هندسة البرمجيات المُقادة بالوكلاء. تغطية متكاملة لدورة الحياة البرمجية

المهندس لؤي جيرودية* **الدكتور محمد الحجي****

الملخص

يركز هذه البحث على تقصي النقاط التي لم تغطِ بشكل كامل ضمن منهجيات هندسة البرمجيات المُقادة بالوكلاء (AOSE) وقد اعتمد على مراجعة موسعة لهذه المنهجيات. ترتكز هذه الورقة بشكل أساسي على أن كل منهجية لها نقاط قوتها وضعفها وقد تركّز على بعض مراحل دورة الحياة البرمجية وليس عليها كلها مجتمعة. العمل المقدم هنا يقدم تمديداً لواحدة من أهم منهجيات هندسة البرمجيات المُقادة بالوكلاء (MaSE) التي تعدُّ من أقوى المنهجيات ولكنها لا تعير اهتماماً كافياً لمرحلة التحليل المبكّر للمتطلبات. يعتمد العمل على المكاملة مع إحدى المنهجيات التي تركز على مرحلة المتطلبات، وهذه المكاملة اقتضت تطوير مجموعة موسعة من قواعد التحويل بين بيئتين برموز ومفاهيم متباينة ، وبناء برنامج يؤتمت عملية التحويل. وتعرض هذه الورقة أيضاً مثالاً كاملاً case study نطبق عليه العمل المنجز.

الكلمات المفتاحية: وكلاء، الوكلاء الأذكياء، هندسة برمجيات، مخططات UML، مخططات AUML، نماذج تصميم.

* أعد هذا البحث في سياق رسالة الماجستير للمهندس لؤي جيرودية بإشراف الدكتور محمد الحجي

** مدرس بجامعة دمشق - كلية الهندسة المعلوماتية- جامعة دمشق

مقدمة:

يعدُّ البعض أن تصميم وبناء البرامج الحاسوبية من أعقد المهمات التي قد يقوم بها الإنسان وذلك نظراً إلى صعوبة تحقيقها بالشكل الأمثل ولكثرة العوامل التي تؤثر بها بشكل مباشر أو غير مباشر، لذا فإن إيجاد مبادئ منهجية لبناء البرامج كان أمراً حتمياً لضمان نجاح عملية التطوير. وبذلك كانت مبادئ هندسة البرمجيات التي أتت لنمذجة عمليات التطوير وإدارتها. أمَّا الوكيل البرمجي (Agent) فيُعد برنامجاً مُصغّراً يقوم مبدأ عمله على تأدية مهمات محددة لخدمة برنامج أساسي بهدف تنفيذ هذه المهمات بشكل منفصل عن تنفيذ البرنامج وعزل إجرائياتها عن الإجرائية الأساسية للبرنامج. أما هندسة البرمجيات المُقادة بالوكلاء (AOSE) فتعدُّ من الاتجاهات الحديثة في تطوير البرمجيات الحاسوبية، إذ تتميز بمقدرتها العالية على تمثيل الوكلاء ضمن النظام البرمجي بمقدار عالٍ من التجريد، كما أنها تعدُّ الصيغة الأكثر تطوراً لتحليل وتصميم وبناء الأنظمة البرمجية عالية التعقيد وذلك لقدرتها الكبيرة على التأقلم مع التغيرات المتلاحقة على بنية النظم البرمجية ومكوناته، وكذلك لدعمها لمبادئ التطوير المستقل عن بيئة العمل النهائية (نظام التشغيل). ففي تطبيقات الويب مثلاً قد يكون على الوكيل أن يتعامل مع العديد من أنظمة التشغيل المُختلفة وأن يتعامل مع اختلافاتها بشكل دائم.

1- مفاهيم الوكيل البرمجي (Agent)

لنبدأ أولاً بتعريف الوكيل البرمجي أو ما يُسمى بالوكيل الذكي، وهو عبارة عن جزء من برنامج مُحدد ومُستقل بذاته يسعى لتحقيق أهداف أساسية مُعيّنة لخدمة البرنامج الكلي، ولتحقيق هذه الأهداف يؤدي الوكيل عدة أدوار من خلال تنفيذه لعدة مهام إما تكون مسندة له بشكل مباشر أو أنّ الدور الذي يؤديه هذا الوكيل قد يقتضي القيام بمهمات إضافية جديدة يقوم الوكيل بأدائها. قد يكون هذا التعريف غامضاً بعض الشيء، ولكن يمكننا من خلاله أن نميز الطبيعة الحقيقية لعمل الوكيل البرمجي [Tveit, 2001]. لنوضّح ذلك نأخذ مثال الوكيل البرمجي الخاص بإدارة رحلات مُستخدم مُعيّن (الشكل 1) إذ نجد أنه يسعى للقيام بعدد مُعيّن من الأهداف المُحددة له، فعلى سبيل المثال يسعى الوكيل لتحقيق عدد من الأهداف الأساسية للبرنامج الرئيسي مثل تنظيم رحلة جوية مع العائلة، وتنظيم رحلة برية مع الأصدقاء، وتنظيم رحلة بحرية خاصة. فيتولى البرنامج الصغير (الوكيل) هذه المهمات الجزئية نيابة عن البرنامج الأساسي. كل هدف من هذه الأهداف يتطلب من الوكيل أن يؤدي أدواراً مُعيّنةً في أثناء عمله لتحقيق هذا الهدف، فمثلاً لتنظيم رحلة جوية مع العائلة يجب على الوكيل أن يمر بأدوار عديدة مثل البحث بين رحلات الطيران كلها عن رحلات تحقق شروط المُستخدم وعائلته، وتحديد أفضل رحلة من بين المجموعة الناتجة وفقاً لمعايير يُحددها المُستخدم، وحجز الرحلة المُختارة لأفراد العائلة كلها. بينما يقوم كل دور من هذه الأدوار بتنفيذ عدد من المهام والإجراءات التي يتطلبها الدور، فلحجز رحلة ما لكل أفراد العائلة قد يقوم الوكيل بمهام مثل اختيار أماكن متقاربة على الطائرة، ومعالجة عملية دفع قيمة التذاكر لشركة الطيران من الحساب المصرفي للمُستخدم.

