Master II Informatique
Parcours : Génie logiciel

Évolution dirigée par la qualité des architectures à base de services

Chouki.TIBERMACINE@lirmm.fr
Maître de conférences en informatique

http://www.lirmm.fr/~tibermacin/ens/hmin306/

Plan du cours

- **Partie 1 :** Évolution des architectures à composants
- **Partie 2 :** Évolution des architectures à services

Contenu de cette partie

- Introduction aux services et problématique
- Approche proposée
- Approche proposée par l'exemple
- Expérimentation de l'approche
- Conclusion

Contenu de cette partie

- Introduction aux services et problématique
- Approche proposée
- Approche proposée par l'exemple
- Expérimentation de l'approche
- Conclusion

Introduction aux services

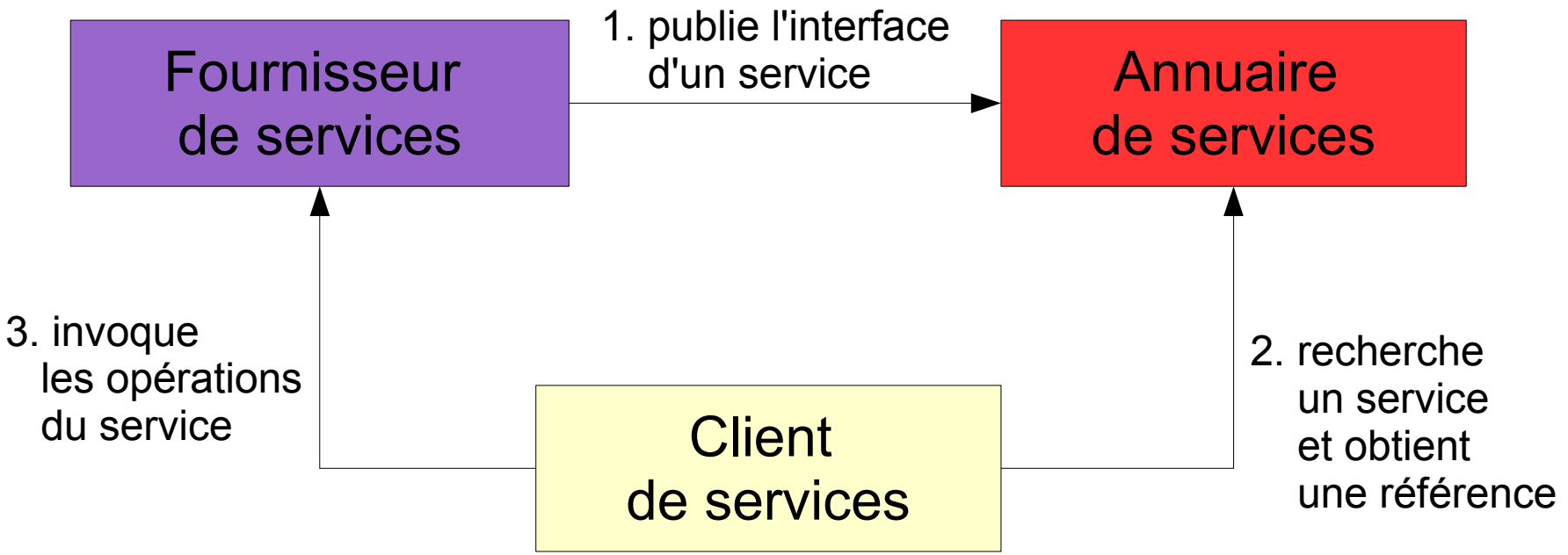
- Un service = ensemble d'opérations (fonctions) :
  1. enregistré par son fournisseur auprès d'un annuaire
  2. pouvant être appelé à distance par des programmes clients

- Dans un même service, pas de conservation d'état entre les appels des opérations avec un même client (*stateless components*) → stateful services existent mais rares (pour des raisons de performances)

- Une interface de service : indépendante d'un langage de prog. ou d'une plate-forme particulière (peut-être utilisée par n'importe quel programme client : écrit en Java, C, PHP, ... sous Linux, Mac OS, Windows, ...)

- Peut être publié sur le Web : on parle alors de service Web

Parties prenantes dans une architecture à services



Interfaces et protocoles standards

- Les interfaces des services, publiées dans les annuaires, sont définies avec des langages standards : WSDL pour les services Web, par exemple
- Les protocoles de communication (format d'échange de messages/paramètres, par exemple) entre les clients et ces services sont aussi standardisés : SOAP et HTTP, par exemple
- L'implémentation de ces services peut être faite avec n'importe quel langage supporté par les serveurs d'applications des fournisseurs de services : des classes Java, des EJB, des programmes C, PHP, Python, ...

Exemple de service Web

- Service Web pour l'obtention du nom d'une ville à partir d'un code postal et inversement (+ fuseau horaire, ...) : http://www.ripedev.com/webservices/ZipCode.asmx?WSDL

```
– <wsdl:portType name="ZipCodeSoap">
  – <wsdl:operation name="ZipCodeToCityState">
    – <wsdl:documentation>
        Returns a list of City+State for a supplied zip code.
      </wsdl:documentation>
      <wsdl:input message="tns:ZipCodeToCityStateSoapIn"/>
      <wsdl:output message="tns:ZipCodeToCityStateSoapOut"/>
    </wsdl:operation>
  – <wsdl:operation name="CityToZipCode">
    – <wsdl:documentation>
        Returns a list of City+State for a supplied zip code.
      </wsdl:documentation>
      <wsdl:input message="tns:CityToZipCodeSoapIn"/>
      <wsdl:output message="tns:CityToZipCodeSoapOut"/>
    </wsdl:operation>
```

Compositions de services

- Les compositions de services sont des applications qui incluent des appels vers les opérations de différents services

- Deux sortes de modèles de compositions de services :
  - Orchestrations de services : applications qui invoquent les opérations de différents services et centralisent les données et les traitements sur ceux-ci au sein d'un même processus métier
  - Chorégraphies de services : collaborations entre services qui s'appellent mutuellement

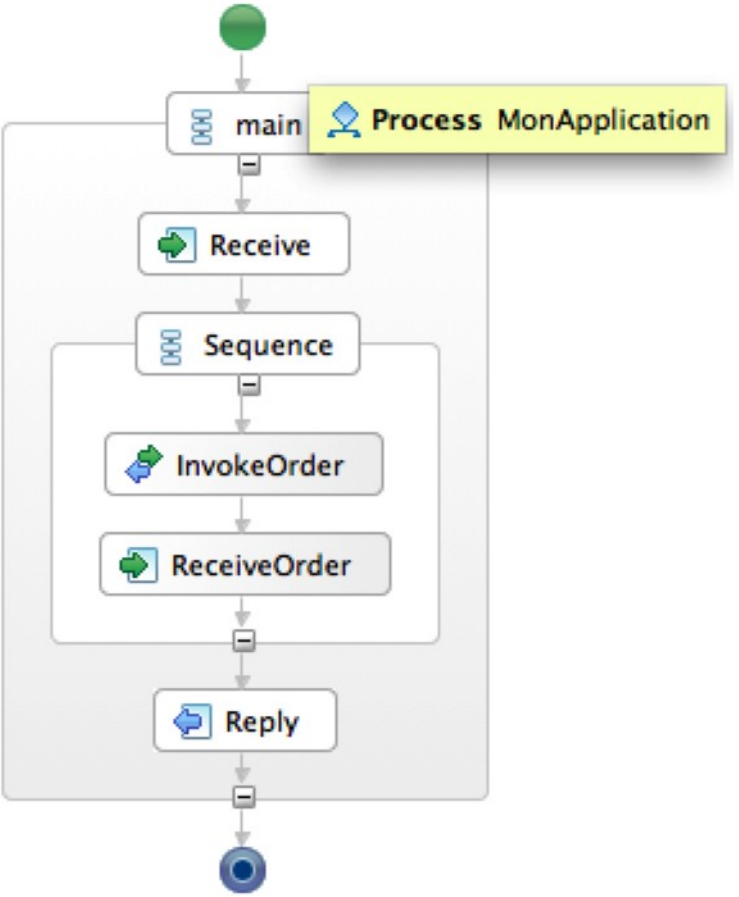
Orchestrations de services Web : processus BPEL

