

Journées

« Systèmes à composants adaptables et extensibles »

**Essayer et Adapter :
une approche pour améliorer la confiance dans
l'usage d'un composant**

**Pham Thi Xuan Loc^(*), Ph. Mauran et G. Padiou
Institut de Recherche en Informatique de Toulouse**

(*) Université de Cantho, Vietnam

{mauran,padiou,ptxl}@enseeiht.fr

17-18 octobre 2002

Plan

- Problématique
- Sûreté d'usage des composants
- Spécification des règles d'usage
- Réalisation d'un service de sûreté d'usage
- Conclusion et perspectives

Problématique (1)

Les usagers se distinguent par :

- L'utilisation des méthodes (rôles)
- Les attentes par rapport au contexte d'utilisation (performances, sécurité,...)
- Les attentes par rapport au service lui-même (propriétés de sûreté)

Vision centrée « usager »

- • Mieux spécifier l'usage du composant par l'utilisateur (profil)
- • Augmenter la confiance de l'utilisateur

Problématique (2)

Objectifs

- contrôler dynamiquement le comportement du composant,
(\mapsto notion de bac à sable)
- adapter le service assuré par le composant, en associant un traitement aux propriétés définies par le profil,
(\mapsto mécanisme classique d'exception)
- permettre l'évolution dynamique du service en lui attachant systématiquement un profil donné (\mapsto mécanisme d'adaptation).

Sûreté d'usage des composants (1)

Interactions usager-composant : profil

- Propriétés d'usage **implicites** observées(ables) par le composant
- Propriétés d'usage **explicites** fournies par l'usager

Types de propriétés du profil portant sur une trace :

interactions	1 composant	tous les composants
1 usager	locale	globale à 1 usager
tous les usagers	globale à 1 composant	globale

Sûreté d'usage des composants (2)

Le contrôle de l'usage des composants

- Vérifier les propriétés de sûreté par invariant :
 - Habituel : mauvais usage \Rightarrow exceptions ;
 - Extension : usage plus restrictif \Rightarrow contrôle explicites liés à un usage(r) particulier.
- Instrumentation du composant pour contrôler de nouvelles assertions

Exemple (1)

```
module Conversion {
  /* Service de change proposé par les institutions */
  interface Changeur { ...
    float Combien(in string devVal, in float val ) ;
    float Changer ( in string devVal, in float val ) ;
  };
  /* Serveur central enregistrant les changes disponibles */
  typedef sequence<Changeur> Changeurs;
  interface ChangeDeMonnaies { ...
    void Enregistrer (in Changeur unChangeur);
    Changeurs Disponibles(in string Dev1, in string Dev2) ;
  };
};
```

Exemple (2)

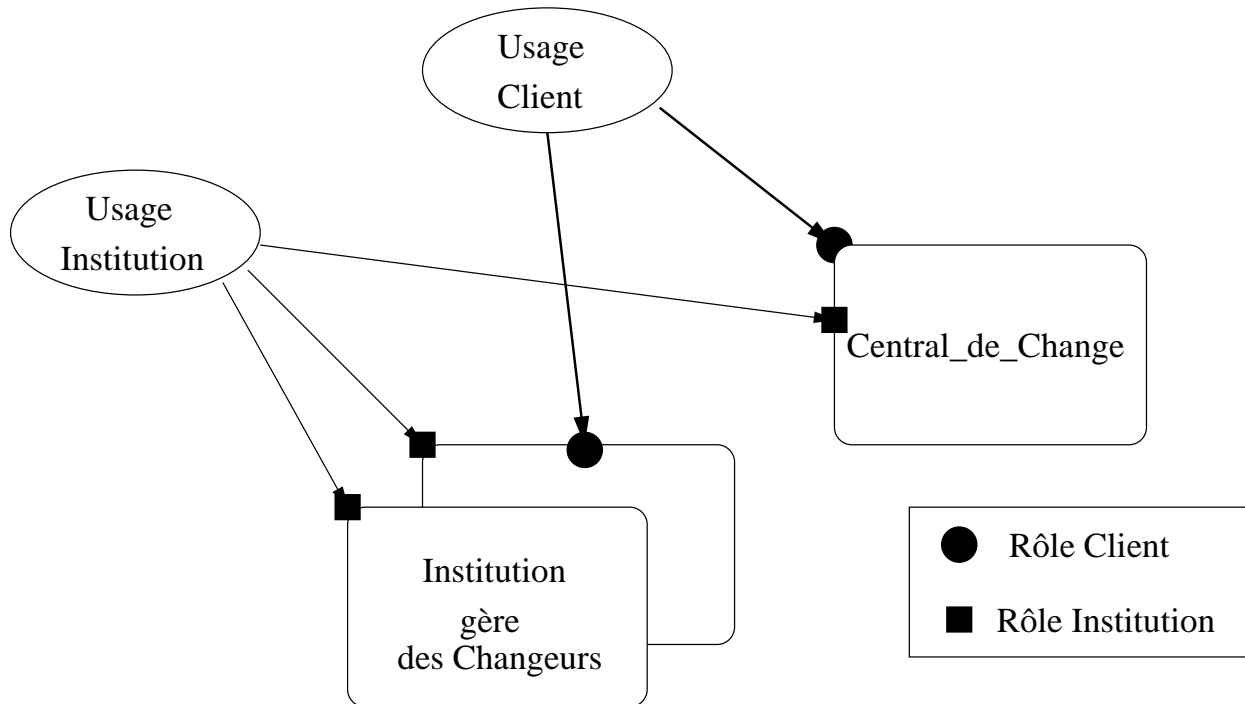


FIG. 1 – *Architecture logicielle*

Spécification des règles d'usage: Profil (1)

Contraintes d'usage « Institution »

- Enregistrement de tout changeur auprès du serveur central ;
- Périodicité des variations de taux ;
- Existence d'un nombre minimum de possibilités de change ;
- Contrôle des écarts de taux ;
- ...

