Modèles sémantiques de composants pour l'ingénierie des systèmes d'information

Gwladys Guzélian, Corine Cauvet

Modèles Sémantiques de Composants pour l'Ingénierie des Systèmes d'Information

Gwladys Guzélian, Corine Cauvet

LSIS
Université Paul Cézanne, Aix-Marseille III
Avenue Escadrille Normandie Niemen
13397 Marseille cedex 20
gwladys.guzelian@lsis.org
corine.cauvet@univ.u-3mrs.fr

Résumé. L'ingénierie à base de composants s'impose peu à peu dans le développement des systèmes d'information (S.I.). Pourtant, elle n'a pas encore atteint son niveau de maturité. Le terme même de composant est souvent défini de façon imprécise et parfois contradictoire, les modèles de composants proposés peuvent avoir des finalités et des contextes d'utilisation très différents. Dans ce papier, nous nous intéressons aux modèles sémantiques de composants, c'est-à-dire des modèles visant à associer aux composants une connaissance qui guide leur usage. Nous étudions trois types de modèle : les approches « Web Services Sémantiques », les approches «patrons » et les approches de modélisation de domaine. L'étude comparative de ces approches permet de dégager des différences importantes en terme de finalité, de spécification et d'aide à l'utilisation des composants.

Mots-clés : Composants sémantiques, systèmes d'information, Web Services, Patrons, Modélisation de domaine.

1 Introduction

L'ingénierie à base de composants s'impose peu à peu dans le développement des systèmes d'information (S.I.). Le rôle croissant et diversifié que jouent le Web, l'Internet et l'Intranet dans la conception et la mise en ligne d'applications va amplifier ce phénomène.

Pourtant, l'ingénierie des systèmes d'information à base de composants n'a pas encore atteint son niveau de maturité. Sur le plan méthodologique, il n'existe pas d'approche de développement complètement orientée composant. Le terme même de composant est souvent défini de façon imprécise et parfois contradictoire, les modèles de composants proposés peuvent avoir des finalités et des contextes d'utilisation très différents (Fellner & Turowski, 2000), (Heineman & Councill, 2001). Malgré cette diversité, la tendance qui s'impose est de considérer un composant comme une brique logicielle permettant d'organiser et de réutiliser du logiciel. En conséquence, les

composants sont des artefacts logiciels, leur spécification est relativement technique et leur usage reste limité à la phase de conception d'architecture logicielle et de production logicielle.

La possibilité d'utiliser une approche orientée « composant sémantique » présente pourtant un réel intérêt. La description du problème auquel le composant répond, du contexte dans lequel il peut être utilisé et des lois de composition qui permettent de l'assembler à d'autres composants est essentielle pour mettre en œuvre une approche systématique de développement à base de composants.

Dans ce papier, nous étudions trois types de modèles de composants qui relèvent de l'approche « composant sémantique ».

- L'approche **« Web Services Sémantiques »**,
- L'approche **« Patrons »**,
- L'approche **« Modèles de domaine »**.

Il existe plusieurs études comparatives des approches orientées composants (Fellner & Turowski, 2000), (Heineman & Councill, 2001). Ces études ont essentiellement porté sur les composants technologiques. L'émergence de l'approche « Web Services Sémantiques » conduit à une nouvelle forme de composant qui vise à intégrer dans chaque composant une description sémantique pour en faciliter l'usage (la recherche, l'adaptation et la composition). Nous pensons que cette approche présente des ressemblances avec les approches « patrons » en génie logiciel et les approches « modèles de domaine » en ingénierie des connaissances. Dans ce papier, nous comparons ces différentes approches sur un même ensemble de critères. Les critères ont été choisis pour permettre une comparaison des modèles conceptuels sous-jacents aux différentes approches.

Le papier est organisé en 3 sections. Dans la **partie 2**, nous présentons les principes de spécification d'un modèle sémantique de composant. Dans la **partie 3**, nous analysons chacune de ces approches en fonction des principes présentés. Dans la **partie 4**, nous effectuons une analyse comparative de ces approches.

2 Principes de spécification des composants sémantiques

Il existe de nombreuses définitions de la notion de composant. Toutes ces définitions font apparaître une grande diversité des modèles proposés pour décrire les composants. Nous rappelons qu'un modèle sémantique de composants vise à associer aux composants une connaissance qui guide leur usage. Nous caractérisons un modèle sémantique au moyen de six principes : l'abstraction, la variabilité, la contextualisation, la composition, l'interopérabilité et la localisation. Ces six principes ont été choisis pour, d'une part, caractériser les approches à un niveau conceptuel et d'autre part mettre en évidence à la fois leurs points communs et leurs différences.