- Un langage concret permettant de définir des orchestrations de services est BPEL (Business Process Execution Language)

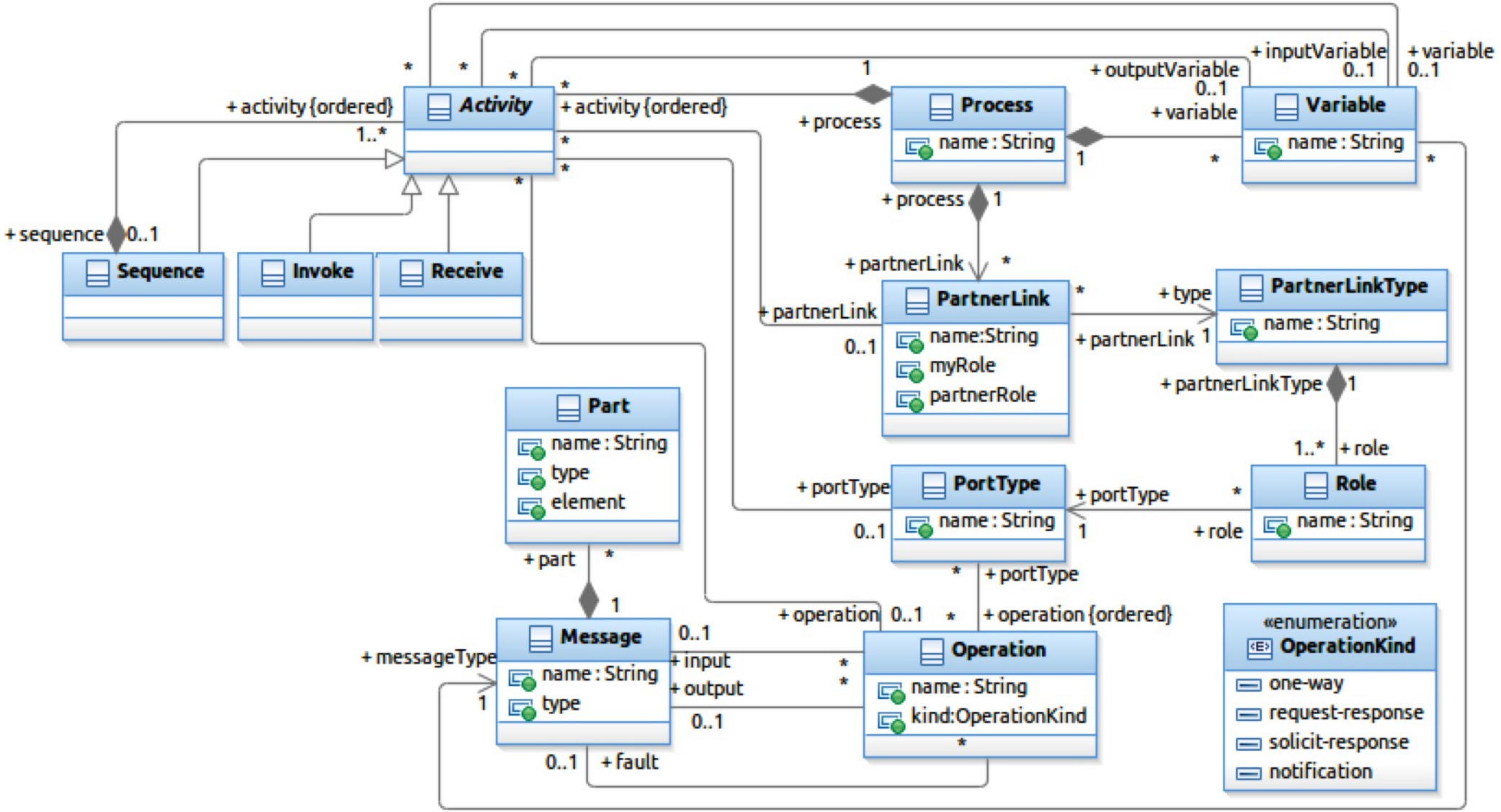
- BPEL : un langage pour la modélisation et l'exécution de processus métier composé d'activités (affectations, structures conditionnelles et itératives définies de façon simple pour les non-spécialistes, des invocations, des réceptions de résultats, …)
  BPEL = langage de workflow

- Ces mêmes processus BPEL peuvent être publiés en tant que services

Exemple simple de processus BPEL



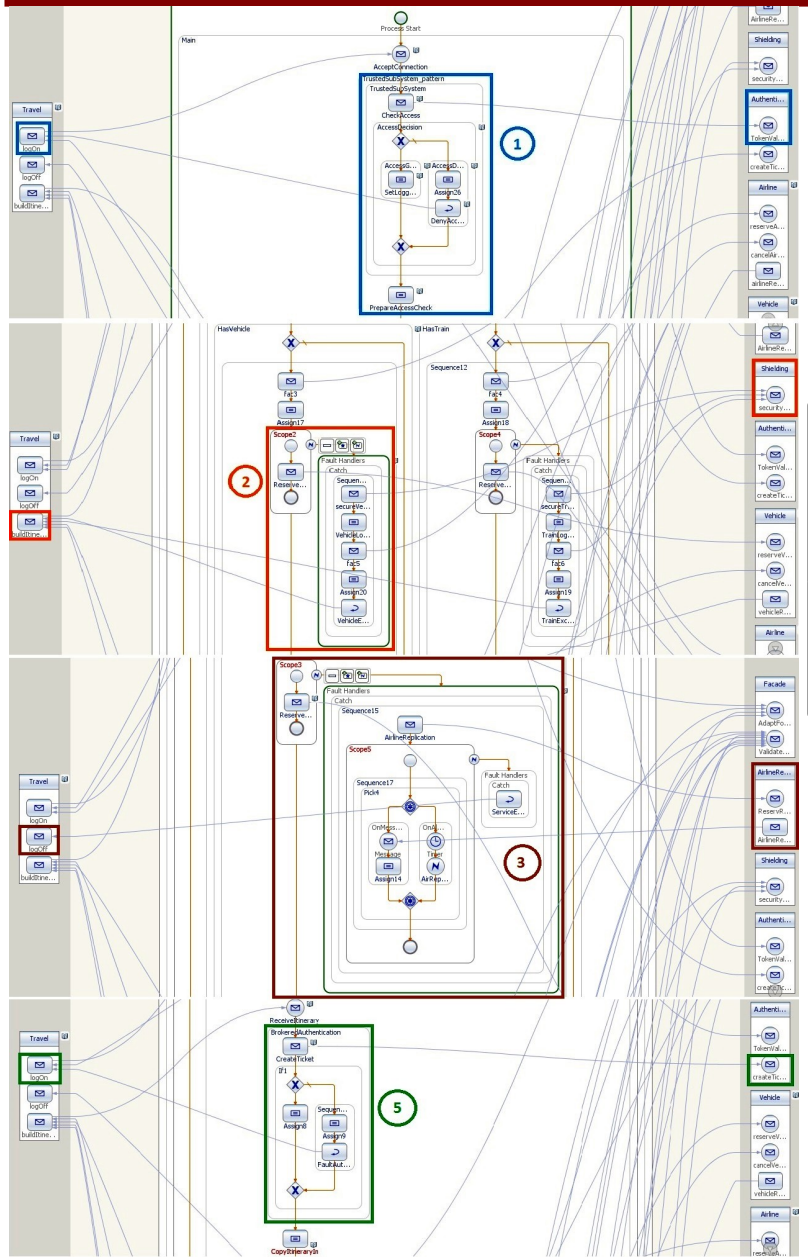
Extrait des méta-modèles de WSDL et BPEL