الشكل 1 - الوكيل البرمجي الخاص بإدارة الرحلات

2- مفاهيم ومبادئ هندسة البرمجيات المُقادة بالوكلاء (AOSE)

جاءت منهجيات هندسة البرمجيات المُقادة بالوكلاء في تطوير البرمجيات واعدةً بتحقيق الكثير من الآمال في مجال تطوير الأنظمة الحاسوبية المُعقّدة المعتمدة على الوكلاء كتسهيل إجرائيات التطوير وزيادة فعالية البرامج الناتجة من خلال التركيز على تحسين مراحل التطوير المختلفة في دورة الحياة البرمجية.

تعتمد مبادئ هندسة البرمجيات المُقادة بالوكلاء على التركيز على عدد كبير من المفاهيم الخاصة لتطوير البرامج بصورة قياسية ومنتظمة وذلك تماشياً مع مواصفات البرامج ونوعية استعمالاتها. لذلك كانت العوامل المؤثّرة في نمذجة الأنظمة البرمجية تختلف بحسب نوع هذه الأنظمة وميزاتها وبحسب مستويات النمذجة المختلفة المطلوبة كما يأتي:

1. لنمذجة وكيل واحد: يتم التركيز على المبادئ التنظيمية والاجتماعية لتوصيف الوكيل بحد ذاته وهي مبادئ الدور (لتوصيف ميزات سلوك الوكيل)، الوظيفة (مجموعة الأدوار التي يقوم بها الوكيل)، الفاعل (المُنفّذ الحقيقي للمهمة)، الهدف (لتحديد آلية تحقيق دور معين)، السماحية (الحقوق المتّصلة بالدور)، الموارد (إدارة الموارد الفيزيائية)، النشاط (المهام الداخلية المحققة من قِبل الوكيل لإنجاز أهدافه)، الخطة (طريقة تحقيق الهدف)، والمهمّة (هدف مُحدد للوكيل).
2. لنمذجة وكيلين: يتم التركيز على المبادئ التنظيمية والاجتماعية لتوصيف الاتصال بين الوكيلين وهي مبادئ الاعتماد (لتوصيف العلاقات بين الوكلاء)، وبروتوكول الاتصال.
3. لنمذجة عدة وكلاء: يتم التركيز على المبادئ التنظيمية والاجتماعية لتوصيف مجتمع الوكلاء وهي مبادئ المجموعة (مجموعة وكلاء)، بنية المجموعة (لتوصيف آلية تعريف المجموعة)، الهدف العام (هدف كبير تُحققه مجموعة من الوكلاء)، المنظمة (المجموعة الكلية التي تحوي كل الوكلاء) [Bergenti et al, 2004].

إن الطبيعة الخاصة للوكلاء (القدرة على الانتقال، المرونة، القدرة على الانتظام في مجموعات) تَعِد بتقديم حلول أفضل لعمليات التطوير البرمجي كونها تعتمد على إيجاد طريقة جديدة كلياً للتفكير في أثناء مراحل التطوير المختلفة، ويتميّز هذا المنهج عن غيره من مناهج تطوير البرمجيات باعتماده على مبدأين أساسيين وهما:

1. إن التجسيد الفعلي لأفكار الوكلاء يمكن أن يُحسّن من قدرات البرامج الناتجة ويزيد من إمكاناتها. إذ يمكن للأنظمة المكوّنة من عدة وكلاء مستقلين أن تزيد من تعقيدها وجودتها.
2. أن القدرات التعبيرية الكبيرة للوكلاء تُبشّر بحلول أكثر فعالية وموثوقية للإجراءات المُعقّدة [Hoa Dam, 2003].

3- المنهجيات القياسية لتطوير الأنظمة الحاسوبية باستخدام مبادئ هندسة البرمجيات المُقادة بالوكلاء

تم تطوير العديد من المنهجيات في مجال هندسة البرمجيات المُقادة بالوكلاء للاستفادة من ميزات الوكلاء وإضافاتها إلى مراحل التطوير المختلفة من دورة الحياة البرمجية بهدف بناء منهج تطوير متكامل لبناء البرامج أو بهدف إضافة تحسينات جديدة إلى مراحل التطوير لزيادة الفعالية والأداء. حيث قامت كل منهجية من المنهجيات القياسية بالتركيز على بعض مراحل التطوير من دورة الحياة البرمجية لتحسّنها لكي تخدم بعض الأهداف الخاصة في عملية التطوير، مما ساعد على تكوين طيف كبير من المنهجيات المتخصصة والمتمايزة عن بعضها بحسب أماكن قوتها ومجالات تطبيقها. فأتت بعض هذه المنهجيات كتوسيع لمبادئ هندسة البرمجيات المُقادة بالوكلاء في حين أتى بعضها الآخر كتخصيص لها [Henderson-Sellers& Gorton, 2002].

هناك عدد كبير من المنهجيات القياسية لتطوير البرامج اعتماداً على مبادئ هندسة البرمجيات المُقادة بالوكلاء، وسنكتفي بذكر أهمّها:

• المنهجية تروبوس (Tropos): وتتميز هذه المنهجية بكونها نِتاج خبرات عدد كبير من الباحثين في عدد من جامعات كندا وإيطاليا. وهي تقوم على التركيز على تحقيق المبادئ الأساسية الخاصة بتطوير الوكلاء مثل الأهداف والأدوار، فضلاً عن تركيزها على الجمع المبكّر للمتطلبات بهدف تحقيق أهداف العناصر المعنية بالنظام (Stakeholder) بشكل كامل ودون أي لبس[Hoa Dam, 2003] [Troposproject.org, 2010] .