Spécification des règles d'usage: Profil (2)

- Utilisation d'un langage déclaratif

Contenu

- composants référencés
- contraintes d'ordonnancement des appels
- contraintes de QoS (temporelles, performances) : temps de réponse, disponibilité
- contraintes sur les données : fréquence de mise-à-jour,...
- préférences : traitants liés à des conditions particulières d'exécution

Spécification des règles d'usage: Profil (3)

```
Profil Institution = {  
  Use Central_de_Change Central ;  
  Object c : Changeur (d1,d2) ∈ self ;  
  Scenario = { ...  
    once Central.Enregistrer(c) ;;  
    ...  
  }  
  Constraints = { ... rate(c.setTaux)<2/h ;; }  
  Preferences = { a ∈ Central.Disponibles(d1,d2) :  
    when |(c.Taux - a.Taux)/c.Taux| > 10%  
      { ... code d'ajustement du taux ... };;  
  }  
}
```

Réalisation d'un service de sûreté d'usage

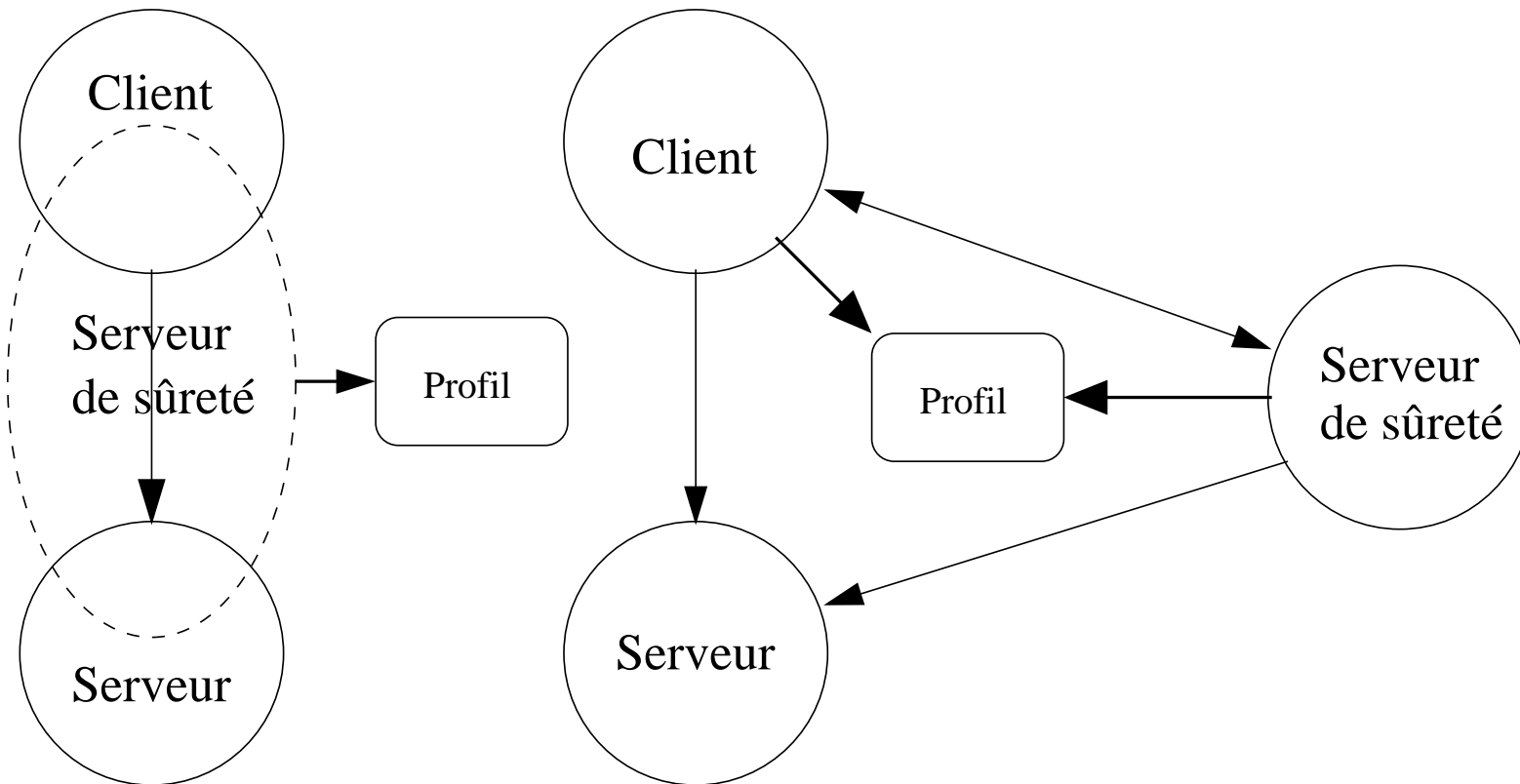


FIG. 2 – *Instrumentation des composants (Variantes)*

Conclusion

Travail en cours

- Instrumentation des composants ;
- Spécification des profils ;
- Spécification du service de sûreté (confiance) ;
- Mise en œuvre via CORBA ou J2EE ;

Perspective Formaliser cette notion de profil (utilisation du MOF)