2.1 Principe d'abstraction

Par analogie avec l'approche objet, un composant comporte une « interface » visible par les clients susceptibles de l'utiliser et une « implémentation » qui contient la connaissance effectivement réutilisable fournie par le composant. Appliqué aux composants sémantiques, le principe d'abstraction vise à mettre en avant le besoin auquel un composant répond plutôt que les détails de la solution qu'il fournit.

Nous pensons qu'un composant doit posséder une interface centrée sur un problème métier, la partie implantation devant fournir une solution technique. Aujourd'hui, un certain nombre d'auteurs proposent d'utiliser des modèles de buts (« Business Goals ») ou des modèles de tâches pour décrire ces problèmes métier (Eriksson & Penker, 2000), (Motta & Zdrahal, 1998).

2.2 Principe de variabilité

Le principe de variabilité permet d'associer à un même concept plusieurs points de vue. Par exemple, le concept de *« personne »* peut avoir différentes représentations dans le domaine de la gestion, il peut correspondre à un abonné que l'on gère dans une bibliothèque, à un client que l'on gère dans une banque ou encore à un assuré géré dans une compagnie d'assurance.

Dans les composants orientés problème, la variabilité est exprimée à travers les différentes solutions possibles d'un même problème. Au niveau technique le principe de variabilité est mis en œuvre par différents mécanismes (Gurp & Bosch, 2001) comme les paramètres, l'héritage, les générateurs d'applications, la programmation « orientée aspect », la technologie à base de « frames »… Ce principe est la clé de la réutilisation, il permet l'adaptation du composant dans différentes situations et donc sa réutilisation dans différents contextes.

2.3 Principe de contextualisation

Complémentaire au principe de variabilité, le principe de contextualisation permet de discriminer les différentes variantes d'un même concept. Ce principe est essentiel pour guider le choix d'un composant. Dans la spécification d'un composant, la connaissance contextuelle définit la situation dans laquelle le composant est particulièrement adapté. Ce principe prend toute son importance lorsqu'un composant a été spécifié selon une approche orientée problème. En effet un problème de conception peut être résolu de différentes manières, en fonction de besoins particuliers à satisfaire ou de contraintes spécifiques à respecter. Ces différentes solutions doivent être différenciées en explicitant leur contexte d'application pour guider le choix de la meilleure.

2.4 Principe de composition

La mise en œuvre d'une approche à base de composants nécessite l'assemblage des composants pour aboutir à une solution globale. Les règles d'assemblage des composants doivent être intégrées et spécifiées au sein des composants. Les règles d'assemblage peuvent être décrites à différents niveaux d'abstraction. Au niveau

logiciel, les langages de description d'architectures (ADL : « Architecture Description Languages ») utilisent des connecteurs pour modéliser les interactions entre les composants et les règles qui régissent ces interactions. A un niveau conceptuel, les connecteurs peuvent exprimer les lois de fonctionnement d'un domaine telle une relation entre un client et une ressource ou une relation contractuelle entre deux partenaires, un client et un fournisseur par exemple. Dans une approche composant, il est important de gérer les composants et les connecteurs comme des concepts de même niveau. Les composants comme les connecteurs doivent pouvoir être réutilisés et adaptés.

2.5 Principe d'interopérabilité

L'interopérabilité est essentielle pour permettre l'assemblage et la coopération de composants hétérogènes et souvent distribués.

Comme pour la composition et la variabilité, l'interopérabilité peut être traitée à différents niveaux. Au niveau technique, les modèles tels que CORBA, COM / DCOM ou EJB visent cet objectif d'interopérabilité au moyen de protocoles de communication. Au niveau conceptuel, l'interopérabilité est basée sur des modèles de référence ou des ontologies (Vargas Solar & Doucet, 2002) et des règles de correspondance assurant le passage entre le modèle de référence et les modèles spécifiques à chaque environnement.