تقوم هذه المنهجية على أربع مراحل أساسية هي:

1. تحليل المبكّر للمتطلّبات (Early Requirements Analysis): ويتم التركيز في هذه المرحلة على مقاصد العناصر المعنية بالنظام. حيث يتم تمثيل هذه المقاصد على شكل أهداف، ستقود هذه الأهداف المبدئية إلى المتطلبات الوظيفية وغير الوظيفية للنظام الهدف.
2. تحليل المتطلّبات المتأخّر (Late Requirements Analysis): في هذه المرحلة سيتم تمثيل النظام بفاعل (Actor) واحد أو أكثر موجودين ضمن نموذج اعتمادية استراتيجي، وذلك بهدف تحقيق أهداف العناصر المعنية بالنظام، برفقة عدد من الفاعلين من البيئة التشغيلية. الهدف من هذه المرحلة هو تحديد الارتباطات بين عناصر النظام المختلفة و فاعلية الأساسيين.
3. أسلوب البناء خلال مرحلة التصميم (Architectural Design): في هذه المرحلة سيتم توصيف كيف ستعمل أجزاء النظام مع بعضها. في المنهجية تروبوس يتم تعريف أساليب لبناء المنظّمة فيما يتعلّق بالتطبيقات التعاونية، والديناميكية، والموزّعة وذلك بهدف تنظيم عملية تصميم أسلوب بناء النظام.
4. التصميم التفصيلي (Detailed Design): وتقدم هذه المرحلة معلومات إضافية لكل مكوّن بنائي في النظام. إذ يتم تحدد كيفية تحقيق كل الأهداف المسندة إلى الفاعلين عبر الوكلاء بطريقة أنماط التصميم (Design patterns). كما أن التصميم التفصيلي يتضمّن توصيف طرق التفاعل بين الوكلاء وسلوكهم [Bergenti et al, 2004].

• المنهجية هندسة الأنظمة متعددة الوكلاء (Multi-agent Systems Engineering Methodology - MaSE): وهي مشابهة للمنهجية جايا فيما يتعلق بالتركيز على العمومية ودعم مجالات عمل البرنامج النهائي، لكنها كذلك تركز على مبادئ توليد الرماز النهائي (code) للبرنامج بشكل أوتوماتيكي من خلال أدوات خاصة بهذه المنهجية

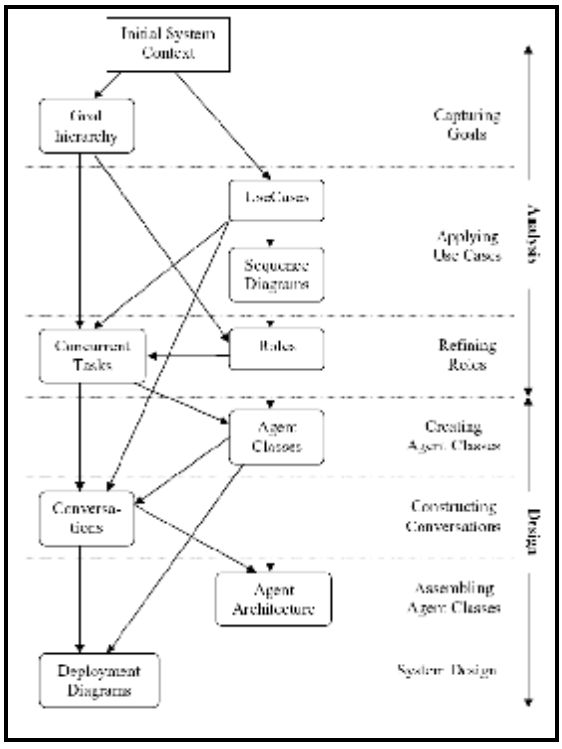
بهدف تقديم منهج متكامل وثابت لتطوير برمجيات الوكلاء وبشكل عـالي الجـودة والأداء [Tveit, 2001]. تركز هذه المنهجية على مرحلتي التحليل والتصميم (مثل المنهجية جايا) لكنها تتميز بكـون هذه المراحل أكثر تفصيلاً ووضوحاً فضلاً عـن تضمن مبادئ النمذجه باستعمال اللغـة الموحـدة للنمذجـه (Unified Modeling Language UML) ضمناه. تتألف منهجية هندسـة الأنظمـة متعددة الوكلاء من سبع خطوات موزّعـة علـى مرحلتين أساسيتين كما يأتي:

1. مرحلة التحليل: وتتألف من 3 خطوات هي:

a. تحديد الأهداف (Capturing Goals): وتتضمن هرمية الأهداف (Goal Hierarchy).
b. تطبيـق حـالات الاسـتخدام (Applying Use Cases): وتتضمن مرحلة تحديد حـالات العمـل (Use Cases) والمخططات التتـالي (Sequence Diagrams).
c. تنقيـة الأدوار (Refining Roles): وتتـضمن تحديد المهام المتزامنـة (Concurrent Tasks)، ونموذج الأدوار (Role Model).

2. مرحلة التصميم: وتتألف من 4 خطوات هي:

a. بنـاء صـفوف الـوكلاء (Creating Agent Classes): وتتضمن بنـاء مخططـات صـفوف الوكلاء (Agent Class Diagrams).
b. بنـاء المحادثـات (Constructing Conversations): وتتـضمن بنـاء مخططـات المحادثات (Conversation Diagrams).
c. تجميع صفوف الـوكلاء (Assembling Agent Classes): وتتضمن بناء مخططات أسلوب بنـاء الوكلاء (Agent Architecture Diagrams).
d. تصميم النظـام (System Design): وتتـضمن مخططـات النـشر (Deployment Diagrams) [Macr.cis.ksu.edu,2010].



الشكل 2 - مراحل العمل في منهجية هندسة الأنظمة متعددة الوكلاء

تعدُّ منهجية هندسة الأنظمة متعددة الوكلاء تخصيصاً لمنهجيات هندسة البرمجيات التقليدية. وإحـدى أهـم ميزاتها هي قدرتها على اقتفاء أثر التغييـرات خـلال مراحل العمل، بحيث يمكن اقتفـاء أثـر أي غـرض خلال مرحلتي التحليل والتـصميم للأمـام والخلـف [Bergenti et al, 2004].

