

**UNIVERSITE SAAD DAHLEB DE BLIDA**

**Faculté des Sciences**  
Département d'Informatique

**MEMOIRE DE MAGISTER**

Spécialité : Systèmes d'informations et de connaissances

**IASA, UNE APPROCHE ARCHITECTURE  
LOGICIELLE ORIENTEE ASPECT**

Par

**KHIDER Hadjer**

Devant le jury composé de :

N. Benblidia	Maître de conférences A, U.S.D., de Blida	Présidente
W. Hidouci	Maître de conférences A, E.S.I. Alger	Examineur
D. Djenouri	Maitre de Recherche, C.E.R.I.S.T. Alger	Examineur
D. Bennouar	Maître de conférences U.S.D., de Blida	Promoteur

BLIDA, Juin 2012

## ملخص:

تقنيات البرمجة و الطرائق المرتبطة بها قد عرفت تطورا كبيرا عبر تاريخ الإعلام الآلي مع تطور أنظمة البرمجيات, هذه الأنظمة أصبحت أكثر تعقيدا مع مرور الوقت. أ  
البرمجة اعتمادا على القطع المركبة قد أثبتت نجاعتها في التحكم في مدى تعقيدات البرامج المطورة وأصبحت عامل رئيسي في نجاح تطوير المشاريع البرمجية عن طريق تسهيل صيانتها وتطور البرامج و السماح بتطوير أنظمة أكثر ضخامة وأكثر تعقيدا.  
هذا النمط من البرمجة يعد بإعادة الاستخدام ولكنه واجه مشاكل التشتت ومزج الخصائص مما استدعى تطبيق ال AOP على القطع البرمجية لمعالجة هذه المشاكل.  
البرمجة عن طريق فصل الخصائص الوظيفية عن الخصائص اللاوظيفية هو نموذج البرمجة الحديثة فهو امتداد للتقنيات البرمجية الحالية ولكن بشكل أكثر بساطة وأسهل للتغيير.  
اليوم الجوانب اللاتقنية والقطع البرمجية هما نموذج واعد الذي يعزز من إعادة الاستخدام وتبسيط وتطوير البرامج حتى الآن, التنفيذ المتزامن لهذين النموذجان هو حقل بحث واستكشاف مهمش. حتى الآن لا يوجد نموذج للقطع المركبة يدعم صراحة فصل الجوانب اللاوظيفية عن الجوانب الوظيفية.  
نقدم في هذه الأطروحة IASA-AOP, امتدادا للنموذج IASA المطور في المخبر LRDSI.  
الهدف من هذا العمل هو تدعيم نموذج IASA بفصل الجوانب اللاوظيفية بجميع جوانبها.

كلمات مفتاح: البرمجيات المركبة, البرمجة الموجهة اللاوظيفية, القطع المركبة, المنافذ,القطع اللاوظيفية, نقاط الوصل, نقاط العمل,النسج.

## RESUME

Les techniques de programmation et les méthodologies liées ont fortement évoluées tout au long de l'histoire de l'informatique avec l'évolution des systèmes logiciels, ces systèmes ont en effet tendance à devenir de plus en plus complexes. La programmation à base de composants logiciels a prouvé ses intérêts dans la maîtrise de la complexité des logiciels conçus et devenu un facteur critique dans la réussite de développement des projets logiciels en facilitant la maintenance et l'évolution du logiciel et autorisant le développement des systèmes volumineux en termes de taille mais aussi de complexité.

Ce style de programmation promet la réutilisation, mais est confronté aux problèmes de dispersion et de mélange de code représentant des propriétés transversales. L'application de la programmation par aspects (AOP) sur les composants logiciels permet de faire face à ces problèmes. La programmation dite par aspect permettant de gérer, de manière modulaire, ces préoccupations en les séparant du code de base.

La Programmation orienté aspect , un nouveau paradigme de la programmation étendant l'existant qui fait parties des techniques de programmation qui ont permis de simplifier l'écriture des programmes informatiques, en les rendant plus modulaire et plus faciles a faire évoluer.

Aujourd'hui, les Aspects et les composants logiciels sont deux paradigmes très prometteurs ; qui favorisent la réutilisation et simplifient le développement logiciel. A ce jour, la mise en œuvre simultanée de ces deux paradigmes reste un champ de recherche très faiblement explorée. A ce jour aucun modèle de composant ne supporte de manière explicite les aspects et plusieurs questions restent ouvertes. Parmi elles : Comment intégrer la représentation des aspects dans les composants logiciels ? Comment gérer les interactions et chevauchements entre aspects ?

Nous présentons dans ce mémoire IASA-AOP, une extension du modèle de composant IASA<sup>1</sup> définie au laboratoire LRDSI qui supporte la programmation par aspects. Cette extension consiste à doter l'approche IASA des composants orienté aspect et des ports orienté aspect.

L'objectif du travail est de faire supporte au modèle de composant IASA le concept d'aspect dans toute sa dimension : Une fois ce concept supporté, un architecte pourrait définir ses propres composants Aspect qu'il instancierait dans la partie contrôle d'un composant.

