

# Modélisation de Services Web

– sémantique de service et client adapté –

RGE

10 mars 2005

Sylvain Rampacek

[sylvain.rampacek@univ-reims.fr](mailto:sylvain.rampacek@univ-reims.fr)

Thèse encadrée par :

Serge Haddad – LAMSADE

Patrice Moreaux – CReSTIC

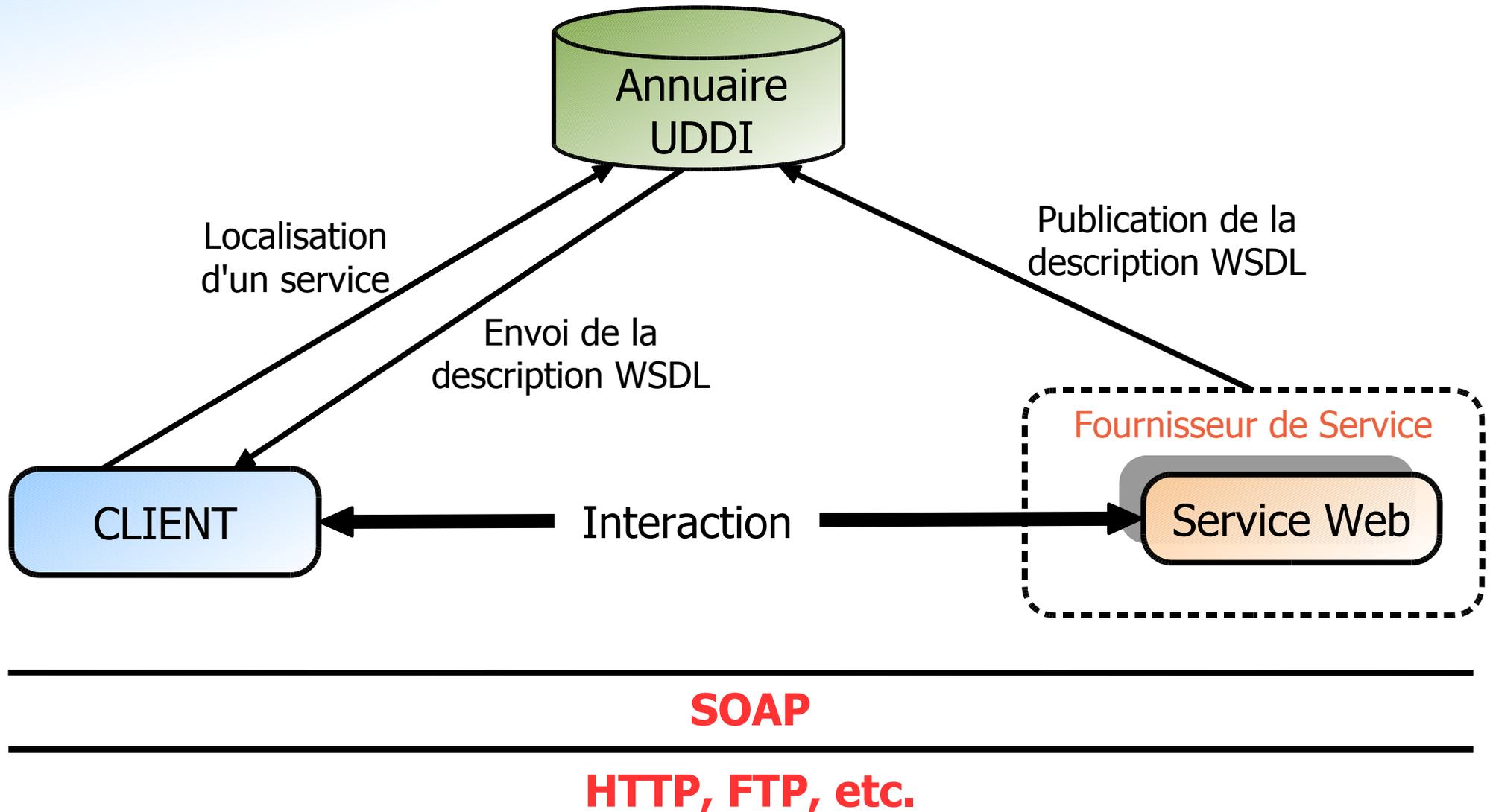
# Plan

- ◆ Services Web
- ◆ Description de la Plateforme
- ◆ Sémantique des Services Web - Xlang
- ◆ Synthèse du client
  - ◆ TIOTS
  - ◆ Automates Temporisés