2.6 Principe de localisation

Les composants sont des artefacts dont l'usage nécessite la mise en œuvre d'un processus de recherche. Le principe de localisation répond à un besoin de recherche de composants dans une ou plusieurs bases. Ce principe peut être mis en œuvre au sein d'une base de composants par un modèle d'organisation de type hiérarchique, réseau… qui guide la recherche. Une autre façon de satisfaire ce principe est de spécifier et d'implanter le processus de réutilisation pour supporter entièrement la recherche en tenant compte du besoin exprimé et de la situation courante. Le principe de localisation doit aujourd'hui être étendu pour prendre en compte des sources de composants distribuées.

Ces six principes permettent de faire la différence entre objet et composant. Ils permettent aussi de distinguer composant sémantique et composant logiciel. La prise en compte de la variabilité, de la connaissance contextuelle et de l'orientation problème dans la spécification des composants permet d'associer dans un composant une connaissance réutilisable et une connaissance qui guide sa réutilisation.

3 Modèles sémantiques de composants

Dans cette section, nous étudions trois types de modèles sémantiques de composants sur la base des principes définis dans la section précédente.

3.1 Les approches WSS (Web Services Sémantiques)

Dans ces approches, les composants sont considérés comme des services externes (disponibles sur le Web) pouvant coopérer pour réaliser une certaine fonctionnalité. Nous présentons d'abord les « Web Services » et ensuite leur extension avec les « Web Services Sémantiques ».

3.1.1 Les Web Services

D'une façon générale, les Web Services sont des fragments d'application de gestion identifiés par une URL. Ils correspondent à des composants logiciels réutilisables qui réalisent des tâches spécifiques en utilisant des mécanismes standards orientés Web (OWL-S, 2004).

Un service est décrit à l'aide du langage WSDL qui le spécifie en termes de messages et d'opérations qu'il propose. La description d'un service avec WSDL comprend deux parties indépendantes **(Fig. 1)** : l'**interface** qui définit les éléments caractérisant les informations communes d'une catégorie de services et la partie **implémentation** qui décrit comment un service d'une interface particulière est implanté.

La **figure 1** (Srivastava & Koehler, 2003) illustre avec le méta langage WSDL un service de planification de voyage dans une agence. Chaque opération est associée à un message d'entrée (« input message ») et/ou de sortie (« output message »). De ce fait, l'opération « Replan Itinerary To Flight Service » associée au message de sortie « BookingFailure » signifie que l'exécution de l'opération «re-planification d'une réservation d'un vol d'avion » est effectuée lorsque la réservation est annulée. L'expression des Web Services avec WSDL décrit les messages de façon syntaxique sans préciser leur sémantique.

3.1.2 Les Web Services Sémantiques

Les « Web Services Sémantiques » visent à enrichir la représentation des Web Services avec des connaissances relatives à l'objectif du service, au processus mis en œuvre par le service et à ses contraintes d'utilisation. Ces connaissances peuvent alors être exploitées par des agents pour automatiser la recherche**,** l'invocation, la composition et l'exécution des Web Services. Les langages de description des « Web Services Sémantiques » assurent le marquage sémantique des Web Services, et rendent explicite la connaissance nécessaire à la sélection et à la composition (McIlraith et al., 2001).

Par exemple, l'approche OWL-S (OWL-S, 2004) est une extension de l'approche DAML-S qui propose une ontologie de services dans laquelle chaque service est décrit selon trois dimensions : le **« Profile »** du service qui précise les fonctionnalités du service, le « **Modèle** » du service qui décrit le processus mis en œuvre par le service, et le « **Grounding** » du service qui spécifie en termes de messages les règles d'interaction avec le service.

Cette ontologie de services enrichit WSDL et UDDI pour permettre un accès et une utilisation du service à partir de son contenu (objectif, messages….) plutôt que par des mots clefs.

3.1.3 Analyse des approches WSS

Tout d'abord, l'approche OWL-S vérifie le principe de localisation et de contextualisation. Par exemple, si l'on cherche un service permettant de commander un appareil photo numérique, on peut formuler des requêtes complexes comme « commander un appareil photo numérique de marque Sony et d'un poids inférieur à 500 gr ». On peut également affiner le contexte d'utilisation du service en précisant les notions de coût et de qualité de service tels les délais de livraison.