**Mots clés :** Architecture logicielle, Programmation Orienté Aspect, Composant, Port, Aspect, Point de jonction, Point d'action, Tissage, Advice.

---

<sup>1</sup> IASA : Integrated Architecture Software Approche développé au sein de laboratoire LRDSI, Blida

## ABSTRACT

The techniques of programming and methodologies strongly evolved throughout the history of data processing with the evolution of the software systems, these systems indeed tend to become increasingly complex. Component-Based Software Development proved its interests in the control of the complexity of the conceived software, and became a critical factor in the success of development of the software projects by facilitating the maintenance and the evolution of the software and authorizing the development of the bulky systems in terms of size but also of complexity.

This style of programming promises the re-use, but is confronted with the problems of *code scattering and tangling*. The application of Aspect-Oriented Programming on the software components makes it possible to face these problems. Programming called by aspect allowing managing, in a modular way, these concerns by separating them from the basic code.

Aspect-Oriented Programming, a new paradigm of the programming which made possible to simplify the writing of the programs data-processing, while making them more modular and easier has to make evolve.

Today, the software Aspects and components are two very promising paradigms which support the re-use and simplify the software development. To date, implementation the simultaneous of these two paradigms remains a field of research very slightly explored. To date no model of component supports in an explicit way the aspects and several questions remain open. Among them: How to integrate the representation of the aspects in the software components? How to manage the interactions and overlappings between aspects?

We present in this dissertation IASA-AOP, an extension of the model of component IASA defined in the laboratory LRDSI which supports the Aspect-Oriented Programming. This extension consists in equipping approach IASA with the aspect components and aspect ports.

The objective of work is to make supports to the model of component IASA the concept of aspect in its entire dimension: Once this concept supported, an architect could define his own Aspect components which it instantiated in the part controls of a component.

Key Words: Software architecture, Aspect-Oriented Programming, Component, Aspect, JoinPoint, PointCut, Weaving, Advice.

## REMERCIEMENTS

Je voudrais tout d'abord remercier les membres du jury pour m'avoir fait l'honneur d'accepter de juger ce travail de thèse.

Je tiens à remercier très chaleureusement Dr. BENNOUAR DJAMEL sans qui rien de tout cela ne serait arrivé. J'ai énormément apprécié les années de travail sous votre direction. Merci pour votre soutien sans lequel je n'aurais jamais réussi à aller au bout, vos conseils toujours lumineux et votre patience. Merci aussi pour le temps que vous m'as consacré au jour le jour pendant ces années.

Je remercie tout particulièrement mes chers parents pour leurs patiences, encouragement et leurs soutiens précieux tout au long ma carrière.

Je souhaite remercier vivement mes collègues de l'école doctorale Nadjat Zerf, Bou Jabbour Karim et mon ami Mohamed Zouggar.

Enfin, par ces remerciements, je tiens à ne pas oublier les membres de l'école doctorale SIC particulièrement la présidente de l'école doctorale Mlle N. Benblidia Maitre de conférences, U. de Blida ainsi que Mme S. Oukid, Maitre de conférences, U. de Blida.

Merci a tous ceux qui de près ou de loin m'ont aidé et soutenu pour que ce travail soit réalisé.

## TABLE DES MATIERES

RESUME	2
REMERCIEMENTS	5
TABLE DES MATIERES	6
LISTE DES ILLUSTRATIONS, GRAPHIQUES ET TABLEAUX	10
INTRODUCTION	14
1. Etat de l'art	21
1.1 Introduction	22
1.2 La programmation orientée aspect	22
1.3 L'orienté aspect et les composants logiciel	23
1.3.1 Les cadres de travail basé sur les composants logiciels (CBSE) ( <i>Component Based Software Engineering frameworks</i> )	25
1.3.1.1 Spring AOP	25
1.3.1.1.1 Introduction	26
1.3.1.1.2 Les aspects dans Spring AOP	26
1.3.1.1.3 Exemple illustratif	30
1.3.1.1.4 Evaluation	33
1.3.1.2 JBOSS-AOP	34
1.3.1.2.1 Introduction	34
1.3.1.2.2 Les aspects dans JBoss AOP	34
1.3.1.2.3 Exemple illustratif	37
1.3.1.2.4 Evaluation	38
1.3.1.3 JAC	39
1.3.1.3.1 Introduction	39
1.3.1.3.2 Les aspects dans JAC	40
1.3.1.3.3 Exemple illustratif	43
1.3.1.3.4 Evaluation	44
1.3.2 Les Aspects dans les langages basés sur le composant	44
1.3.2.1 Aspectual Component	45
1.3.2.1.1 Introduction	45
1.3.2.1.2 Les Aspects dans Aspectual component	45
1.3.2.1.3 Exemple illustratif	47
1.3.2.1.4 Evaluation	49
1.3.2.2 JASCO	49
1.3.2.2.1 Introduction	49
1.3.2.2.2 Les Aspects dans JAsco	49
1.3.2.2.3 Exemple illustratif	52
1.3.2.2.4 Evaluation	53
1.3.2.3 Caesar	54
1.3.2.3.1 Introduction	54
1.3.2.3.2 Les Aspects dans Caesar	54
1.3.2.3.3 Exemple illustratif	55
1.3.2.3.4 Evaluation	57
1.3.2.4 Open module	58