Problématique

- Comme toute architecture logicielle, une orchestration de services (implémentation possible d'une architecture à services : SOA) doit nécessairement évoluer
- Comme toute architecture logicielle, la conception d'une orchestration implique la prise de décisions architecturales (patrons SOA, par exemple) garantissant des attributs qualité
- Mettre en place ces décisions est une tâche difficile et coûteuse en temps

Exemple de processus BPEL intégrant des patrons

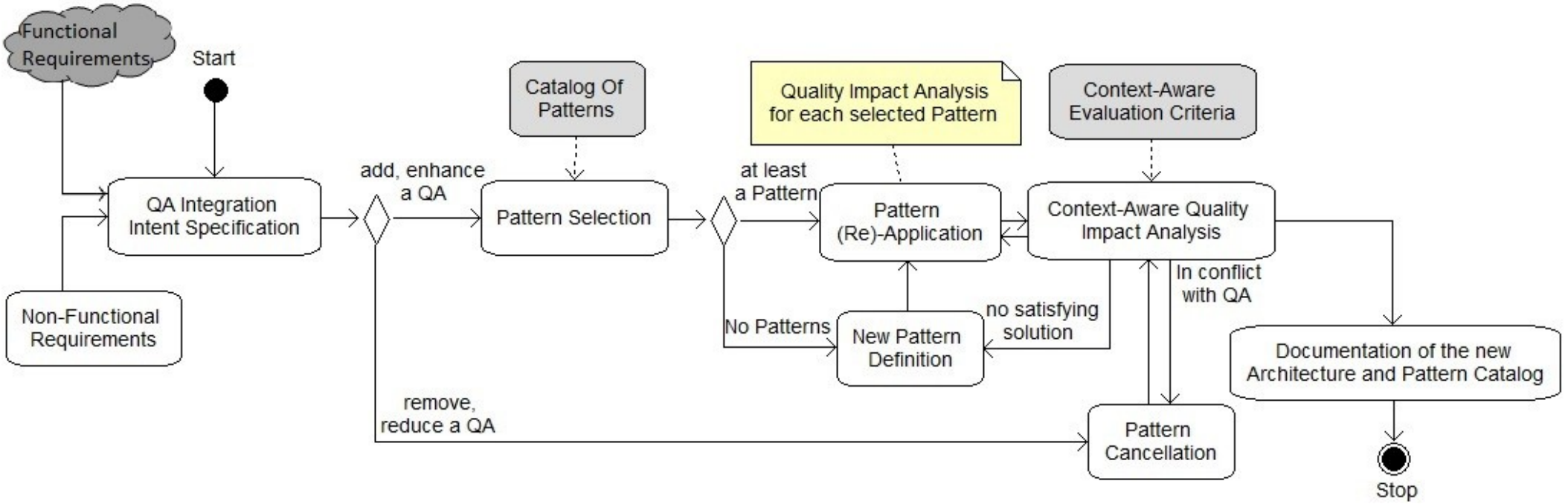


Pattern	Trusted Sub-system (1)	Exception Shielding (2)	Replication (3)	Service Fa-cade (4)	Brokered Authen-tication (5)
Quality At-tribute	Access Security (QA1)	Data Security (QA2)	Reliability (QA3)	Portability (QA4)	Access Security (QA1)

Contenu de cette partie

- Introduction aux services et problématique
- Approche proposée
- Approche proposée par l'exemple
- Expérimentation de l'approche
- Conclusion

Approche proposée



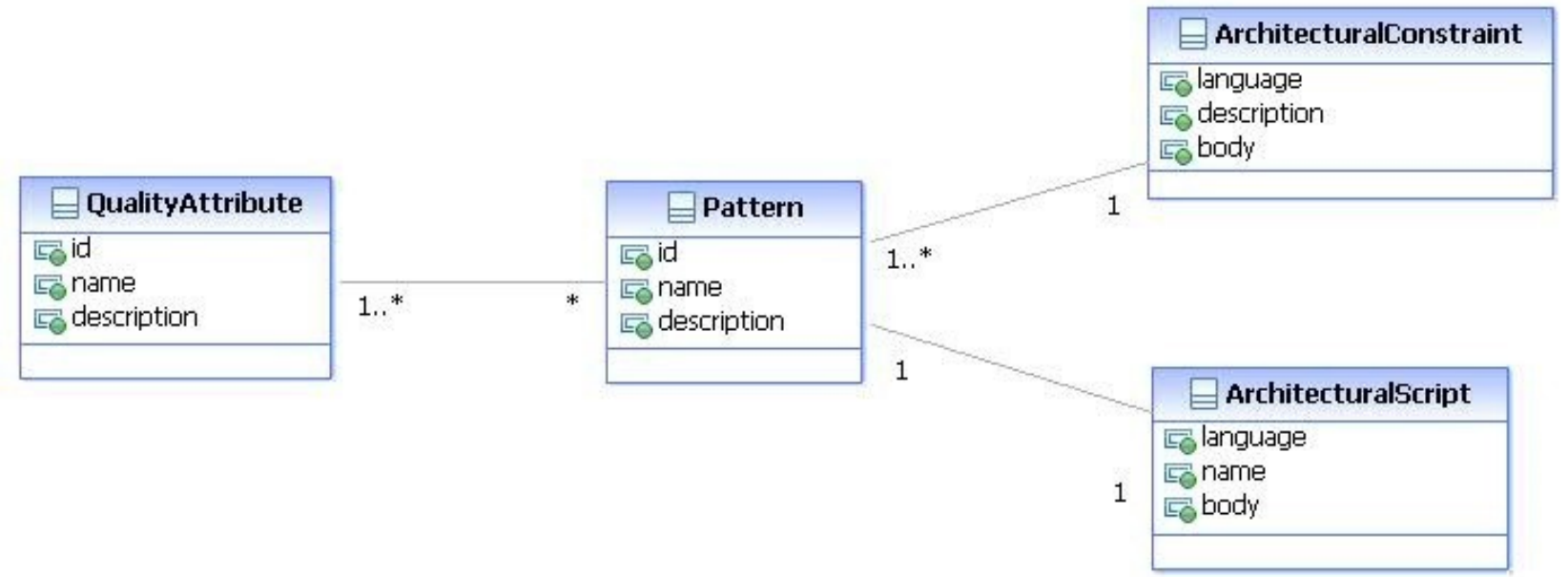
Template pour documenter une intention d'intégration d'un attribut qualité

Element	Scope	Description
Quality Attribute Integration	What?	State the quality attribute targeted by the integration activity.
Integration Kind	How?	State if the integration targets to add a new quality attribute, enhance, weaken or withdraw the quality attribute.
Related Quality Attribute	Ultimately what?	If the integration kind is withdrawing or weakening the quality attribute, state here the quality attribute which will be ultimately enhanced or added (left empty otherwise).
Architectural Area	Where?	Indicate where in the orchestration changes will occur.

