Çok-Etmenli Yazılım Sistemleri için Yürütülen Modelleme Dili Çalışmaları ve Bunların Anlamsal Web Desteği Perspektifinde Değerlendirilmesi

Geylani Kardaş[1], Oğuz Dikenelli[2]

[1] Ege Üniversitesi, Uluslararası Bilgisayar Enstitüsü,
35100 Bornova, İzmir, Türkiye
geylani.kardas@ege.edu.tr
[2] Ege Üniversitesi, Bilgisayar Mühendisliği Bölümü,
35100 Bornova, İzmir, Türkiye
oguz.dikenelli@ege.edu.tr

Özet. Bu bildiride çok etmenli sistemlerin geliştirilmesi amacıyla üzerinde çalışılmakta olan yazılım modelleme dilleri, tanımladıkları yapılar ve sundukları modelleme uzantılarına yönelik bir inceleme yer almaktadır. Etmen tabanlı yazılım geliştirmeye yönelik çeşitli araştırma gruplarınca sunulan model dillerinden henüz hiçbiri bir standart olarak kabul edilmemiştir ve biçimsel (formal) yapılarının geliştirilmesine devam edilmektedir. Diğer taraftan, biz de birçok yazılım etmeni araştırmacısı gibi yazılım etmenlerinin gelecek nesil bilgisayar ağı olarak adlandırılan Anlamsal Web ortamında vazgeçilmez bir rol oynayacağına inanmakta ve etmen modelleme dil(ler)inin anlamsal web yapılarını da bünyelerine dahil etmeleri gerektiğini düşünmekteyiz. Bu nedenle, yürütülen çalışmaların bu bakış açısından değerlendirilmesi ve bizim anlamsal web ortamında çalışacak etmen sistemlerinin hazırlanmasına yönelik modelleme yaklaşımımız da bu bildiri de yer almaktadır.

1 Giriş

Yazılım etmenleri (software agents) ve bunların oluşturduğu çok-etmenli sistemler (multi-agent systems), bilgi ve iletişim teknolojisi iş senaryolarının çok çeşitli alanlarda uygulanması sırasında karşılaşılan zorlukların giderilmesini sağlayan güçlü bir teknoloji olarak ortaya çıkmışlardır. [1]'de algılayıcıları yardımıyla ortamı algılayan ve etkileyicileri yardımıyla bu ortamı etkileyen bir sistem olarak tanıtılan etmenleri, aynı zamanda otonom yazılım ortamlarını da göz önünde bulundurduğumuzda kullanıcısının adına bir takım görevleri yerine getirmek üzere davranma yeteneği olan yazılım bileşenleri olarak kabul etmek yerinde olacaktır. Tek bir etmenin yalnız başına kendi bilgi ve bireysel yeteneklerini kullanarak çözemediği veya etkin bir biçimde çözemeyeceğini düşündüğü problemleri birbiriyle işbirliği yaparak eşgüdümlü bir biçimde çözmek için bir araya gelen etmenlerin oluşturduğu sistemler de çok-etmenli sistemler adını almaktadır [2].

Daha şimdiden bir çok endüstriyel deneyim, etmenlerin üretim sürecinde, web servislerinde ve dağıtık ağ yönetiminde fayda sağladığını ispatlamaktadır. Bunlara ek

olarak geleceğin bilgi sistemlerinde yer alması düşünülen yayılmış hesaplama (pervasive computing), ızgara hesaplama (grid computing) ve anlamsal web (semantic web) gibi teknolojilerin hayata geçirilmesinde etmenlerin ve çok-etmenli sistemlerin kullanılmasını öneren akademik araştırma ve çalışmalar devam etmektedir [3].

