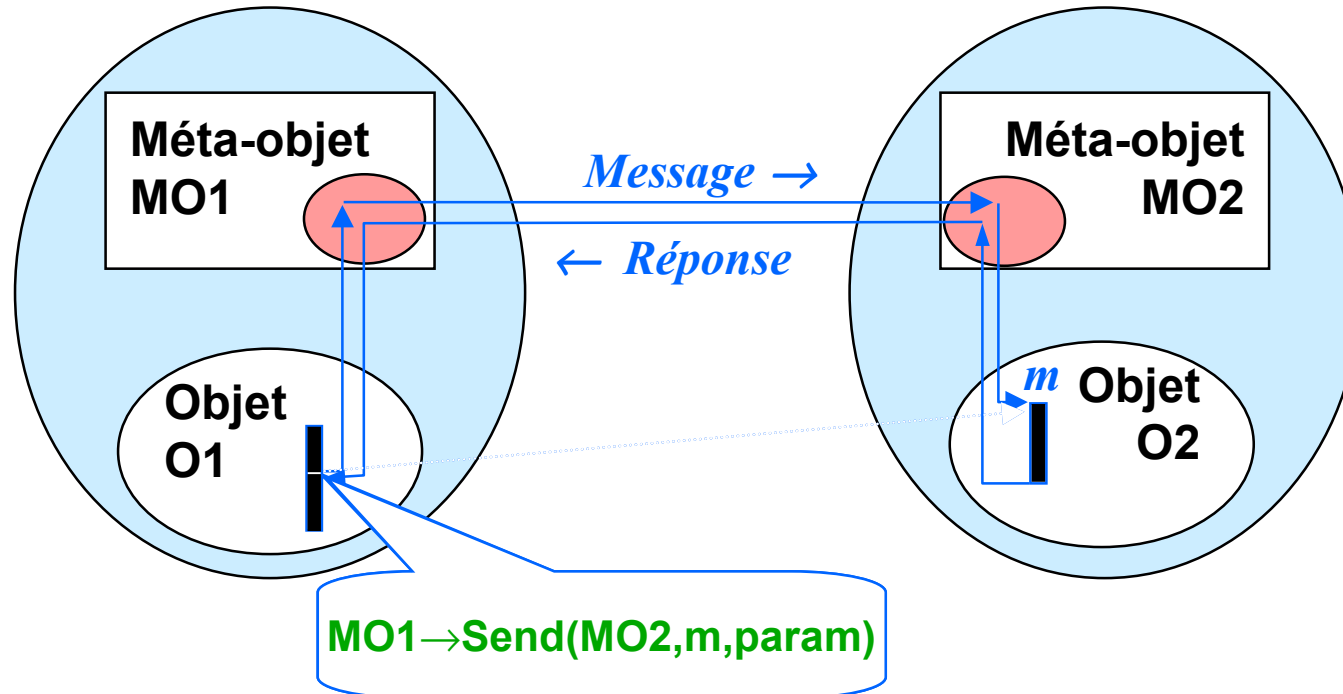
☞ domaine de l'application

Communication par messages

14

Objet Actif 1

Objet Actif 2



dans ARTO...

O2 → m (param) est traduit en **MO1 → Send(MO2, m, param)**


- 👉 messages contrôlés par les méta-objets
- 👉 objets isolés par les méta-objets

- **Message**

- **Spécifie le service demandé par le client**
- **Paramètres incluent :**
 - Échéance, période,..., importance, précision, ... garantie, délégation,... réplication, règles d'exécution...**
- **Le serveur décide comment réaliser le service**

- **Méthode**

- **Plusieurs méthodes pour un même service**
- **Chaque méthode a des caractéristiques particulières**
 - Validité selon état, compatibilité entre méthodes, durée, précision, ressources...**
 - Temps d'exécution : Pire cas, Optimiste, Minimum, autres...**

 **Un «ensemble de méthodes» est associé à chaque message**

() Approche Objet et Applications Temps Réel

16

- **Approche Objet :**

- ☞ qualités logicielles améliorées (réutilisabilité, portabilité...)
- ☞ entités autonomes, données encapsulées...