4- مكاملة التحليل المبكر للمتطلبات مـع منهجيـة هندسة الأنظمة متعددة الوكلاء

بناءً على الدراسة النظرية التي قمنا بها على عدد من المنهجيات الأبرز و الأكثر تداولاً و الأقوى تأسيـساً في مجال هندسة البرمجيات المُقادة بالوكلاء، اخترنـا منهجية هندسة الأنظمـة متعـددة الـوكلاء MaSE بوصفها أكثر البرمجيات نضوجاً ومناسـبةً لتطـوير الأنظمة متعددة الوكلاء، كما أنها الأقرب ببنيتها إلـى

منهجيات هندسة البرمجيات القياسية بوصفها تستعمل مخططات معروفة وقياسية.

أما من الناحية العملية فقد اعتمدنا على بيئة التطـوير Agent Tool 3 [Garcia- Ojeda & De Loach & Robby, 2009] [Agenttool.cis.ksu.edu,2010] وعلى المخططات المدعومة منها بحيث اعتبرناها بيئة التطوير الأكثر نضوجاً وخاصة مع تعدد مخططاتهـا وإمكانياتها التحليلية. كما اعتمدنا في عملنا على دعـم هذه البيئة بالبرامج والمكتبـات التـي سـتزيد مـن إمكانياتها وستساعدنا على مكاملة عدد مـن المبـادئ الجديدة التي اقترحناها معها.

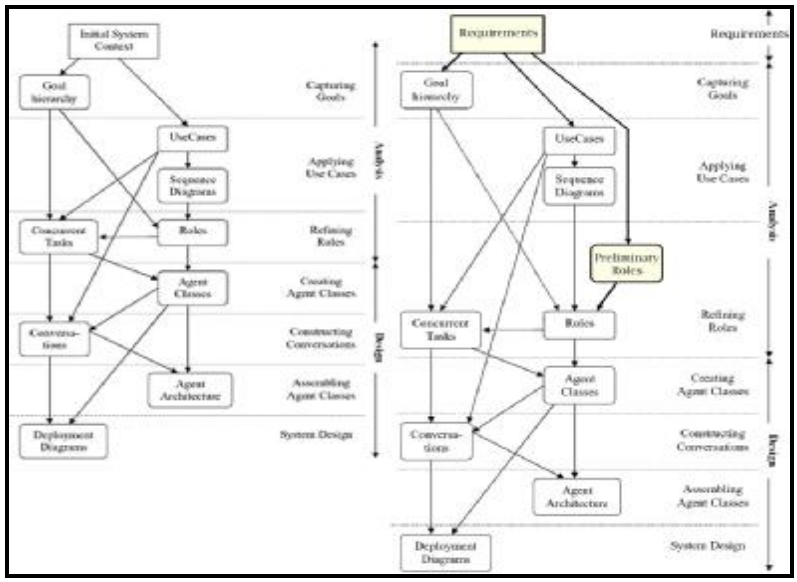
يقوم الجزء الأكبر من بحثنا على دعـم هـذه المنهجيـة وإغنائها عبر إضافة عدد من المراحل الرّديفة إليها والتي ستساعد على زيادة قدرتها على التفاعـل والتـأقلم مـع متطلبات عملية التطوير كلها الخاصة بالأنظمة متعـددة الوكلاء.

نركز في هذه الورقة على إضافة مرحلة للتوصـيف المبكر للمتطلبات وما يتبع ذلك من تعديل على مخطط مراحل MaSE. من جهة أخرى فقـد قمنـا بإضـافة مخطط جديد إلى خطوة تنقية الأدوار وهو المخطـط المبدئي للأدوار (Preliminary Roles). التعـديلات المقترحة موضّحة كلها بالشكل 3.

4-1 إضافة مبادئ التحليل المبكّر للمتطلّبـات إلـى منهجية هندسة الأنظمة متعددة الوكلاء

تعاني منهجية هندسة الأنظمة متعددة الوكلاء من قصور في مرحلة جمع المتطلبات وتحليلهـا، إذ عـدت هـذه المرحلة خارج مراحل العمل الخاصة بهذه المنهجية.

لاحظنا في أثناء دراستنا للمنهجية وجود عدد كبير مـن المخططات التي قد يحتاجها المحللون والمطوّرون للنظام متعدد الوكلاء في أثناء عملهم، لكننا لاحظنا كذلك عـدم وجود أي مخطط يُعنى بالمتطلبات أو يربط بين المخطط الأساسية للنظام (مخطط الوكلاء، الأهـداف والأدوار). كما أنه وفي حال وجود أي تعديل على متطلبات النظـام يجب علينا تعديل كل مخطط على حدة وبشكل منفـصل. لذلك فقد قمنا في بحثنا بإضافة مرحلة تحليل المتطلبـات إلى مراحل العمل الخاصة بالمنهجية، كما اقترحنا تقـسيم المراحل إلى ثلاثة أجزاء بحيث يُعنـى الجـزء الجديـد بمرحلة جمع المتطلبات وتحليلها.



الشكل 3 -مقارنة بين مخطط العمل الأصلي لمنهجية هندسة الأنظمة متعددة الوكلاء و المخطط المعدّل بناءً على اقتراحنا

أما من الناحية العملية فقد رأينا أنه من الأفضل الاعتماد على المخططات الخاصة بالتحليل المبكّر للمتطلبات المُستعمل في منهجية تروبوس بدلاً من الإتيان بمخططات وترميزات جديدة، ولاسيَّما أن هذه المخططات من أكثر المخططات فعالية في منهجية تروبوس التي تتميز بقوتها الكبيرة في مرحلة التحليل.

قمنا باختيار بيئة التطوير *SI [Sesa.dit.unitn.it,2010] التي تعدُّ واحدة من أكثر بيئات العمل الداعمة للمنهجية تروبوس تطوراً. وقمنا بدراسة مخططات التحليل المبكر فيها والمخططات الرئيسية في بيئة التطوير AgentTool 3 (مخطط الوكلاء والأهداف والأدوار) بهدف بناء مجموعة من قواعد التحويل لنقل المعلومات من مخطط التحليل المبكر في بيئة التطوير *SI إلى مخططات الوكلاء والأهداف والأدوار في بيئة التطوير AgentTool 3.