Etmen tabanlı hesaplama (agent-based computing); uygulama tasarımı ve geliştirmeyi, bir ortamda yer alan ve birbirleri ile yüksek seviyeli protokoller ve diller aracılığı ile etkileşimde bulunan otonom yazılım varlıklarının (etmen) geliştirilmesi olarak ele almaktadır. Son birkaç yılda etmen tabanlı hesaplamanın yeni bir yazılım mühendisliği paradigması olarak kabul görmesiyle birlikte karmaşık yazılım sistemlerinin çok-etmenli sistemler olarak geliştirilmesini destekleyecek uygun model ve tekniklerin belirlenmesi ve tanımlanmasına yönelik yoğun araştırmalar yapılmaktadır [4]. *Etmen Yönelimli Yazılım Mühendisliği* adı altında toplanabilecek bu çalışmalar özellikle etmen-yönelimli paradigmaya uyan oldukça çeşitli yeni formal modelleme yaklaşımları, geliştirme yöntemleri ve modelleme tekniklerini önermektedirler. Ancak [3]'te de belirtildiği gibi bu alana yönelik araştırma henüz çok yeni ve başlangıç aşamasındadır ve etmen yönelimli yazılım mühendisliğinin vaat ettiklerini yerine getirmesi, geniş kapsamda kabulü ve kompleks yazılım sistemlerinin geliştirilmesinde pratik bir şekilde kullanılabilir bir paradigma olması için etmen sistemlerinin geliştirilmesine yönelik çeşitli engellerin aşılması gerekmektedir.

Çok-etmenli yazılım sistemlerinin tasarımı ve hayata geçirilmesi sırasında bir metodolojinin izlenmesi gerekmektedir. Bir yazılım metodolojisinin bir *modelleme dili* ve bir *yazılım süreci* ile karakterize edilmesini [5] göz önünde bulundurduğumuzda süreci oluşturan aktivitelerden önce etmen sistemine ait elemanların özel bir notasyon ve onunla ilişkili anlamlarla (semantics) tanımlanarak ortaya konması sağlanmalıdır. İlgili formal gösterimlerin yazılım modelleme dilleri aracılığıyla yapılması gerektiği açıktır.

Bu bildiride çok-etmenli sistemlerin modellenmesi amacıyla üzerinde çalışılmakta olan çeşitli model dilleri, etmen tasarlamaya yönelik tanımladıkları yapılar ve sundukları modelleme uzantıları da göz önüne alınarak incelenmiştir. Şu an için çok-etmenli yazılım sistemi modellemeye yönelik hiçbir model dili standart olarak kabul edilmemiştir. Bunun nedeni olarak yazılım etmenlerinin herkes tarafından kabul görecek yapılarının henüz ortaya çıkmamış olması, araştırma alanının oldukça aktif olması ve farklı gruplar tarafından farklı etmen bakış açılarının savunulması gösterilebilir. Çok-etmenli sistemlerde yer alan etmenlerin işbirliği yapabilmeleri ve birbiriyle iletişimde bulunabilmesi amacıyla standartlaşma çalışmaları yürütmekte olan FIPA'nın (Foundation for Intelligent Physical Agents) [6] bile henüz tam anlamıyla ortaya koyduğu bir modelleme dili yoktur. Bildirinin bir sonraki bölümünde değinileceği gibi sadece başlangıç seviyesinde kabul edilecek bir çalışması vardır.

Bildiride ilk olarak şu an için literatürde adları geçen belli başlı etmen modelleme dillerine ait inceleme yer almaktadır. İzleyen bölümlerde modelleme dillerinin anlamsal web desteği açısından bir değerlendirmesi ve bizim çok-etmenli sistemlerin geliştirilmesine yönelik savunduğumuz yaklaşım verilmiştir. Son olarak sonuç kısmı yer almaktadır.

2 Çok-Etmenli Sistemlere Yönelik Modelleme Dili Çalışmaları

Her ne kadar etmen tabanlı yazılım geliştirme ayrı bir paradigma olarak kabul görse de günümüzde, etmen sistemlerini hazırlamaya yönelik etmen geliştirme yazılım çatıları (software framework) nesne tabanlı yazılım geliştirme dilleri ile hazırlanmakta; etmen modellemeye yönelik önerilen model dillerinin çoğu da UML'i (Unified Modeling Language) temel almakta ve/veya UML'i uzatan yapılar sunmaktadır.