Ensuite, les principes de composition et d'interopérabilité sont étendus afin d'engendrer des services complexes combinant différents Web Services. Par exemple, la composition peut être utilisée pour faciliter la planification des vacances d'une personne qui souhaite ne pas dépenser plus de 1000 euros entre la réservation d'hôtel, le service de location de voiture, et les billets d'avion. Le plus souvent un service complexe est modélisé comme un processus (processus d'orchestration) faisant coopérer plusieurs activités. De plus, le principe d'interopérabilité est implicitement satisfait puisqu'il est possible de composer des services différents. Même s'ils sont implantés différemment, il faut par exemple, pouvoir comparer deux dates fournies par des services différents. L'utilisation d'une ontologie de services dans OWL-S permet ainsi de définir un langage commun à tous les services pour le partage et la réutilisation des données sur le Web.

D'autres approches de ce type, autres que OWL-S, possèdent des aspects sémantiques légèrement différents. Par exemple, dans l'approche IRS (Cabral et al., 2004), l'orientation problème est suggérée au moyen de modèles de tâches pour décrire sémantiquement les Web Services et les objectifs auxquels ils répondent (Motta & Zdrahal, 1998). En ce qui concerne l'approche WSMF, elle exploite la notion de but pour décrire les problèmes que devraient résoudre les Web Services (Cabral et al., 2004).

3.2 Les approches « Patrons »

L'approche « patrons » répond au besoin de formaliser et capitaliser des solutions à des problèmes récurrents de conception. Dans cette approche, un composant (appelé « patron ») est spécifié sous la forme d'un triplet <Problème, Solution, Contexte> (Gamma et al., 1995), (OMG, 2003), (Fowler, 1997).

3.2.1 Description des approches « Patrons »

Dans le triplet <Problème, Solution, Contexte>, le problème est un objectif de conception que souhaite atteindre le concepteur en réalisant un système d'information. La solution est un produit de conception représenté sous forme de modèles conceptuels, d'architectures… Le contexte définit la situation pour laquelle la solution décrite dans le composant est adaptée. La **figure 2** illustre partiellement un patron.

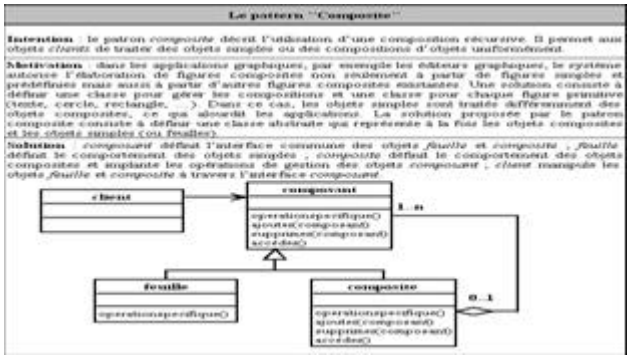

Fig. 2 – Un exemple de patron : le patron « Composite » (Gamma et al., 1995).

Ce patron fournit une représentation (sous forme d'un diagramme de classes) pour décrire des objets complexes. La structure du patron comprend l'intention, la motivation et la solution. Par analogie au triplet <Problème, Solution, Contexte>, l'intention correspond au problème à résoudre, la solution est décrite avec le langage UML et la motivation représente le contexte sous forme d'exemples d'application.

3.2.2 Analyse des approches « Patrons »

Dans ces approches, la plupart des composants proposés fournissent des solutions réutilisables qui aident à résoudre des problèmes-type de conception. Les principes mis en oeuvre par ce type d'approche concernent essentiellement les principes d'abstraction et de contextualisation. Le principe d'abstraction est vérifié par l'orientation problème. En effet dans cette approche, la description d'un composant contient de manière explicite une partie problème et une partie solution.

Les différentes approches telles que les « patrons de conception » (Gamma et al.,1995), les « patrons d'analyse » (Fowler, 1997) et les « Assets » de l'OMG (OMG, 2003) proposent différentes rubriques pour exprimer la connaissance contextuelle. Ces rubriques précisent les situations et les choix de conception qui ont conduit à la solution.

3.3 Les approches de modélisation de domaine

Une autre approche pour la conception de composants sémantiques est basée sur la modélisation de domaine. Ces approches sont issues soit du domaine de

l'Intelligence Artificielle (IA), par exemple l'approche LISA soit du domaine du Génie Logiciel (GL), telle que la méthode FODA. Ces approches visent à produire des modèles génériques ou modèles de domaine. Elles se distinguent dans la manière de considérer un domaine. Dans l'approche IA, un domaine est vu comme un ensemble de problèmes à résoudre et est conceptualisé par un ensemble de buts qui peuvent être opérationnalisés. Dans l'approche GL, un domaine est vu comme un ensemble de caractéristiques communes aux applications de ce domaine. Un domaine est aussi appelé une « ligne de produit ».