1.3.2.4.1	Introduction	58
1.3.2.4.2	Les Aspects dans Open module	59
1.3.2.4.3	Exemple illustratif	61
1.3.2.4.4	Evaluation	63
1.3.3	Aspect et architecture logicielle	64
1.3.3.1	TranSAT	64
1.3.3.1.1	Introduction	65
1.3.3.1.2	Les Aspects dans TranSAT	68
1.3.3.1.3	Evaluation	69
1.3.3.2	FAC	71
1.3.3.2.1	Introduction	71
1.3.3.2.2	Les Aspects dans FAC	72
1.3.3.2.3	Exemple illustratif	75
1.3.3.2.4	Evaluation	77
1.3.3.3	CAM/DAOP-ADL	78
1.3.3.3.1	Introduction	78
1.3.3.3.2	Les Aspects dans CAM/DAOP	78
1.3.3.3.3	Exemple illustratif	80
1.3.3.3.4	Evaluation	82
1.3.3.4	AspectLEDA	83
1.3.3.4.1	Introduction	83
1.3.3.4.2	Les Aspects dans AspectLEDA	84
1.3.3.4.3	Exemple illustratif	85
1.3.3.4.4	Evaluation	86
1.3.3.5	AC2-ADL	87
1.3.3.5.1	Introduction	87
1.3.3.5.1.1	Les Aspects Dans AC2-ADL	87
1.3.3.5.2	Exemple illustratif	90
1.3.3.5.3	Evaluation	91
1.3.3.6	Aspectual ACME	93
1.3.3.6.1	Introduction	93
1.3.3.6.2	Les Aspects dans Aspectual ACME	94
1.3.3.6.3	Exemple illustratif	94
1.3.3.6.4	Evaluation	95
1.3.3.6.5	Les Aspects dans AO-ADL	96
1.3.3.6.6	Evaluation	98
1.3.3.7	DAOP-ADL	99
1.3.3.7.1	Introduction	99
1.3.3.7.2	Les Aspects dans DAOP-ADL	100
1.3.3.7.3	Exemple illustratif	101
1.3.3.7.4	Evaluation	103
1.4	Conclusion	105
2.	Le modele de composant IASA	109
1.5	Introduction	109
1.6	LE MODELE A COMPOSANTS IASA	110
1.6.1	La vue externe et le concept d'enveloppe	111
2.2.1.1	Les enveloppes marquées	112
2.2.1.2	Les enveloppes de deploiement	113
2.2.2	La vue Interne du composant IASA	114
2.2.2.2	La partie opérative	114

2.2.2.3	La partie contrôle	115
2.2.2.2.1	Les composants de la partie contrôle	118
2.1	LE Point d'accès	118
2.3	Le Port	129
2.4	Conclusion	135
3.	Le modèle de composant IASA pour l'intégration des aspects	136
3.1	Introduction	138
3.2	L'architecture logicielle orientée aspect avec 3ADL	136
3.2.1	Le composant aspect	139
3.2.2	Les ports orientés aspects non métiers	141
3.2.3	Le point d'accès orienté aspect (ASPOAP)	142
3.2.4	Les points de jonctions/ joinpoints	143
3.2.5	Les coupes /pointcut	144
3.2.6	Déclaration des advices	146
3.2.7	Mécanisme d'injection d'un Advice	146
3.2.8	Tissage d'aspect	148
3.3	La composition et L'ordres d'exécution des aspects	149
3.4	La réutilisation des aspects	151
3.5	Evaluation	152
4.	IASA Studio	153
4.1	Introduction	160
4.2	Présentation générale d'IASA Studio	161
4.2.1	L'Editeur graphique	162
4.2.2	L'arbre <i>Architecture system</i>	163
4.2.3	L'arbre Source View	164
4.2.4	Le générateur de code	164
4.2.4.1	Projection vers ArchJava	165
4.2.4.2	Projection vers java	166
4.2.4.3	Projection vers aspectj	166
4.2.5	Outils de validation de l'architecture logicielle	168
4.2.5.1	Outil Compiler	168
4.2.5.2	Outil Weaver	168
4.2.6	Un consol d'affichage	168
4.2.7	Bibliothèque des composants	169
4.2.8	Bibliothèque des aspects	169
4.2.9	Archive de système	169
5.	<i>Validation de l'approche 3ADL</i>	170
5.1	Introduction	170
5.2	Application de mise en place de Guichet Automatique de Banque	170
5.2.1	Spécification de composant Bank	172
5.2.1.1	La spécification de composant <i>BANK AVEC 3ADL</i>	172
5.2.1.2	Composant_Compte_Bancaire	173
5.2.2	Spécification de composant composite <i>Distributeur de bille</i>	174
5.2.2.1	Spécification de composant SecurityCom	177
5.2.2.2	La spécification de composant Aspect VérificationSolde	177
5.2.2.3	La spécification de composant Aspect LogCmp	178
5.2.2.4	La spécification de composant AuthentificationComp	178
6.	Conclusion et perspective	180