# Contexte des Services Web

- ◆ Système réparti (un service  $\neq$  un serveur)
- ◆ Interopérabilité maximale (XML, SOAP, WSDL, ...)
- ◆ Gestion de l'hétérogénéité :
  - ◆ Couche supplémentaire (conserve la couche métier)
  - ◆ Évolution des systèmes répartis à objets
  - ◆ Accès orienté service (vers le SOA)

# Architecture des Services Web

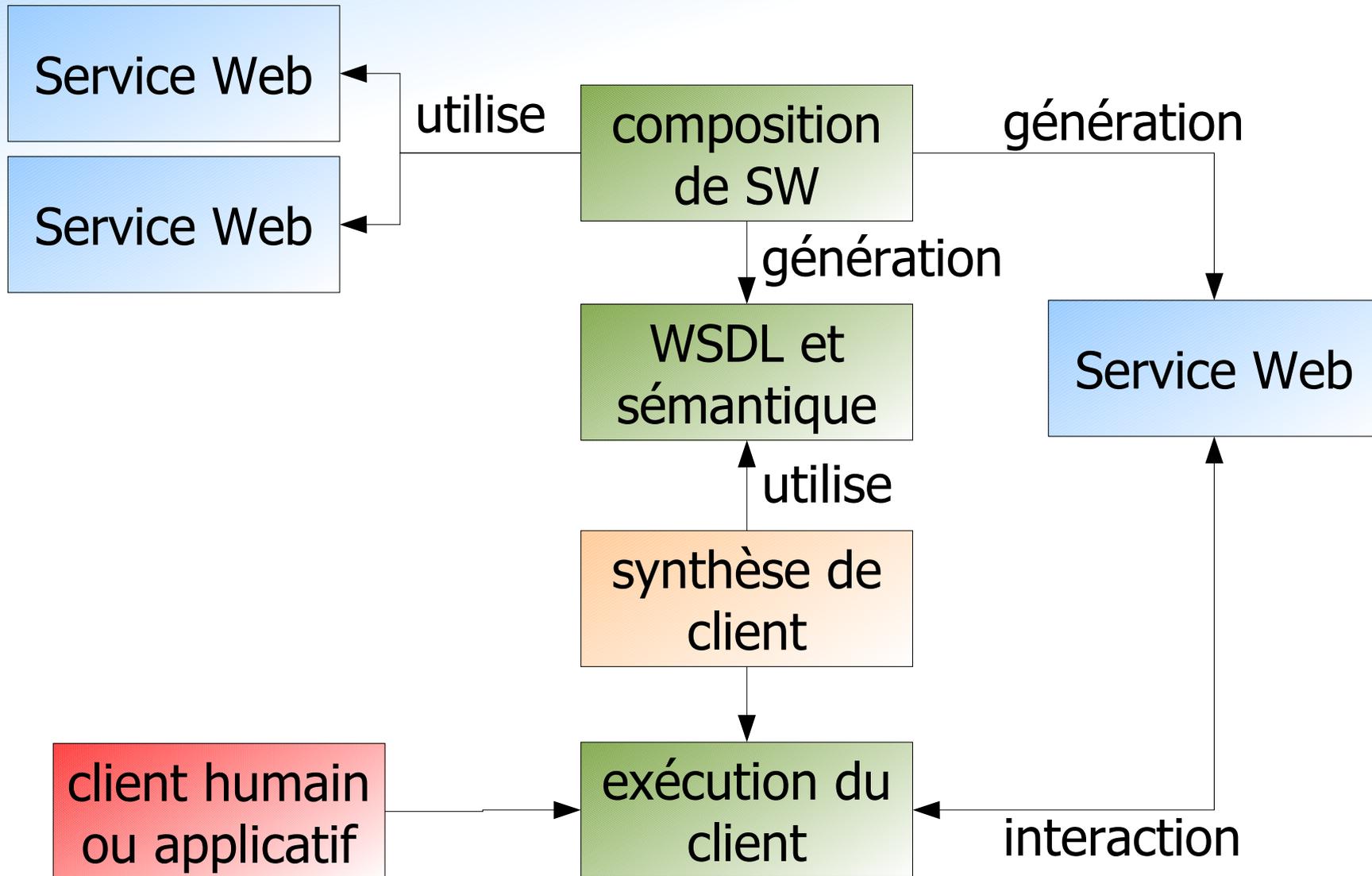


# Architectures Orientées Services ?

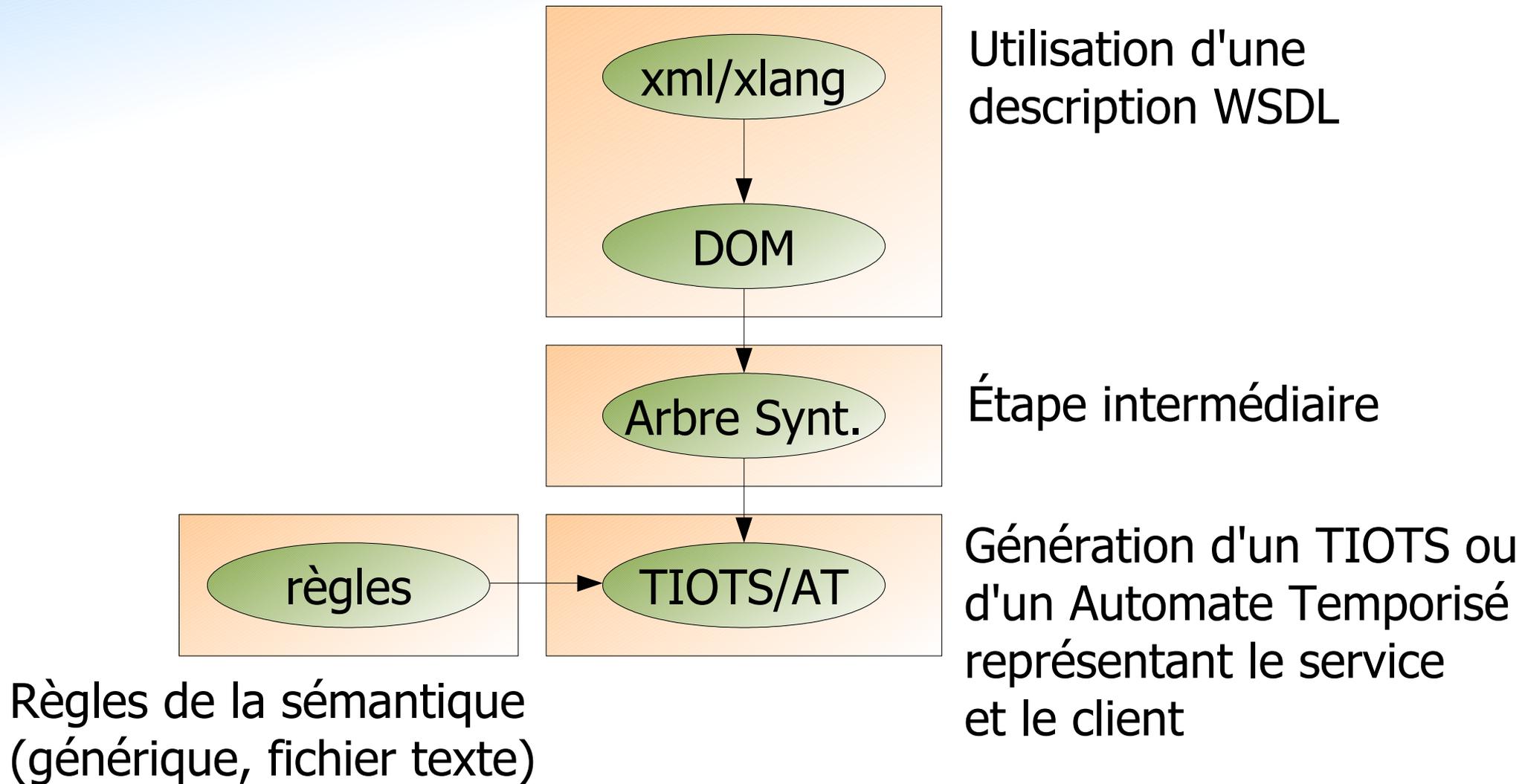
- ◆ Extension de la description WSDL
  - ◆ XLANG, BPML, BPEL4WS...
- ◆ Gestion automatique des flux
- ◆ Description de la logique métier du Service Web
- ◆ Principal problème : orchestration