⇒ **nature DISTRIBUEE**

- **Applications Temps Réel :**

- ☞ contraintes temporelles, ressources partagées et limitées...
- ☞ gestion de ressources et prise de décision rationnelle

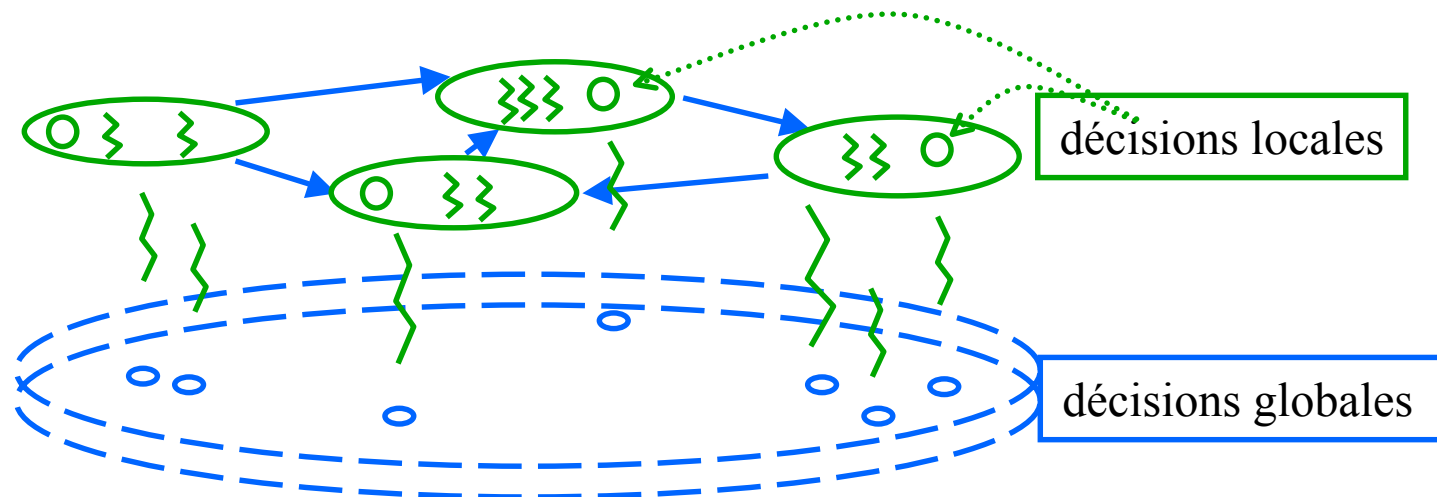
⇒ **nature CENTRALISEE**

OO (distribué) + RT (centralisé) => conflits :-)

ARTO : décisions à deux niveaux

17

- **décisions locales** ⇒ **AUTONOMIE DES OBJETS**
- **décisions globales** ⇒ **GESTION RATIONNELLE DES RESSOURCES PARTAGEES**