قمنا بعد ذلك ببناء قواعد التحويل من مخطط تحليل المتطلبات المبكّرة ضمن *SI إلى المخططات الثلاثة الرئيسية ضمن بيئة التطوير. و من ثم طوّرنا برنامجاً خاصاً بعملية التحويل باستخدام القواعد تلك.

4-2 قواعد التحويل و الأداة المستخدمة

سنتحدث هنا عن القواعد التي طورناها واللازمة للتحويل من مخطط تحليل المتطلبات المبكّرة إلى المخططات الثلاثة الرئيسية ضمن بيئة التطوير AgentTool 3 (مخطط الوكلاء والأهداف والأدوار).

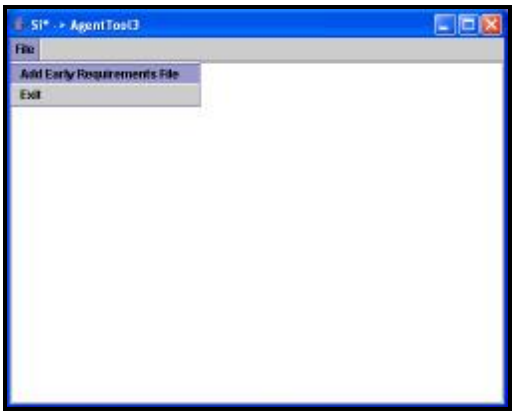
ستقوم بتوزيع قواعد التحويل إلى 4 فئات، ونكتفي هنا بمثال عن كل فئة لتجنب الإطالة:

1. تحديد قواعد النقل المباشر: وسيتم فيها نقل المفاهيم بشكل كامل من مخطط المتطلبات إلى المخططات الثلاثة. ومثال ذلك مقابلة كل وكيل في بيئة التطوير *SI بوكيل مقابل له في مخططات الوكلاء والأهداف والأدوار في بيئة AgentTool 3.
2. تحديد قواعد النقل الجزئي: وسيتم فيها نقل المفاهيم بشكل جزئي من مخطط تحليل المتطلبات المبكّرة إلى المخططات الثلاثة. ومثال ذلك مقابلة كل علاقة "يزوّد (Provides)" (فقط من الوكلاء إلى المهام) في مخطط المتطلبات في بيئة التطوير*SI بعلاقة "يزوّد (Provide)" مقابلة لها في مخطط الوكلاء والأدوار. كما سيتم مقابلة كل علاقة "يزوّد (Provides)" (فقط من الأدوار إلى المهام) في المخطط *SI بعلاقة "يزوّد (Provide)" مقابلة لها في مخطط الوكلاء.
3. تحديد قواعد النقل الاعتباري: وسيتم فيها نقل المفاهيم بشكل كامل من مخطط تحليل المتطلبات المبكّرة إلى المخططات الثلاثة، ولكن ضمن اعتبارات محددة وشروط مسبقة تحدد صلاحية وسلامته. ومثال ذلك مقابلة كل مورد (Resource) في مخطط المتطلبات في بيئة التطوير*SI بإمكانية (Capability) مقابلة له في مخططات الوكلاء و الأدوار.
4. تحديد قواعد النقل غير القابل للتمثيل: وسيتم فيها نقل المفاهيم التي ليس لها تمثيل ضمن المخططات الثلاثة إلى صيغة داعمة غير بيانية مثل: قمنا بتحويل العلاقات الموجودة ضمن مخطط تحليل المتطلبات والتي ليس لها مقابل ضمن المخططات الثلاثة الرئيسية إلى ملاحظات تكتب في ملف منفصل.

بعد تعريف القواعد المطلوبة جميعها في عملية النقل هذه، قمنا بتطوير برنامج خاص بعملية التحويل من

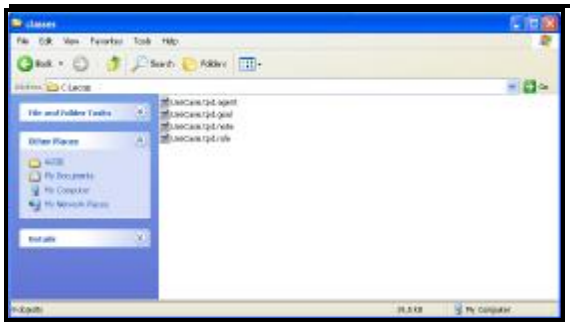
مخطط تحليل المتطلبات المبكّرة في بيئة التطوير *SI إلى المخططات الثلاثة الرئيسية ضمن بيئة التطوير النهائية في بيئة التطوير AgentTool3 باستخدام القواعد السابقة. حيث سيقوم هذا البرنامج بقراءة ملف الـ XML الخاص بمخطط تحليل المتطلبات وتحليل مكوناته ثم يقوم بترجمتها إلى مكوّنات وعلاقات بين المكونات الناتجة، وفي النهاية يقوم بتوليد ملفات الـ XML الخاصة بمخططات الوكلاء، الأهداف والأدوار، كما أنه سيولّد ملفاً نصياً يحتوي على الملاحظات الإضافية بعد عملية التحويل. فيما يلي بعض التفاصيل الخاصة باستخدام هذا البرنامج.

1. من القائمة File نختار Add Early Requirements File (الشكل 4).



الشكل 4- اختيار مخطط التحليل المبكّر

2. يتم فيما بعد تحديد ملف المتطلبات (المصدر) ثم يقوم البرنامج بقراءة ملف المتطلبات و بناء المخططات الثلاثة الناتجة (مخطط الوكلاء، الأهداف والأدوار) في ملفات جديدة كل حسب لاحقته (agent لمخطط الوكلاء، goal للأهداف و role للأدوار). أمّا الملاحظات فيتم توليد ملف خاص بها (باللاحقة note) (الشكل 5).