3.3.1 Les approches modèles de domaine orientées « IA » : La méthode LISA

L'approche LISA place l'utilisateur dans le contexte de résolution de problèmes. Dans le langage LISA, les buts caractérisent l'ensemble des problèmes et sous-problèmes. A chaque but sont associées des " méthodes " qui sont déclarées à priori comme les mieux adaptées pour le résoudre. Dans cette approche, le modèle de buts est relativement riche : un but est défini à l'aide d'un état-but, d'un contexte, de critères de satisfaction, de méthodes associées et de préférences. Dans la **figure 3**, le but « définition du réseau à étudier » est associé à trois méthodes possibles pour le réaliser. La première et la seconde méthode sont décomposées en deux sous buts et la troisième méthode est terminale (Jacob-Delouis & Krivine, 1995).

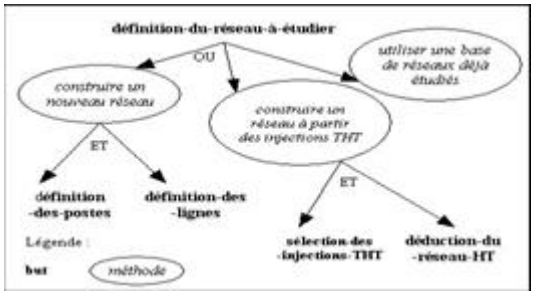

Fig. 3 – Le but « définition du réseau à étudier » et ses méthodes associées.

3.3.2 Les approches orientées « ligne de produit » : La méthode FODA

Dans l'approche FODA (Kang et al., 2003), le modèle le plus pertinent du point de vue de la réutilisation est le modèle de caractéristiques appelé *« features model »*. Ce modèle de caractéristiques permet de mettre en évidence, sous la forme d'un ensemble de hiérarchies, les caractéristiques communes et discriminantes des systèmes d'un même domaine. Les caractéristiques peuvent exprimer des fonctionnalités, des besoins en performance ou en ergonomie, mais aussi des besoins d'implantation (type de système d'exploitation…).

Le modèle précise les caractéristiques **obligatoires** pour un système du domaine ainsi que celles qui sont **optionnelles**, il fournit également les **alternatives** qui engendreront des choix au moment de la construction d'un système particulier. Le modèle définit également des **règles de composition** entre des caractéristiques. Par

exemple, l'utilisation d'une caractéristique peut imposer l'utilisation d'une autre. Enfin, les modèles de caractéristiques fournissent des **arguments** qui aident dans le choix d'options ou d'alternatives.

3.3.3 Analyse des approches de modélisation de domaine

D'un côté, LISA satisfait les principes d'abstraction en utilisant une orientation problème et de contextualisation grâce à l'introduction du concept de contexte dans la définition des buts.

Le principe de variabilité est également satisfait puisque chaque but est associé à plusieurs méthodes de résolution. Les concepts de but, méthode et contexte capturent des connaissances qui guideront l'utilisation des composants.

Le principe de localisation est aussi respecté. Le processus de recherche est un processus de satisfaction de buts induit par la hiérarchie de buts. Ce processus exploite la notion de contexte pour guider le choix de la méthode la plus adaptée au problème que souhaite résoudre l'utilisateur.

D'un autre côté, l'approche FODA respecte le principe de variabilité puisque le modèle des caractéristiques exprime tous les systèmes possibles d'un même domaine. De plus, le principe de contextualisation est satisfait au moyen, d'une part, des caractéristiques qui sont des propriétés permettant de discriminer les différents systèmes d'un même domaine et d'autre part, des arguments qui guident le choix des caractéristiques (Gurp & Bosch, 2001).

4 Comparaison des modèles sémantiques de composants

Les six principes proposés pour caractériser un modèle sémantique de composant ont permis d'étudier les différents modèles (**Table 1**).