Relation d'interaction entre  
les clients et les services

# Plateforme



# Synthèse de client



# WSDL et Langages Sémantiques

- ◆ XLANG (Microsoft) :
  - ◆ Éléments de bases : action, while, switch, context, ...
    - ◆ Conditions
    - ◆ Boucles
    - ◆ Gestion du temps, des exceptions
- ◆ Évolution en cours vers BPEL4WS
  - ◆ Fusion de XLANG et BPML
  - ◆ Principaux auteurs : BEA, IBM, Microsoft

# Xlang

- ◆ Formalisation de XLANG à l'aide des APT [HMMR04a]
  - ◆ Ajoute de la précision pour la sémantique de XLANG
  - ◆ Optique de la généralité (passage à BPEL4WS)
- ◆ Plusieurs types d'actions :
  - ◆ Envoi/Réception d'un message
  - ◆ Écoulement de temps (en temps discret)
  - ◆ Action interne
  - ◆ Terminaison

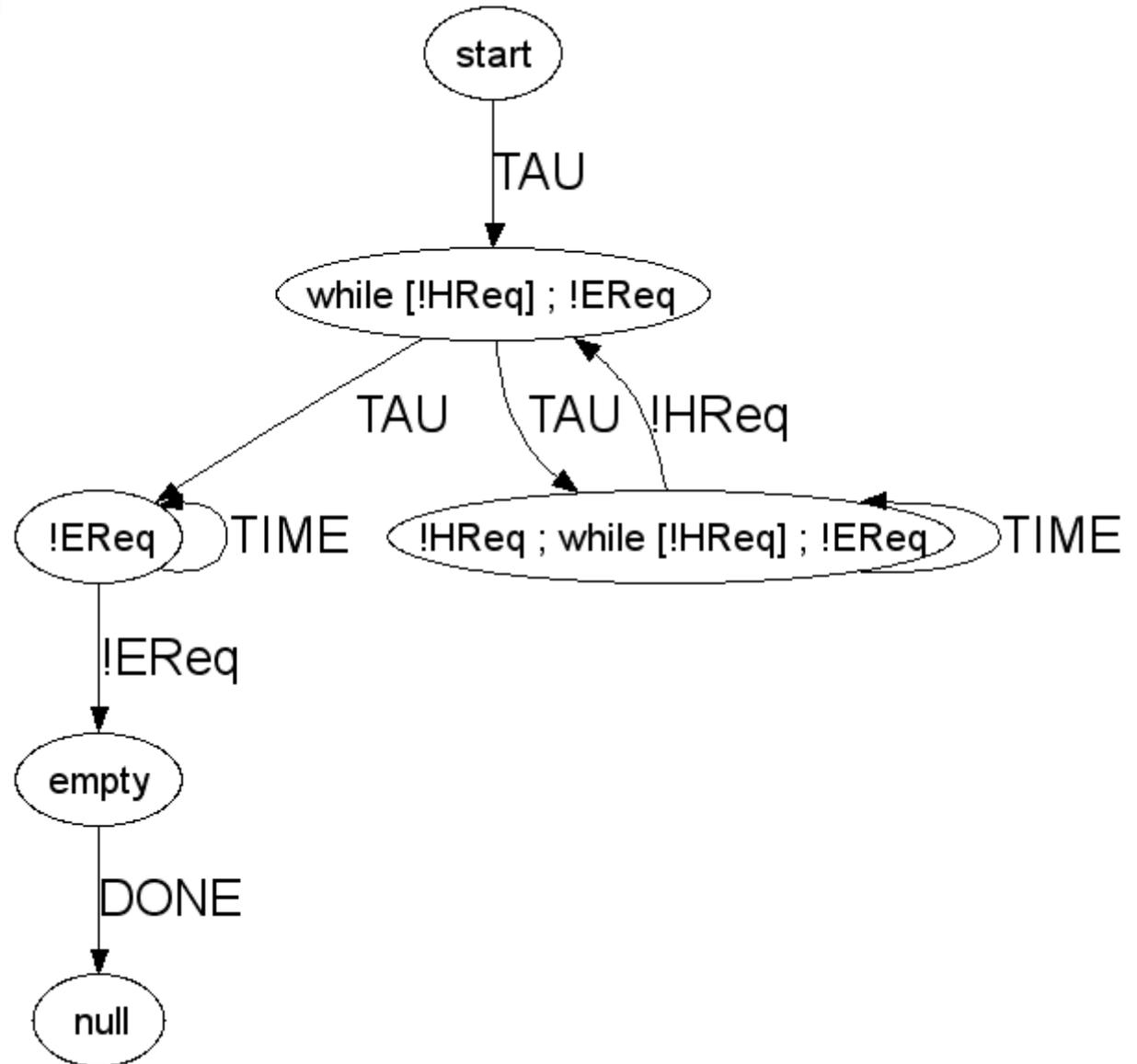
# Synthèse TIOTS générique

- ◆ Besoin d'une synthèse « générique » :
  - ◆ Domaine très mouvant
  - ◆ Ne pas se lier à un seul langage
  - ◆ Adaptabilité de la sémantique
- ◆ Fichier de règles :
  - ◆ Chaque élément de base est décrit par :
    - ◆ Des gardes
    - ◆ Des transitions résultats (et l'état dérivant)
    - ◆ Règles de ré-écritures (pour permettre de simplifier un état)