الشكل 5- الملفات الناتجة

تم اختبار قواعد التحويل والأداة البرمجية على العديد من الأمثلة للتأكد من صحتها وسلامة المخططات الناتجة. إذ قمنا بالتأكد من صحة كل المخططات الناتجة في كل الأمثلة واتساقها مع المتطلبات الأساسية.

تم تقييم جميع المخططات الناتجة بشكل شخصي (اعتماداً على محاولة رسم مخططات النظام دون الاعتماد على البرنامج للتأكد من صحة المخططات) كما اعتمدنا على كمية جيدة من الأمثلة للتأكد من البرنامج. يبقى الاستعمال المتعدد للأداة البرمجية مع الزمن أفضل للتأكد من سلامته و دقته.

5- دراسة حالة (مسألة نظام إدارة المؤتمرات)

سنقود بدراسة نظام إدارة المؤتمرات كمثال على آلية الاستفادة من مكاملة التحليل المبكر للمتطلبات مع منهجية هندسة الأنظمة متعددة الوكلاء. إذ سنستخدم هذا النظام لتوضيح دور الأداة التي قمنا بتطويرها و فعاليتها و لإظهار آلية تطبيق قواعد التحويل التي عرفناها. إذ إننا سننطلق من تعريف مخطط المتطلبات للنظام و سنوضّح دور الأداة في توليد المخططات الثلاثة المطلوبة.

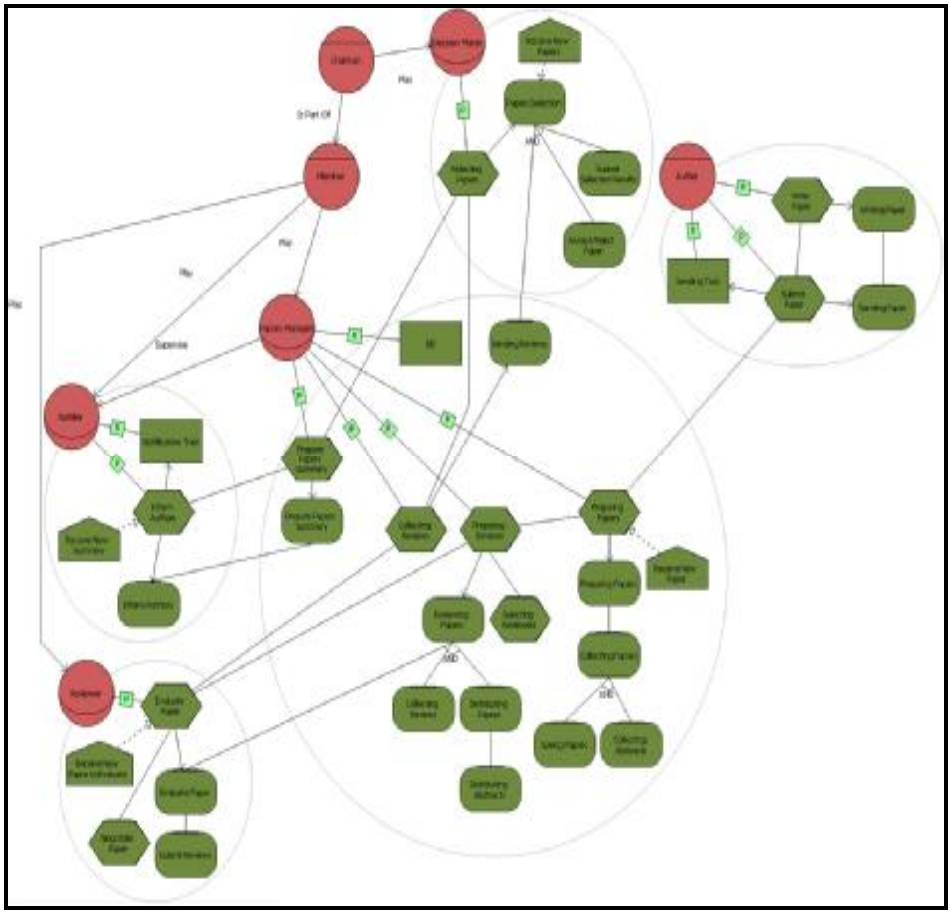
تم اختيار نظام إدارة المؤتمرات بوصفه نظاماً متوسّط الحجم ويتطلّب تعاوناً وتنسيقاً بين عدد كبير من الأفراد والجماعات. وهو نظام خاص باستلام أوراق البحث المقدمة للمؤتمر من مؤلفيها وتقييمها من عدة مدققين اعتماداً على عدد من المعايير المعرفة

والمحددة بشكل مسبق، ومن ثم يتم اختيار الأوراق المقبولة من قبل رئيس النظام ليتم اعتمادها ضمن المؤتمر. سنستخدم هذا النظام لتوضيح دور الأداة التي قمنا بتطويرها وفعاليتها ولإظهار آلية تطبيق قواعد التحويل التي عرفناها. حيث سننطلق من تعريف مخطط المتطلبات للنظام وسنوضّح دور الأداة في توليد المخططات الثلاثة المطلوبة.

نظام إدارة المؤتمرات يُعدُّ منظّمة وأعضاؤها (مدققون، متّخذو قرار، جامعو المراجعات...) والذين قد تتغيّر أدوارهم في أثناء عمل النظام (مثال: قد يتحوّل أحد المدققين إلى متّخذ قرار بعد مدة). تتعامل هذه المنظّمة وبشكل رئيسي مع منظمة الكتاب (المؤلّفين) التي تعدُّ بدورها موردّة لها على اعتبار أن المؤلفين يكتبون المؤلفات ويرسلونها إلى منظمة إدارة المؤتمرات لِتُدرس وُتدَقَق وتُقَيم.