Table 1. Evaluation des approches en fonction des principes de spécification

Approches / Principes	WSS	Patrons	Modèles de domaine	
			IA	Ligne de produit
Abstraction	-	orientation problème	orientation problème	-
Variabilité	oui (mais peu existante)	-	oui (pertinente)	
Contextualisation	formel : méta langage	informel : Phrase en langage naturel	informel : Phrase en langage naturel	
Composition	oui	-	-	
Interopérabilité	SOAP et XML	-	-	
Localisation	UDDI		Hiérarchies	

Dans cette partie, nous exploitons ces principes pour dégager des types d'approches. Nous utilisons trois points de vue conduisant chacun à une classification des modèles:

- le point de vue «**finalité »** étudie les modèles en fonction de l'usage des composants,
- le point de vue «**approche de spécification »** précise l'orientation choisie dans la nature des connaissances spécifiées au sein des composants,
- le point de vue «**support du processus d'utilisation »** caractérise le niveau d'aide fourni à l'utilisateur durant le processus de manipulation des composants.

4.1 Le point de vue finalité

Les propositions de modèles de composants sont nombreuses et leur utilisation dans le développement de systèmes d'information peut viser différents objectifs. Certaines approches sont orientées « réutilisation » et d'autres « intégration ».

-La **réutilisation** : il s'agit de considérer les composants comme des briques de développement existantes, pouvant être réutilisées dans plusieurs situations. Les principes d'abstraction et de variabilité caractérisent l'approche orientée « réutilisation ». Les logiciels de type «ligne de produit », les bibliothèques de composants relatifs à un domaine et les patrons visent essentiellement un objectif de réutilisation. Ces composants possèdent des propriétés de généricité leur permettant d'être réutilisable dans différents contextes.

-L'**intégration** : il s'agit de considérer les composants comme des services externes (disponibles le plus souvent sur le Web). L'intégration implique d'une part, l'accès transparent et uniforme aux services et l'échange de données éventuellement hétérogènes et distribuées. D'autre part, les composants peuvent s'exécuter à distance et coopérer pour réaliser une certaine fonctionnalité. Les principes d'interopérabilité, de composition et de localisation sont caractéristiques de l'approche orientée « intégration » et sont employés essentiellement par les approches WSS.

Bien que les objectifs de réutilisation et d'intégration soient complémentaires, les approches orientées composant existantes visent de façon généralement exclusive l'un ou l'autre de ces objectifs, conduisant ainsi à des propositions de modèles de composant très différents.

4.2 Le point de vue approche de spécification

Les modèles de composants étudiés font apparaître deux orientations dans la nature des connaissances spécifiées dans les composants.

Dans **l'orientation problème** la spécification d'un composant est centrée sur l'expression d'un problème. Dans cette approche le principe d'abstraction est mis en avant en distinguant explicitement les parties problème (interface du composant) et solution (implantation). Par exemple, l'orientation problème est sous-jacente aux nombreuses approches de type patrons qui définissent un composant comme une solution réutilisable dans un certain contexte pour répondre à un problème. Il en est de même dans l'approche LISA qui utilise la notion de but pour exprimer les problèmes et les sous problèmes. Par ailleurs, l'ajout d'une couche sémantique dans les approches

WSS permet de décrire des éléments tels que le besoin auquel le service répond. Des recherches doivent cependant encore être menées pour formaliser les buts et définir de véritables modèles de buts.

A l'inverse, **l'orientation solution** spécifie les composants à un niveau logiciel. Dans ces approches, le principe d'abstraction est peu respecté. Par exemple dans FODA le concept de « caractéristique » est utilisé pour définir des propriétés d'un système (donc d'une solution). On peut noter ici que les Web Services ont été étendus pour faire émerger les Web Services Sémantiques, cette extension ayant contribué à passer d'une approche solution à une approche plus orientée problème des Web Services.

4.3 Le point de vue support du processus d'utilisation des composants

Les différents modèles de composants étudiés permettent de spécifier une connaissance relative à leur utilisation. L'intérêt de cette connaissance est de fournir une aide au moment de l'exploitation des composants. On peut envisager trois niveaux d'aide :

- **Guidage** : Le processus est entièrement exécuté par l'utilisateur. Dans les modèles de type patrons, la connaissance contextuelle n'est pas formalisée et le processus de recherche et d'assemblage reste manuel. Cependant cette connaissance exprimée en langage naturel est facilement compréhensible par les utilisateurs.

- **Semi-automatique** : Le processus est automatisé mais nécessite des choix humains. Par exemple, LISA permet d'interpréter les connaissances spécifiées au sein des buts et méthodes telles que les préférences, les contextes favorables, et les paramètres, ce qui facilite l'automatisation et l'opérationnalisation du modèle LISA. Cependant, l'utilisateur peut intervenir dans le choix des méthodes à appliquer.