# Exemple

## TIOTS – Partie Service

Processus :  
while {!HReq} ; !EReq



# Génération du client

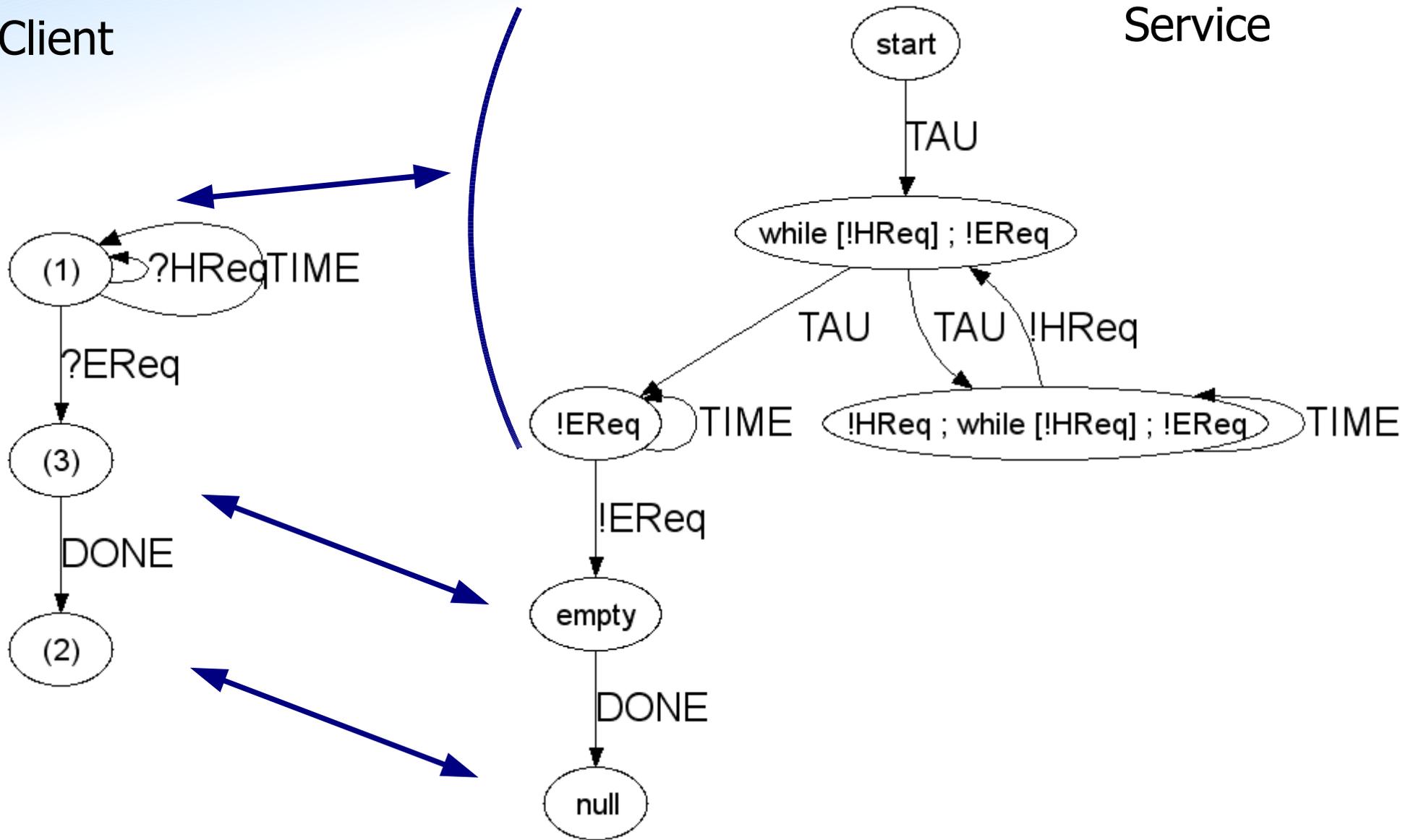
- ◆ Une fois le TIOTS de la partie service généré, on utilise la relation d'interaction pour générer la partie cliente
  - ◆ Relation d'interaction adaptée entre un client et un service :
    - ◆ Si un message est envoyé par le service, alors le client doit pouvoir recevoir un message
    - ◆ Si le service attend un message, alors le client doit pouvoir l'envoyer
    - ◆ Si le temps s'écoule sur le serveur, alors il doit aussi s'écouler sur le client

# Exemple

## TIOTS – Partie Cliente

Client

Service



# Temps discret : le problème

- ◆ L'écoulement de temps est symbolisé par une transition dans le TIOTS
- ◆ Dans le cas de Services Web complexes (comportant des temps d'exécution maximaux imbriqués), le nombre d'états explose !