نقوم في بداية الأمر برسم مخطط المتطلبات للنظام (الشكل 6) للنظام المطلوب الذي يتكون بشكل أساسي من العناصر الآتية:

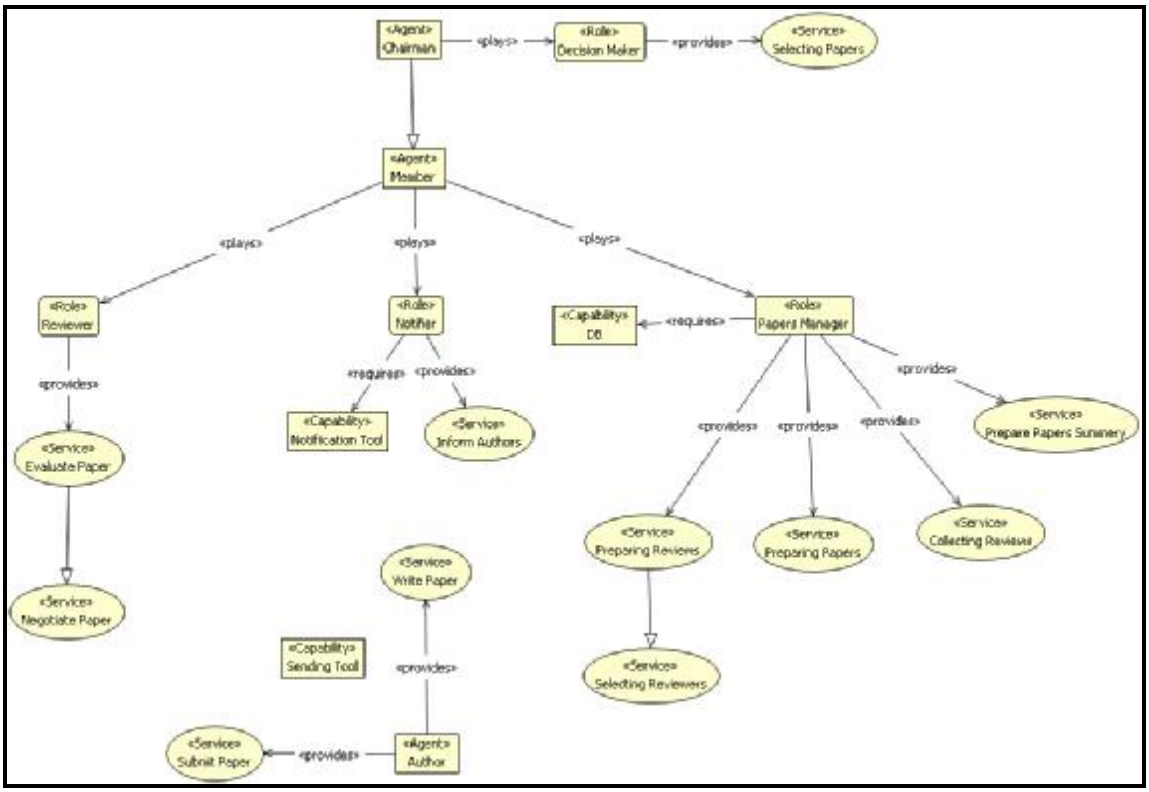
1. الوكلاء: الكتاب (Authors)، الأعضاء في إدارة المؤتمرات (Members) و رؤساء المجالس (Chairmen) (و هم متخذو القرار).
2. الأدوار: الكتاب (Authors) (استعملنا الوكيل السابق)، المسؤولون عن الأوراق (Papers Managers)، المراسلون (Notifies)، المدققون (Reviewers) ومتخذو القرار (Decision Makers).



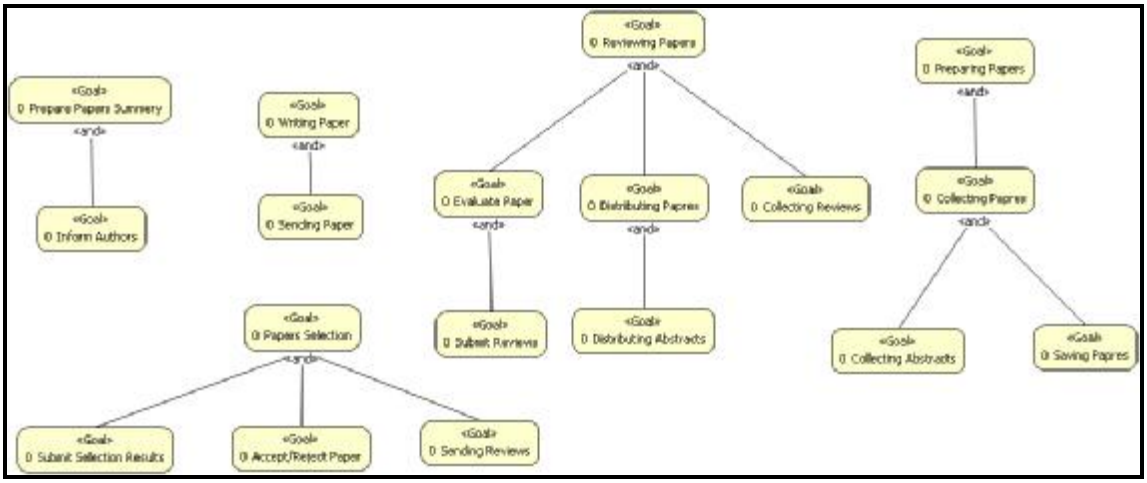
الشكل 6 مخطط المتطلبات لنظام إدارة المؤتمرات

يمثل الشكل 6 مخطط المتطلبات لنظام إدارة المؤتمرات الذي قمنا برسمه ضمن بيئة (*SI) ويعبّر عن جميع العلاقات والأدوار الموجودة في النظام.