Exemple de template

Integration Quality Attribute	Security/Data security
Integration Kind	Add
Related Quality Attribute	
Architectural Area	add pattern before ReserveVehicle, ReserveTrain, ReserveHotel Invoke activities

Documentation des décisions architecturales dirigée par des patrons SOA



Un langage de script pour processus BPEL : WS-BScript

(01) add (BpelElement element, BpelElement AttachedParentelement, int elementPosition)
(02) add (PartnerLinkElement element, String wsdlFileName)
(03) getPosition (String BpelElementName)
(04) create (BpelElement.Kind)
(05) remove (BpelElement element)
(06) wire (BpelElement element,PartnerLinkElement element,
String PartnerLinkOperationName)
(07) unwire (BpelElement element,String PartnerLinkElement,
String PartnerLinkOperationName)
(08) ask (String message)
(09) let variableName
(10) variableName = <expression>
(11) for(variableName : OrderedListVar) <actions>
(12) if (<condition>) <action1 or blocOfActions1>
[else <action2 or blocOfActions2>]
(13) query (String OCLExpression)
(14) scriptCall (String scriptName([parameters])
(15) return (BpelElement element)

Exemple de script WS-BScript : Trusted Subsystem Pattern

```
1  script applyTrustedSubsystemPattern (String BpelElementName,
2  String wsdlFileName, String partnerLinkOperationName,
3  String ProcessOperationName){
4  let position = getPosition (BpelElementName);
5  let ocl = "self->closure(eContents().oclAsType(EObject))->select(a|
6  a.oclIsKindOf(model::BpelType) and a.oclAsType(model::BpelType).name=
7  'BpelElementName')->collect(a:EObject| a.eContainer())->asSet()";
8  let elem = query (ocl);
9  let aAssign = create (BpelElement.Assign);
10 add (aAssign, elem, position+1);
11 let aSequence = create (BpelElement.Sequence);
12 add (aSequence, elem, position+2);
13 let aPartnerLink = create (BpelElement.PartnerLink);
14 add (aPartnerLink, wsdlFileName);
15 let aInvoke = create (BpelElement.Invoke);
16 add (aInvoke, aSequence, 0);
17 wire (aInvoke, aPartnerLink, partnerLinkOperationName) ;
18 let aIf = create (BpelElement.If);
19 add (aIf, aSequence, -1);
20 let aCondition = create (BpelElement.Condition);
21 add (aCondition, aIf, 0);
22 ask(aCondition);
23 let aAssign1 = create (BpelElement.Assign);
24 add (aAssign1, aIf, 0);
25 let aElse = create(BpelElement.Else);
26 add (aElse, aIf, -1);
27 let aSequence1 = create (BpelElement.Sequence);
28 add (aSequence1, aElse, 0);
29 let aAssign2 = create (BpelElement.Assign);
30 add (aAssign2, aSequence1, 0);
31 let aReply = create (BpelElement.Reply);
32 add (aReply, aSequence1, -1);
33 wire (aReply, ProcesspartnerLink, ProcessOperationName);
34 }
```

Algorithme d'assistance

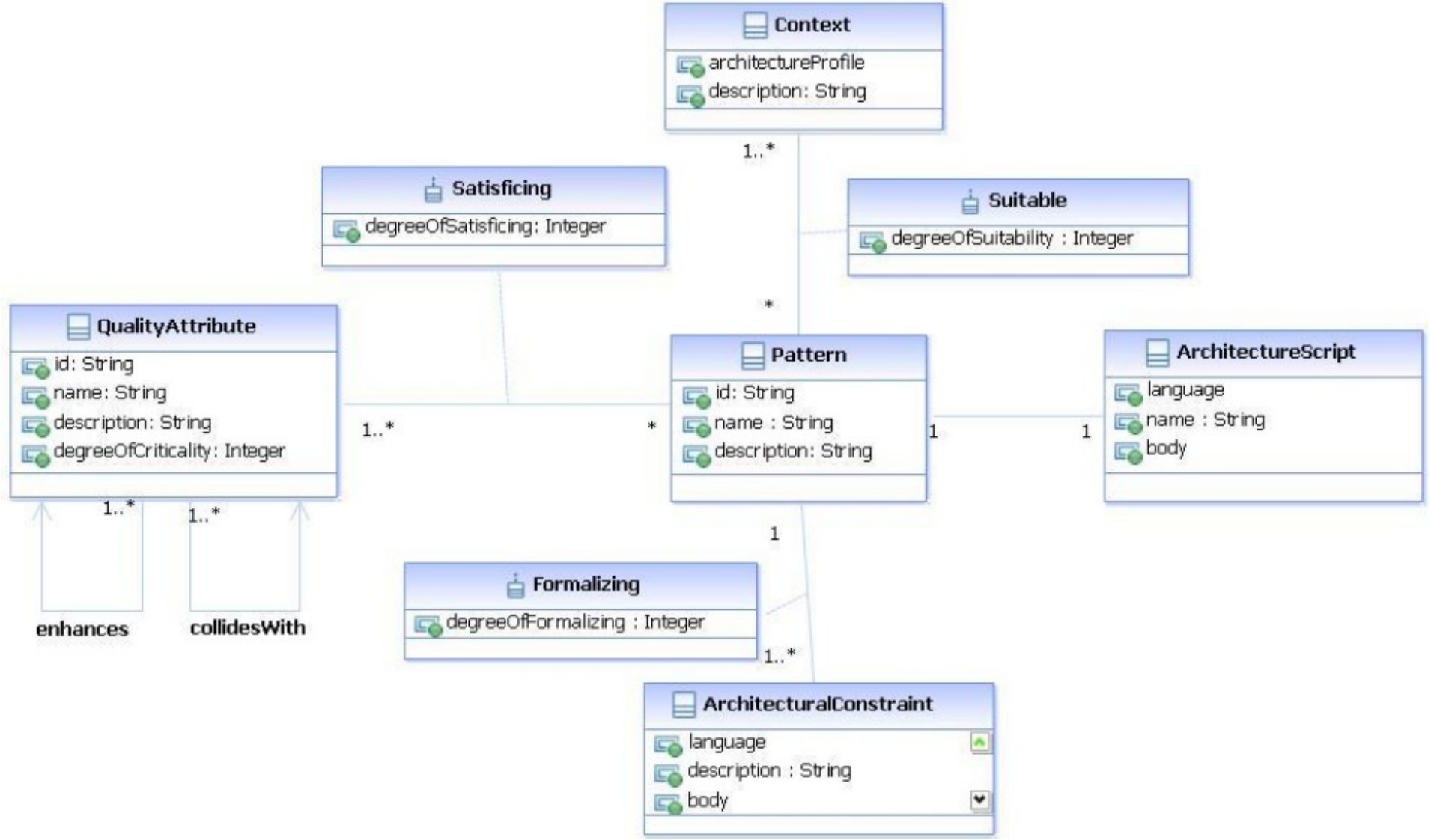
Algorithm 1: Quality Integration Assistance

```
1  begin
2      let AE := Architectural Element;
3      // a service orchestration;
4      and AD := Architectural Decision;
5      and AC := Architectural Constraint;
6      and QA := Quality Attribute;
7      and AT := Architecture Tactic;
8      // a couple composed of a QA and an AD;
9      and Doc:= architecture documentation associated to changed AE;
10     and let wsmParams:= { };
11     // an empty list of WSM system parameters (Aij, Wj);
12     and let rankedPatterns:= { } ;
13     // an empty list of pairs (AD, score);
```