- **Automatique** : Le processus est réalisé par une application logicielle et l'utilisateur n'intervient pas dans le déroulement du processus. Ce type de support est un objectif des recherches actuelles sur les WSS. Les systèmes à base d'agents sont souvent expérimentés avec l'approche WSS pour rechercher les services et les orchestrer.

5 Conclusion

Dans cet article, nous avons présenté, d'une part, les principes que doit satisfaire un modèle sémantique de composants et d'autre part, trois types d'approche orientées modèle sémantique de composants. Nous définissons un composant sémantique comme un composant intégrant dans sa spécification une connaissance relevant de son usage (pour quel problème, dans quel contexte, avec quel autre composant l'assembler ?...). Les approches Web Services Sémantiques (WSS), les approches « Patrons » et les approches modèles de domaine sont représentatives de cette nouvelle forme de composant. Toutes intègrent dans des formes différentes une connaissance contextuelle qui guide plus ou moins l'usage des composants.

L'étude de ces trois types d'approche a permis de dégager des différences en termes de finalité, d'orientation de spécification et de support dans le processus de manipulation.

L'évaluation de ces approches montre une évolution importante des modèles de composants. Ils deviennent de plus en plus riches pour modéliser la connaissance relative à leur usage. Il s'agit d'une évolution indispensable si l'on veut mettre en œuvre de manière systématique et instrumentée une approche orientée composant.

Références

CABRAL L., DOMINGUE J., MOTTA E., PAYNE T., HAKIMPOUR F. (2004), Approaches to Semantic Web Services: An Overview and Comparisons. In Bussler, C. and Davies, J. and Fensel, D. and Studer, R., Eds. Proceedings First European Semantic Web Symposium (ESWS2004) The SemanticWeb: Research and Applications, Lecture Notes in Computer Science 3053(LNCS3053), pages pp. 225-239, Heraklion, Crete, Greece.

ERIKSSON H.E, M. PENKER M. (2000), Business Modeling with UML – Business Patterns at Work, OMG Press, Wiley, ISBN 0-471-29551-5.

FELLNER K.J., TUROWSKI K. (2000), Classification Framework for business Components, Proc. of the 3rd Hawai International Conference on System Sciences.

FOWLER M. (1997), Analysis Patterns – Reusable Object Models, Addison-Wesley, 1997.

GAMMA E., HELM R., JOHNSON R.E., J. VLISSIDES J. (1995), Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley.

GURP J.VAN, BOSCH J. (2001), On the notion of Variability in Software Product Lines, Proc. of WICSA.

HEINEMAN G., W. COUNCILL (2001), Component-Based Software Engineering: Putting the Pieces Together, Addison-Wesley.

JACOB-DELOUIS I., KRIVINE J.P. (1995), LISA : un langage réflexif pour opérationnaliser les modèles d'expertise, *Revue d'intelligence artificielle*, Vol. 9, n°1, 1995, p. 53 à 88.

KANG KYO KIM C., LEE K., LEEJ. , KIM S. (2003), Feature Oriented Product Line Software Engineering : Principles and Guidelines, chapter 2 in Domain Oriented Systems Development: Pratices and Perspectives, Taylor & Francis, 2003, p. 29-46.

MCILRAITH SHEILA A., TRAN CAO S., AND HONGLEI Z., Semantic Web Services, *IEEE Intelligent Systems*, Special Issue on the Semantic Web, (Vol. 16, No. 2) p. 46-53, March/April 2001.

MOTTA E., ZDRAHAL Z. (1998), A Library of Problem-Solving Components Based on the Integration of the Search Paradigm with Task and Method Ontologies, dans *International Journal of Human-Computer Studies*, vol. 49, n°4, octobre 1998.

OMG (2003), Reusable Asset Specification (RAS), Draft RFC submitted to OMG, version 2.1, août 2003.

OWL-S (2004), The OWS Services Coalition : OWL-S : Semantic Markup for Web Services, version 1.0 available at http://www.daml.org/services/owl-s/1.0/owl-s.pdf.

SRIVASTAVA B., KOEHLER J. (2003), Web service composition, current solutions and open problems, ICAPS 2003, Workshop on Planning for Web Services, Trento, Italy.

VARGAS SOLAR G., DOUCET A. (2002), Médiation de données : solutions et problèmes ouverts, In Actes des 2èmes Assises nationales du GdR I3, Décembre 2002.