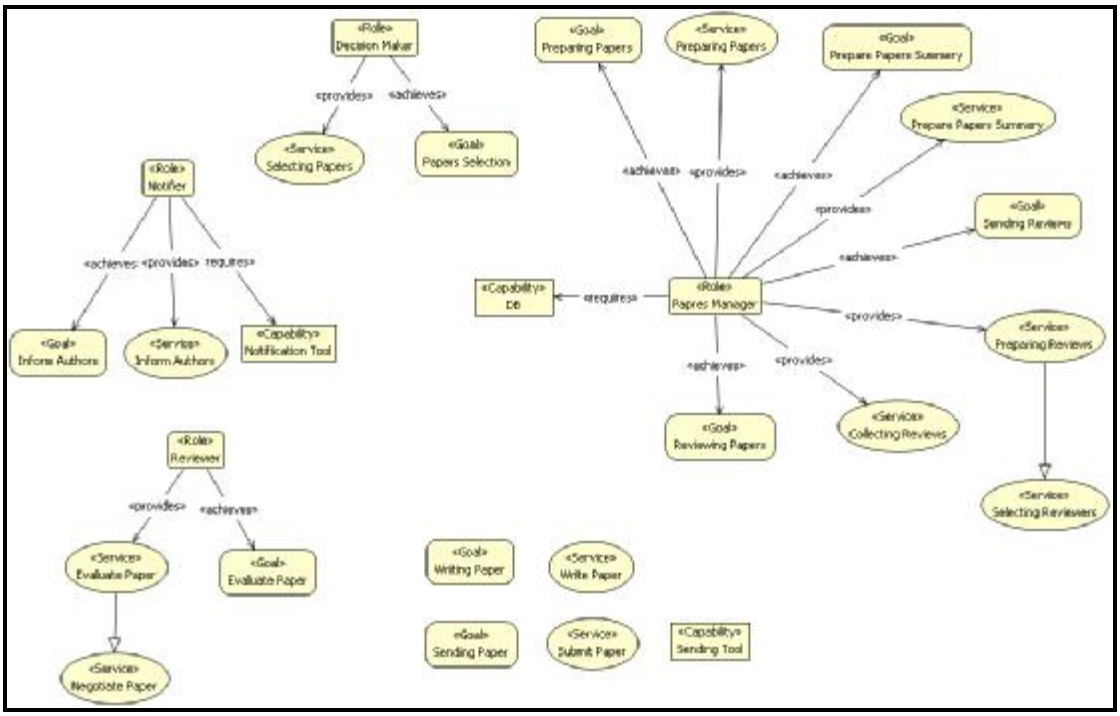
بعد ذلك نقوم بتشغيل البرنامج الذي طوّرناه بهدف توليد المخططات الثلاثة المطلوبة (مخطط الوكلاء، الأهداف والأدوار) مع إدخال مخطط متطلبات النظام الذي قمنا ببنائه مسبقاً كدخل له، ليقوم البرنامج بتوليد المخططات الثلاثة (الأشكال 7 و 8 و 9).



الشكل 7 مخطط الوكلاء لنظام إدارة المؤتمرات



الشكل 8 مخطط الأهداف لنظام إدارة المؤتمرات



الشكل 9 مخطط الأدوار لنظام إدارة المؤتمرات

الخاتمة والنظرة المستقبلية إلى تطور هذا الاتجاه:

قمنا في هذا المقال باستعراض مفاهيم الوكيل البرمجي وتقديم صورة مختصرة عن هندسة البرمجيات المُقادة بالوكلاء بوصفها اتجاهاً جديداً من اتجاهات تطوير البرمجيات الحاسوبية، كما حاولنا عرض نقاط قوته وتحديد المواقع الفضلى لاستخدامه كطريقة أساسية للتطوير. كما تطرّقنا إلى المنهجيات القياسية فيه وتحدثنا عن عدد من بيئات التطوير الداعمة له. وقمنا أيضاً بعرض آلية لدمج تقنيات التحليل المبكّر للمتطلبات مع منهجية هندسة الأنظمة متعددة الوكلاء، وقمنا ببناء برنامج خاص بعملية التحويل من مخطط تحليل المتطلبات المبكّرة في بيئة التطوير *SI إلى المخططات الثلاثة الرئيسية ضمن بيئة التطوير النهائية في بيئة التطوير AgentTool3.

أمَّا النظرة المستقبلية إلى تطور هذا الاتجاه فهي متفائلة إلى حد كبير، لأنّ الكثير من الباحثين يعقدون آمالاً كبيرة عليه وخاصة لفعاليته الكبيرة في نمذجة الأنظمة الموزعة والمعقّدة، إذ يعدّونه الاتجاه القياسي لتطوير مثل هذه الأنظمة، ويعتقدون أنه سيتطور في المدة القصيرة القادمة ليكون المنهج القياسي. التطويرات المستقبلية الممكنة كثيرة ونخص منها المجالات:

1. تضمين ومعالجة الويب الدلالي (Semantic Web) و ذلك بهدف بناء شبكة من الوكلاء القادرة على التفاعل مع بعضها بهدف تحقيق مهام معقدة على الويب.
2. تطوير الأنظمة المفتوحة التي تتميز بقدرتها على السماح لأي وكيل بالمشاركة بمهامها ولكن بشرط تحقيق الوكيل لقواعد العمل في النظام.

المراجع

1- http://agenttool.cis.ksu.edu/, last update 01 October 2009.

2- Bergenti Federico, Marie-Pierre Gleizes, and Franco Zambonelli, METHODOLOGIES AND SOFTWARE ENGINEERING FOR AGENT SYSTEMS, 2004.

3- http://macr.cis.ksu.edu/ last update Jan 2010.

4- Garcia-Ojeda Juan C., Scott A. De Loach and Robby, agent Tool III: From Process Definition to Code Generation, Kansas State University 2009.

5- Henderson-Sellers Brian and Ian Gorton، Agent-Based Software Development Methodologies, Workshop on Agent-oriented Methodologies at OOPSLA 2002، Seattle, USA, November 2002.

6- Hoa Dam Khanh. Evaluating And Comparing Agent-Oriented Software Engineering Methodologies, RMIT University– Melbourne, Australia, Master thesis, 2003.

7- http://sesa.dit.unitn.it/sistar_tool/home.php?7, last update 2009.

8- http://www.troposproject.org/ last update Jan 2010.

9- Tveit Amund. A Survey Of Agent-Oriented Software Engineering, NTNU Computer Science Graduate Student Conference, Trondheim, Norway - May 2001.

تاريخ ورود البحث إلى مجلة جامعة دمشق 2010/3/1.