Algorithme d'assistance -suite-

```
14  Function main(){
15  begin
16      after Pattern Application {
17      foreach (AT in Doc) do
18          QA := QA in AT;
19          AD := AD in AT;
20          checkArchitecturalConstraint(AD);
21           let A2j= ask for the context-suitability decision criterion
            value;
22          wsmParams := wsmParams + (A2j, W2);
23          let score := runWsmSystem( );
24          rankedPatterns := rankedPatterns + (AD,score);
25      end
26      sort(rankedPatterns);
27      displayResults();
28      let newAD := ask for AD associated to the new architecture, if
        any;
29      if newAD ≠ null then
30          let newQA := ask for the QA associated to newAD;
31      end
32      addNewArchitecturalTactic(newAD, newQA);
33      }
34  end
35  }
36 end
```

Relations entre attributs qualité et patrons



Exemple de contrainte d'architecture : Patron Service Façade

- La première activité dans le processus est un Receive lié à un *partner link* représentant le client du processus
- Le processus a comme comme dernière activité une activité de type Reply ayant le même partner link, le même port type et la même opération que l'activité Receive que ci-dessus
- Les seules activités de type Receive admises au milieu du processus doivent être précédées d'activités de type Invoke correspondantes. Ces dernières sont utilisées pour invoquer les opérations de services partenaires. Les activités Receive autorisées serviront à la réception des messages retournées par les opérations des services invoquées

Le patron Service Façade en ACL

- Sous-contrainte 1 : 1ère activité est un Receive

```
context MonApplication : Process inv :
let fst : Activity =
if self.activity->first().activity = null -- n'est pas une activité composite
then self.activity->first()
else if self.activity->first().oclIsTypeOf(Sequence)
  then self.activity->first().oclAsType(Sequence).activity->first()
  else null
  endif
endif
in if fst <> null
then fst.oclIsTypeOf(Receive)
else false
endif
```

Le patron Service Façade en ACL

- Sous-contrainte 2 : dernière activité est un Reply correspondant

```
...
if lst <> null -- En supposant que lst contient la dernière activité
then lst.oclIsTypeOf(Reply)
  and lst.oclAsType(Reply).partnerLink.name
    = fst.oclAsType(Receive).partnerLink.name
  and lst.oclAsType(Reply).portType.name
    = fst.oclAsType(Receive).portType.name
  and lst.oclAsType(Reply).operation.name
    = fst.oclAsType(Receive).operation.name
else false
endif
```

Le patron Service Façade en ACL

- Sous-contrainte 3 : ...

```
and
let activities : OrderedSet(Activity) =
self.activity->excluding(fst)->excluding(lst)->closure(activity)
in activities->forAll(a : Activity |
if a.oclIsTypeOf(Receive)
then activities->exists(aa : Activity |
    activities->indexOf(aa) < activities->indexOf(a)
    and aa.oclIsTypeOf(Invoke)
    and aa.oclAsType(Invoke).partnerLink.name
      = a.oclAsType(Receive).partnerLink.name
    -- and ... (même chose pour le portType et operation))
else true
endif)
```

Contenu de cette partie

- Introduction aux services et problématique
- Approche proposée
- Approche proposée par l'exemple
- Expérimentation de l'approche
- Conclusion

Processus BPEL : Appealed Assessments System

- Un processus qui représente le workflow d'un système de traitement des plaintes des clients
- AD 1 : Un service Data Relayer qui gère les accès à différents entrepôts de données (Service Façade -> Portabilité)
- AD 2 : Un service pour sécuriser les accès aux données : Trusted Subsystem Pattern -> Sécurité
- AD 3 : Faire des copies des données : Partial State Deferral Pattern -> Performance

Documentation d'une orchestration

- Architecture-Documentation :

1. Architecture-Tactic :

This tactic guarantees the Portability quality requirement by using a Service facade pattern

- Quality-Attribute name="Portability" degreeOfCriticality="high"
- Related-Quality name="Performance" relationship="CollidesWith" relationType="tight"
- Architecture-Decision name="Service facade pattern" degreeOfSatisficing="90"
- Architecture-Constraint profile="BPEL" degreeOfFormalizing="80"

Documentation d'une orchestration

2. Architecture-Tactic :

This tactic ensures the Security quality requirement by using a Trusted subsystem pattern

- Quality-Attribute name="Security" degreeOfCriticality="very high"
- Related-Quality name="Availability" relationship="Enhances" relationType="weak" influence="negative"
- Architecture-Decision name="Trusted subsystem pattern" degreeOfSatisficing="70"
- Architecture-Constraint profile="BPEL" degreeOfFormalizing="90"

Documentation d'une orchestration

3. Architecture-Tactic :

This tactic ensures the Performance quality requirement by using a Partial state deferral pattern

- Quality-Attribute name="Performance" degreeOfCriticality="high"
- Related-Quality name="Security" relationship="CollidesWith" relationType="tight"
- Architecture-Decision name="Trusted subsystem pattern" degreeOfSatisficing="60"
- Architecture-Constraint profile="BPEL" degreeOfFormalizing="70"

Scénarios d'évolution

- Scénario 1 : Court-circuiter le service d'authentification
  - Notification :
    - Affecte la décision 2 : Trusted Subsystem Pattern
    - Contrainte formalise jusqu'à 90% la décision
    - Garantit la sécurité jusqu'à 70 %
    - Attribut très hautement critique
    - Couplé faiblement à l'attribut disponibilité : enhancement-negative

- Scénario 2 : Ajout d'un service d'archivage, supprimer le service Data Relayer pour améliorer les performances et la disponibilité
  - Notification : ...

Contenu de cette partie

- Introduction aux services et problématique
- Approche proposée
- Approche proposée par l'exemple
- Expérimentation de l'approche
- Conclusion

Préparation de l'expérimentation

- Question de recherche :
  Comparée à une évolution manuelle de la qualité, est-ce que le support automatique, fourni par l'approche, donne une aide substantielle aux architectes ?