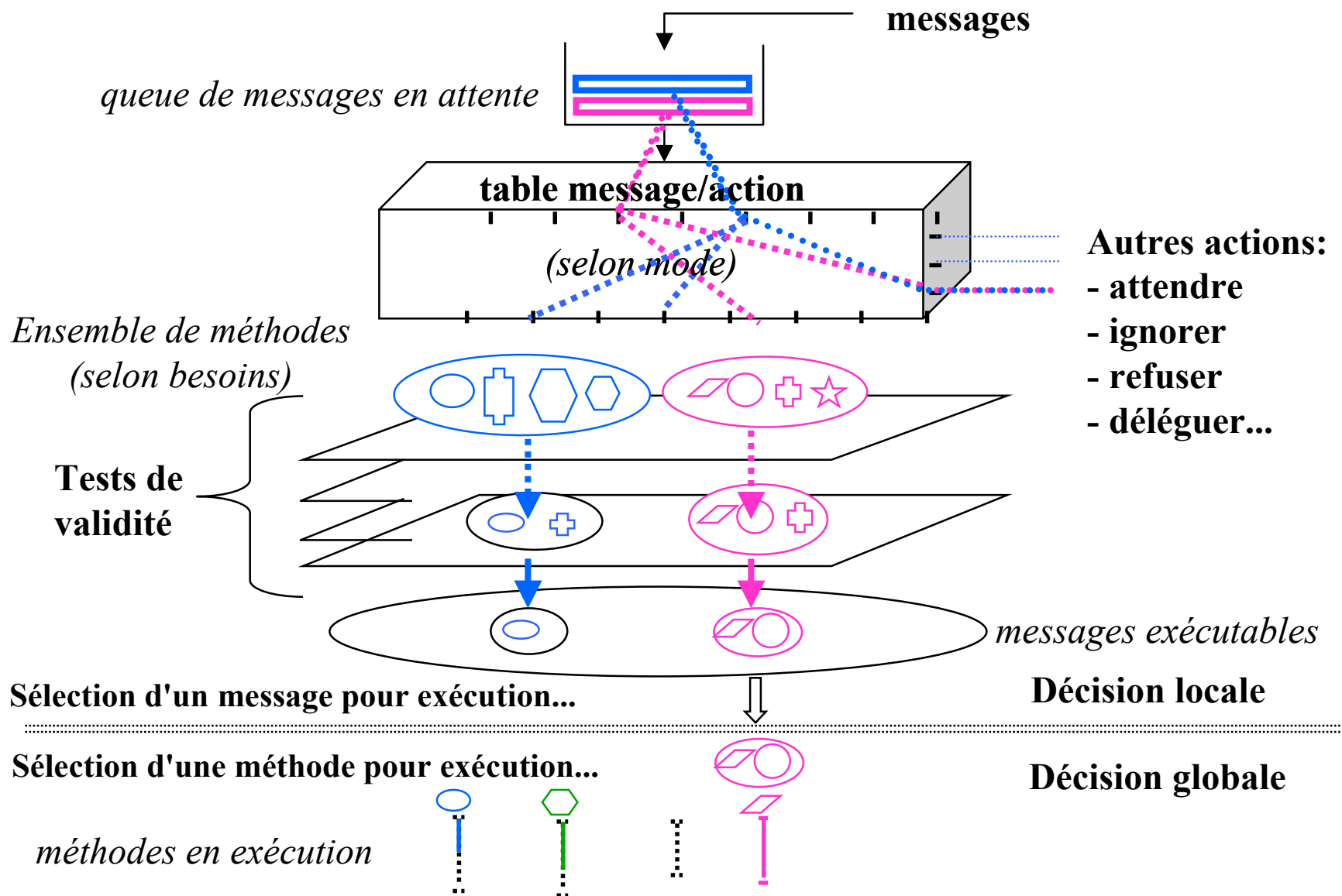


Le contrôleur

- **Contrôle l'activité de l'objet**
 - **Réservation, analyse des messages, envoi de messages...**
- **Réagit aux événements qui affectent l'objet**
 - **Arrivé de message, fin d'exécution de tâche, exceptions (dépassement d'échéance, ...)...**
- **Traite les messages de l'objet**

- **Exécute les décisions globales**
 - **Réalise les décisions globales en coopération avec tous les autres contrôleurs**