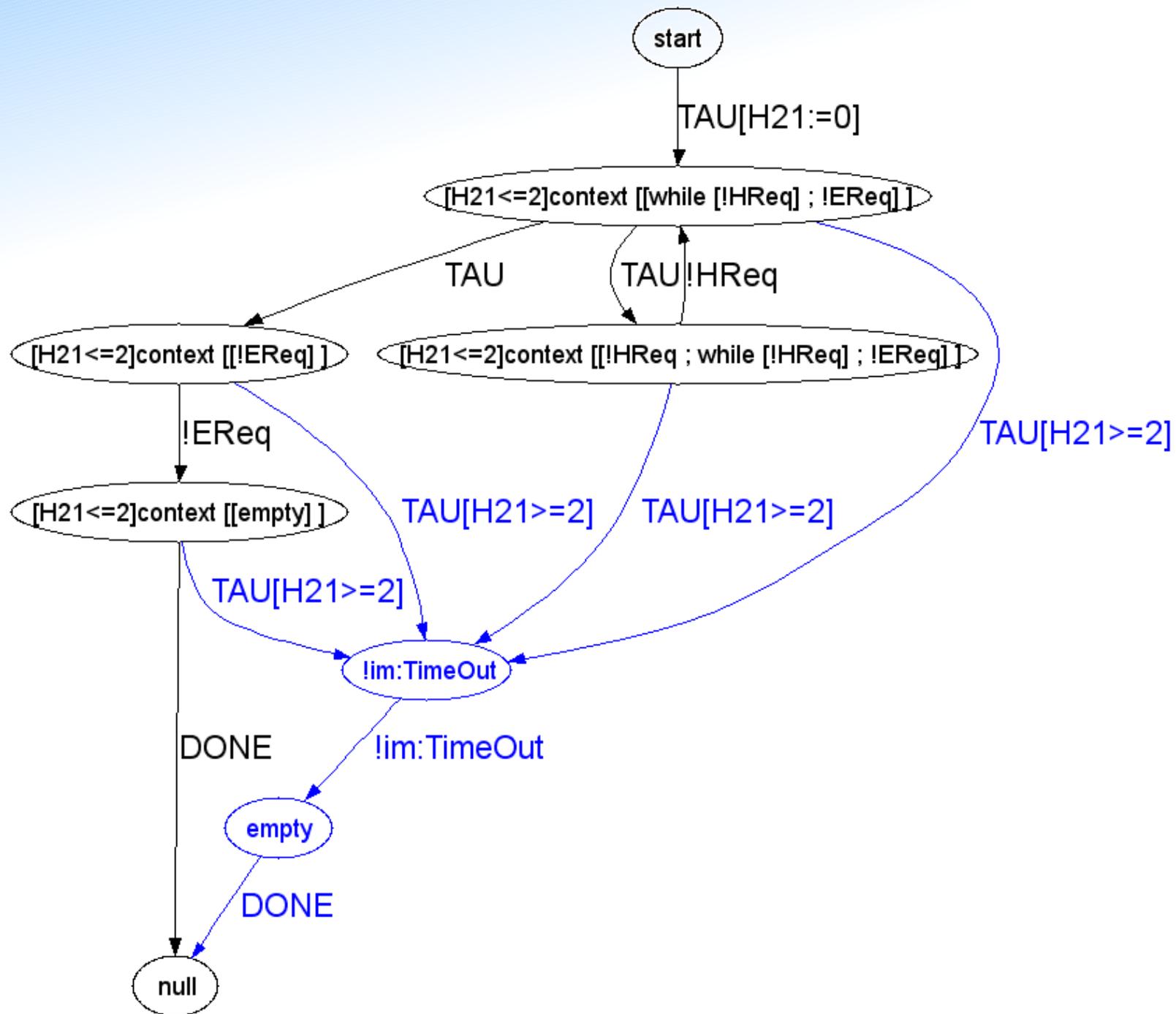
=> Évolution vers les automates temporisés

# Deuxième modélisation

– temps dense –

- ◆ Passage en automates temporisés [HMMR04b]
  - ◆ Adaptation de la sémantique :
    - ◆ Suppression du passage explicite du temps
    - ◆ Ajout de gardes sur les transitions
    - ◆ Ajout d'invariant dans les états
  - ◆ Avantage : nombre d'états beaucoup plus petit !
  - ◆ Inconvénient : aspect non déterministe des automates temporisés

# Exemple – Automate Temporisé – Partie Service



# Conclusion

- ◆ Outil de synthèse d'un client générique pour Service Web
- ◆ Relation d'interaction (en temps discret)
- ◆ Développement en cours : outil de composition de Services Web
- ◆ Technologies très mouvantes
  - ◆ Normalisées par le W3C

# Perspectives

- ◆ Écriture d'un nouvel algorithme d'interaction
  - ◆ Tenant compte de l'aspect non déterministe d'un automate temporisé
  - ◆ Utilisation de certaines classes d'automates temporisés
- ◆ Invocation d'un service à partir de l'automate client
  - ◆ Dans le but de la composition
- ◆ Obtenir une plateforme permettant l'orchestration et la validation d'applications orientées services.

# Bibliographie

- ◆ [HMMR04a] S. Haddad, T. Melliti, P. Moreaux, and S. Rampacek. A dense time semantics for Web services specifications languages (ICTTA'04).
- ◆ [HMMR04b] S. Haddad, T. Melliti, P. Moreaux, and S. Rampacek. Modelling web services interoperability (ICEIS04).