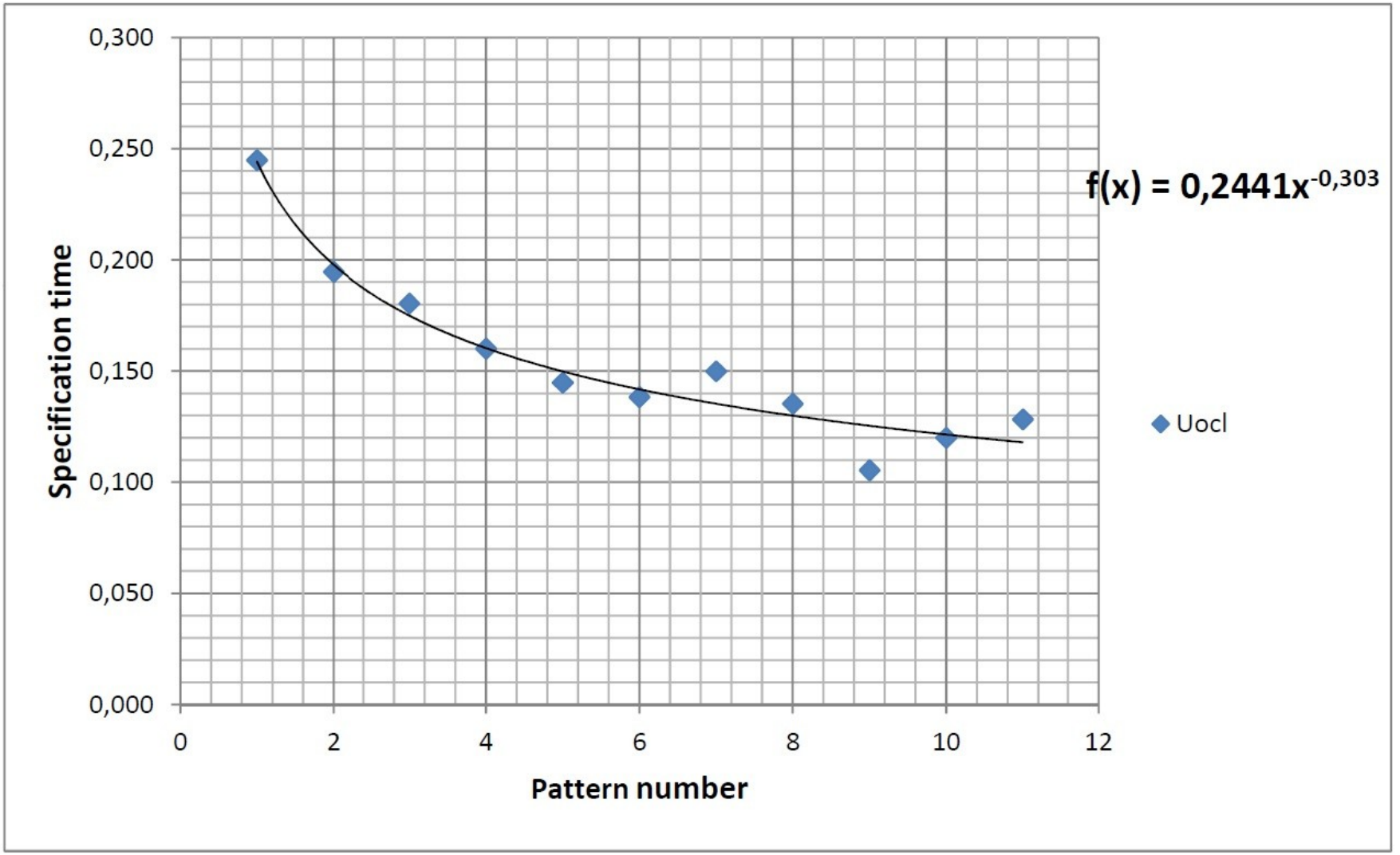
- Mesurer le surcoût de l'approche et à partir de quand l'approche devient rentable

- Dataset : 11 patrons réels + 5 patrons fictifs (variation aléatoire du nombre : 1) d'éléments BPEL pour mesurer le temps de spec des scripts, et 2) de tokens pour le temps de spec des contraintes)

Mesures du temps de spécification des contraintes

Pattern	Tocl (min)	Var	CD	1-CD	Tocl*(1-CD)	Uocl	Uocl-Umec	NbTokens
Facade(1)	40,25	1,79	0,046	0,975	39,24	0,072	0,062	154
Trusted Sub-System(2)	78,5	0,16	0,032	0,954	74,89	0,058	0,048	366
Passive Replication(3)	65,5	4,04	0,268	0,968	63,40	0,055	0,045	333
Smart Replication(4)	115,5	5,54	0,103	0,732	84,55	0,049	0,039	497
Naive Replication(5)	68	0,66	0,015	0,897	61	0,044	0,034	394
Exception Shielding(6)	36	1,16	0,025	0,985	35,46	0,044	0,034	239
Message Screening(7)	63,5	2,79	0,081	0,919	58,36	0,043	0,033	365
Brokered Authentication(8)	56	1,54	0,058	0,942	52,75	0,041	0,031	363
Test-based Partial state Deferral(9)	62,25	1,62	0,149	0,851	52,97	0,036	0,026	459
Event-based Partial state Deferral(10)	75,5	2,16	0,206	0,794	59,95	0,036	0,026	461
Partial Validation(11)	30	1,29	0,018	0,982	29,46	0,034	0,024	213

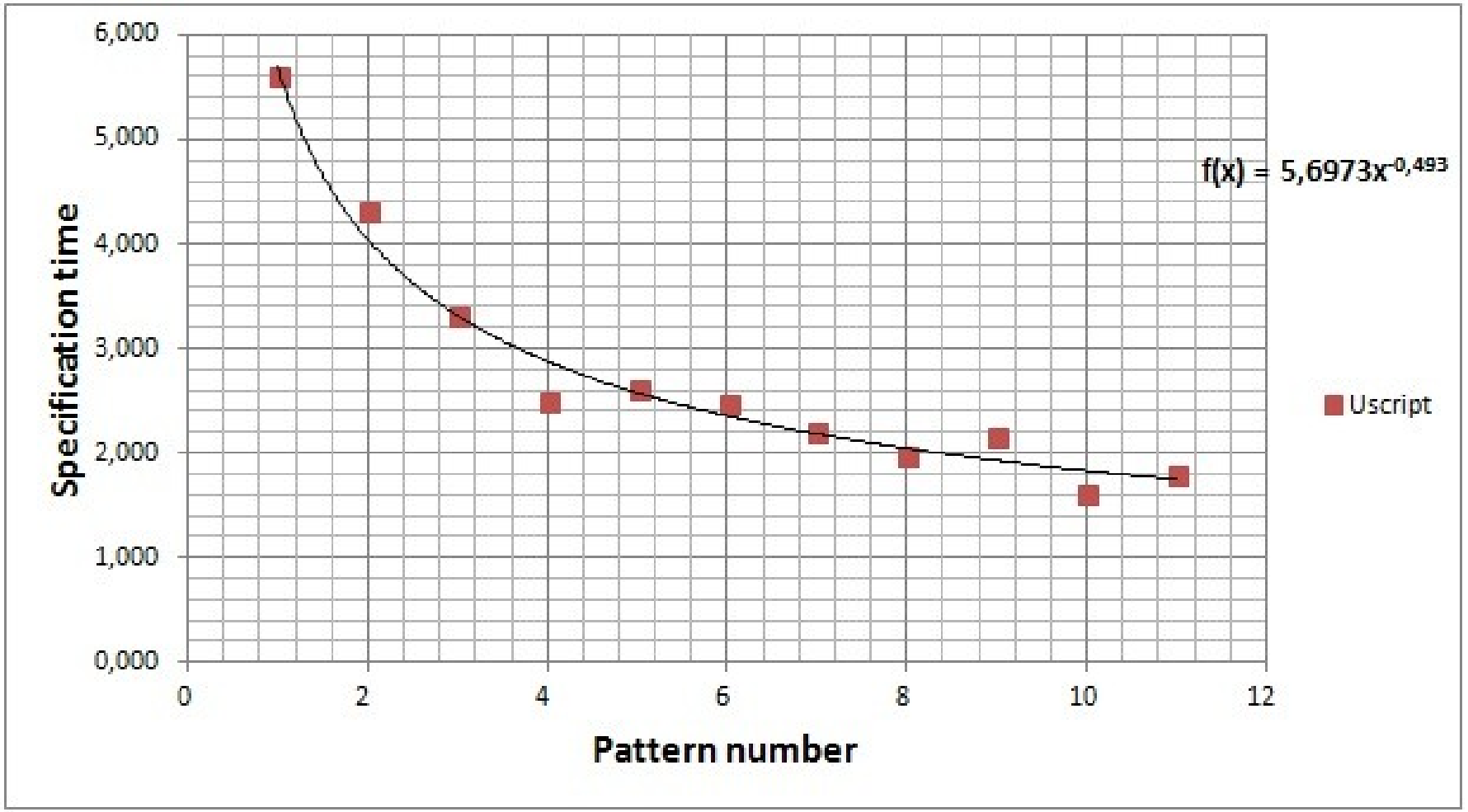
Extrapolation des mesures pour les patrons simulés