Traitement de messages



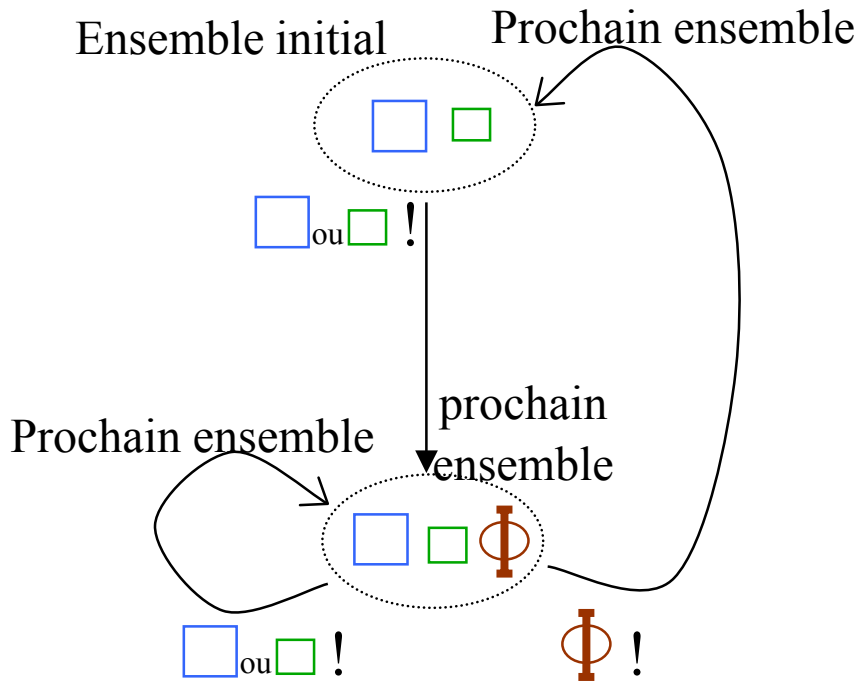
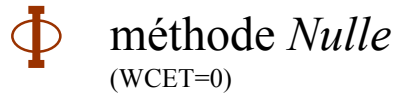
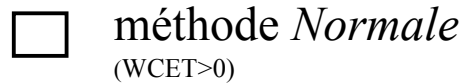
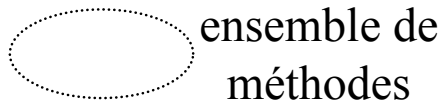
Objectif : “meilleur effort” pour satisfaire les contraintes

- **Construction de l'ensemble de méthodes**
 - polymorphisme, "Task Pair", Calcul Imprécis, valeur estimée, règles d'exécution (n/m, ...)...
- **Élimination de méthodes non exécutables dans l'état courant**
- **Sélection d'un message pour exécution**
 - critère : FIFO, temps, importance, valeur, durée, règles, ...
 - ☞ message (ensemble de méthodes) transmis à l'ordonnanceur global
- **Traitement des cas particuliers**
 - délégation de messages, messages périodiques...

Exemple d'adaptation locale

- Règle : “*exécution d'une méthode au moins 1 fois sur 2*”
 - composition de l'ensemble de méthodes :

Symboles:



Objectifs : a) minimiser les fautes temporelles

b) privilégier les tâches plus importantes

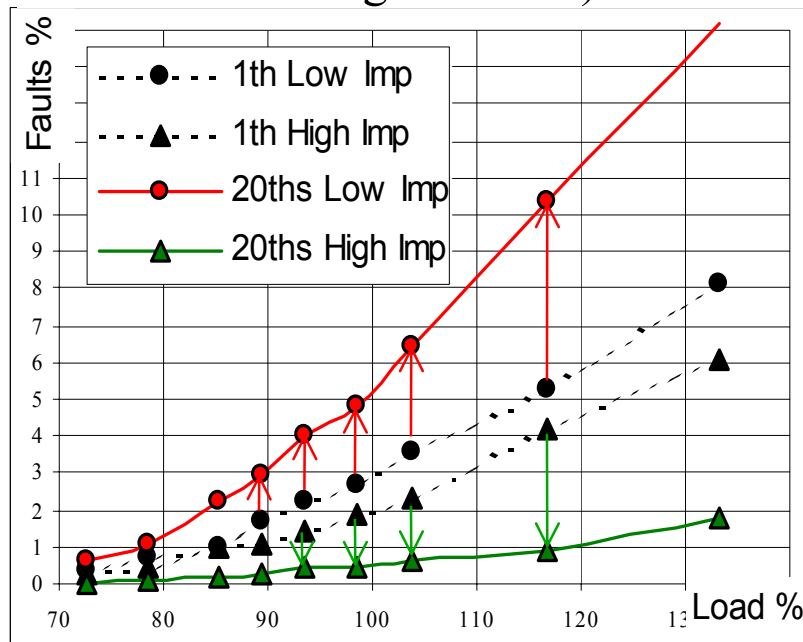
- **Ordonnanceur Global : EDF**
 - simple, dynamique, rapide, optimal (si ordonnançable)
 - traitement : général au niveau global, spécifique au niveau local
- **Caractéristiques:**
 - planning construit avec la méthode la plus rapide
 - méthode à exécuter choisie au dernier moment (*max* qualité possible au sens des critères)
 - surcharge
 - choix selon l'importance
 - tâches non acceptées → dans une queue
 - exception à l'échéance → métaobjet

Effet du nombre de threads par Objet

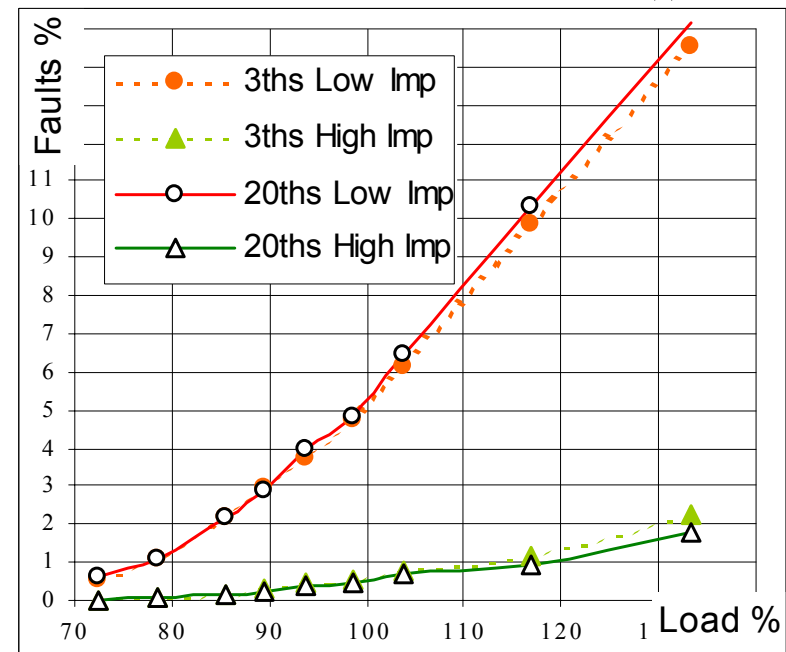
Effets dans la distribution des fautes ?

Combien de threads par O ?

1 thread : pas de grande différence :(
20 ths: bonne dégradation :)



20 threads est optimal :)
Mais... 3 threads suffisent ! :))



Contraintes + Multiples réalisations de services

↓ **Adaptation**

Augmentation de la QoS fournie

ARTO

- ☞ **Réduction des fautes temporelles & Meilleure qualité possible**
- ☞ **Dégradation contrôlée en cas de surcharge**
- ☞ **Préservation des qualités logicielles et autonomie des objets**

- **Prototype**
 - **messagerie, états, concurrence intra-objet, ordonnancement, QoS (échéance, importance, précision), alarmes temporelles...**
 - **temps d'exécution de messages nuls comparable aux outils du marché**