Mesures du temps de spécification des scripts

Pattern	Tscpt	Var	CD	1-CD	Tscpt*(1-CD)	Uscpt	Uscpt-Umec	NbBpel Elem
(1)	39	1,54	0,0378	0,9622	37,52	7,50	5,615	5
(2)	75	2,54	0,0905	0,9095	68,21	6,20	4,311	11
(3)	90	3,5	0,1297	0,8703	78,33	5,22	3,332	15
(4)	81,5	4,66	0,1353	0,8647	70,47	4,40	2,514	16
(5)	50	2,16	0,0967	0,9033	45,17	4,52	2,627	10
(6)	38	1,16	0,0771	0,9229	35,07	4,38	2,494	8
(7)	40,25	0,87	0,0819	0,9181	36,95	4,11	2,216	9
(8)	38,5	1,04	0,0968	0,9032	34,77	3,86	1,974	9
(9)	36	0,29	0,1011	0,8989	32,36	4,04	2,155	8
(10)	50,75	0,79	0,1016	0,8984	45,59	3,51	1,617	13
(11)	19,50	0,29	0,0514	0,9486	18,50	3,70	1,809	5

Extrapolation des mesures de temps de spec des scripts



Mesures de temps sur les patrons réels

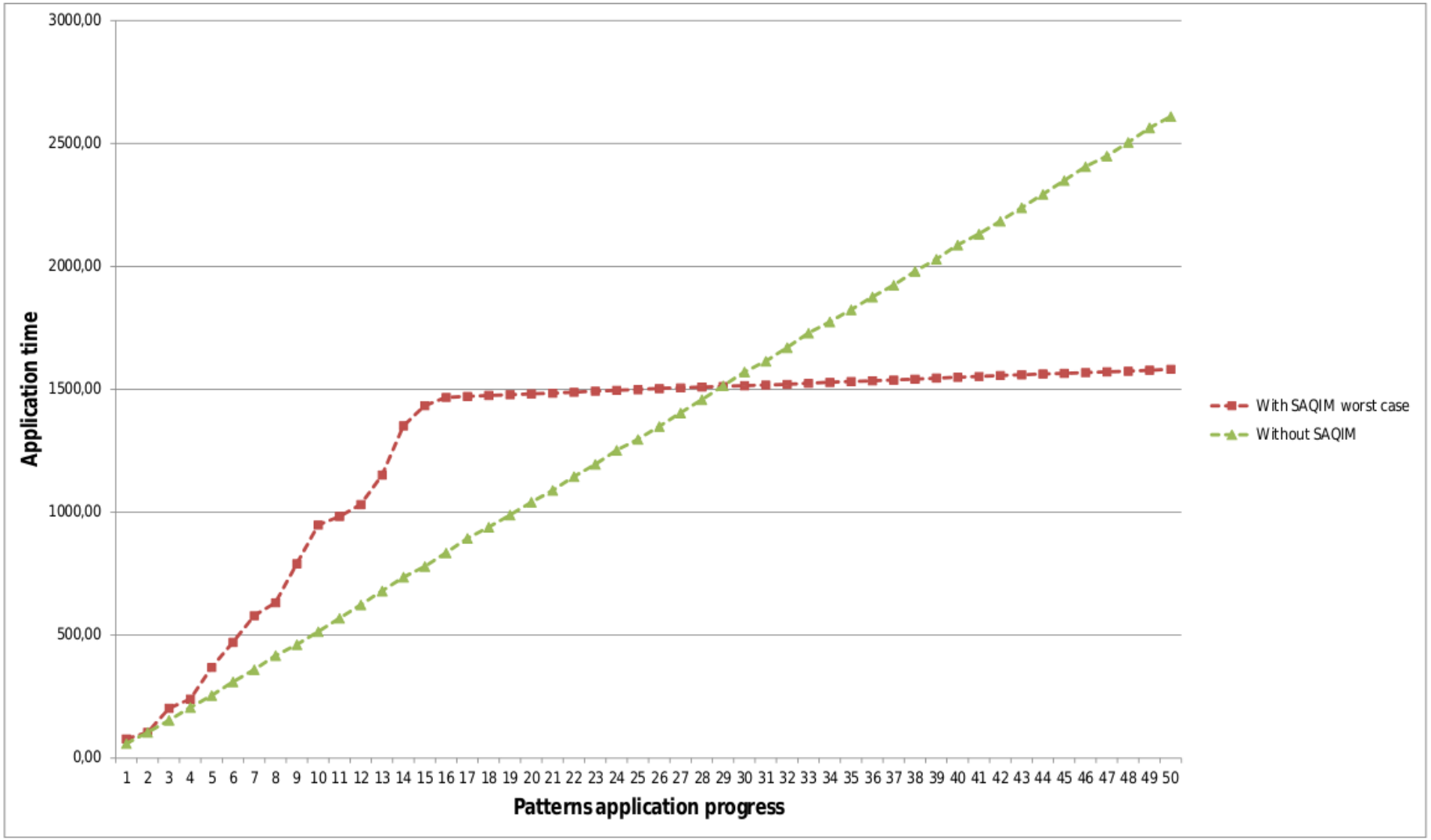
Pattern	Time (minutes)				
	without using SAQIM	using SAQIM			pattern specification time
		(a)	(b)	(c)	
(1)	43,37	0,59	0,72	1,57	76,77
(2)	52,42	0,65	0,39	3,20	143,10
(3)	44,47	0,93	0,66	1,40	141,73
(4)	56,28	0,86	1,00	1,10	155,02
(5)	56,27	0,71	0,65	1,03	106,16
(6)	59,12	0,67	1,18	2,17	70,53
(7)	49,42	1,17	1,23	2,05	95,31
(8)	49,23	0,63	1,03	1,58	87,53
(9)	56,16	0,57	1,18	2,16	85,33
(10)	48,09	0,56	0,98	1,25	105,54
(11)	57,48	0,71	1,09	1,54	47,96

Colonne (a) : temps d'application automatique
Colonne (b) : temps de configuration des contraintes
Colonne (c) : temps de documentation des patrons

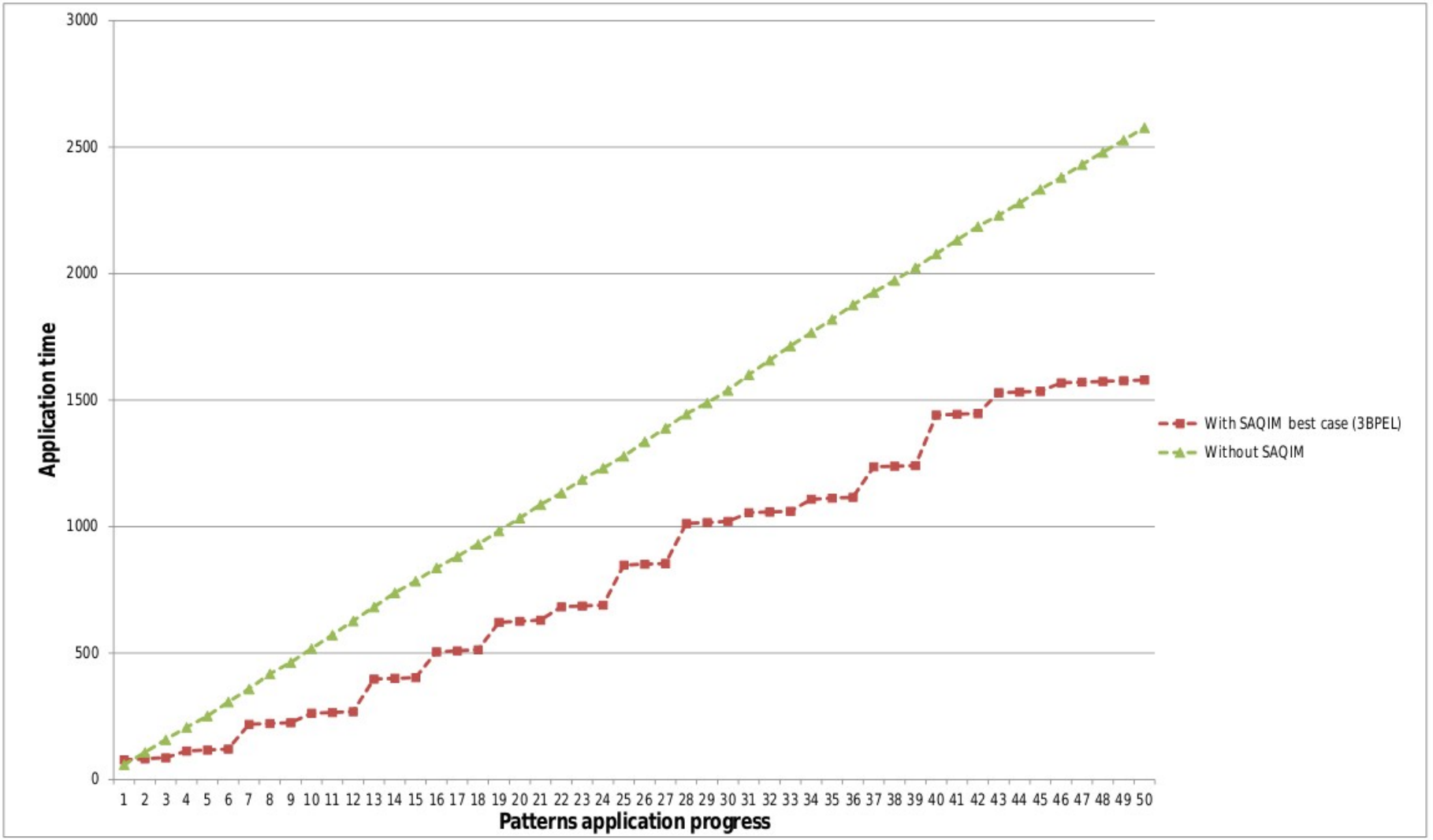
Simulation de l'application des patrons

- 50 applications fictives de ces patrons sur un processus BPEL réel
- Étude de 3 cas :
  - Pire des cas : appliquer d'abord une fois tous les patrons dans un ordre aléatoire, puis ré-appliquer les mêmes patrons
  - Meilleur des cas : appliquer en parallèle les patrons
  - Cas aléatoire

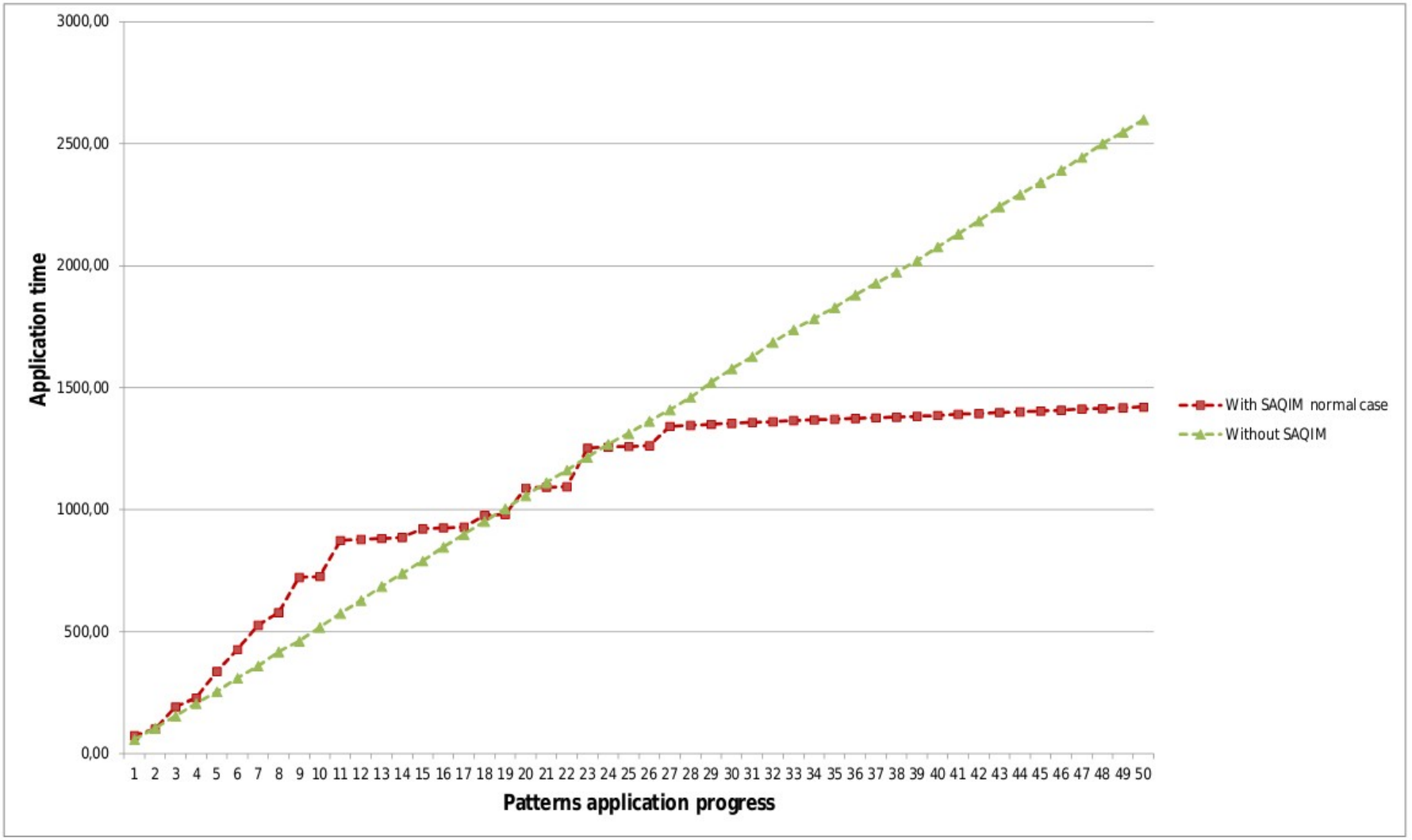
Le pire des cas



Le meilleur des cas



Le cas aléatoire



Contenu de cette partie

- Introduction aux services et problématique
- Approche proposée
- Approche proposée par l'exemple
- Expérimentation de l'approche
- Conclusion

Synthèse et perspectives

- Approche raffinée d'assistance à l'évolution
- Application aux processus BPEL comme implémentations d'orchestrations de services
- Thèse de Tarek Zernadji (qui va être soutenue en février 2016)
- Travaux futurs :
  - Définir un système décisionnel qui étudie l'effet de l'application de toutes les combinaisons possibles de patrons sélectionnés
  - Expérimenter pour évaluer les étapes du processus individuellement

Questions