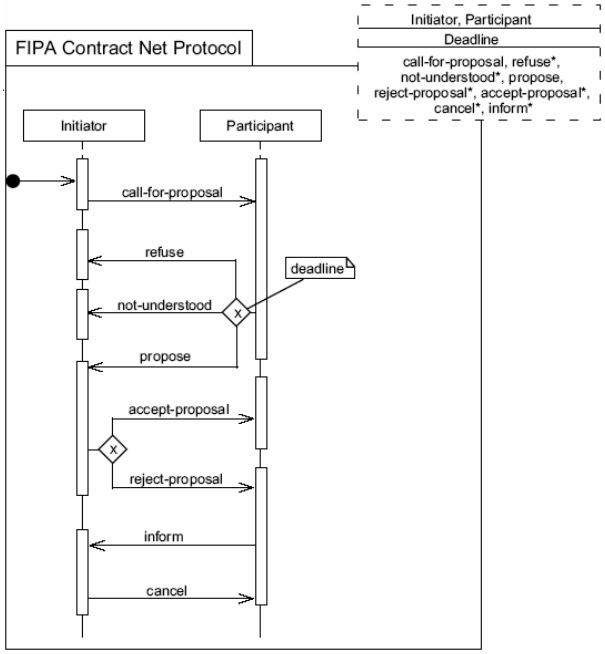
Modelleme alanında üzerinde şu ana kadar en çok çalışılan dillerden biri *Agent UML*'dir [7]. Agent UML, UML'in paket, şablon, sıra diyagramları, etkileşim diyagramları, aktivite diyagramları ve sınıf diyagramlarına etmen temelli uzantılar getirmiştir. UML model semantikleri bir metamodel ile temsil edilmektedir.

Agent UML (AUML), etmen etkileşim protokolleri için üç katmanlı bir gösterim şekli sunmaktadır: Protokol bütünü göstermeye yönelik katman, etmenler arasındaki etkileşimleri gösteren katman ve içsel etmen işleyişini gösteren katman.

AUML, protokol bütününü temsil etmek için UML'in paket ve şablon yapılarını kullanmaktadır. Etmenlerin birbirleri ile olan etkileşimlerini modellemek amacıyla da AUML'de, UML'in sıra (sequence) diyagramlarını, etkileşim diyagramlarını, aktivite diyagramlarını ve durum şemalarını (statechart) uzatan yapılar tanımlanmıştır. Etmen iç işleyişinin modellenmesi için yine aktivite diyagramları ve durum şemalarından yararlanılmaktadır. Bu UML uzantılarına (extensions) ait örnekler yer kısıtından dolayı bu bildiride yer almamaktadır. Ancak [7] ve [8]'de AUML'in sunduğu bu UML uzantılarına ait detaylı bilgi yer almaktadır. Sadece yine [7]'den alınan "FIPA Contract Net" Protokolünün modellenmesine yönelik mesaj sıra diyagramı fikir vermesi amacıyla Şekil 1'de verilmiştir: Haberleşmeyi başlatan etmen (initiator) kendisinden bir iş için teklif (öneri) almak istediği bir diğer etmene (participant) bir teklif isteği (call-for-proposal) mesajı yollar. İçi dolu noktayla iletişimin asenkron gerçekleştiği belirtilmektedir. Kendisinden teklif beklenen etmen belli bir süre içerisinde teklif vermeyi kabul eder veya reddeder. Bir diğer yanıtı da mesajı anlamadığına yönelik olabilir. Yanıt için bir karar mekanizması işlettiği, diyagramda içi "x" işareti ile doldurulan eşkenar dörtgen ile temsil edilmektedir. Teklifi alan "initiator" etmen teklifi kabul edebilir ya da reddedebilir.

Etmen sistemleri için modelleme dilleri üzerine [9]'da yer alan çalışmada ideal bir modelleme dilinin grafiksel ya da metinsel bir notasyona sahip olması, bir semantiğinin olması, soyut yapılar barındırması, farklı bakış açıları içermesi ve belli başlı araçlara (tasarım, kod geliştirme, vb.) sahip olması gerekliliği vurgulanmıştır. Bu özellikler ele alındığında Agent UML'in aşağıdaki dezavantajlara sahip olduğu görülmektedir:

- Görsel notasyon eksiktir ve aslında nesne tabanlı sistemler için önerilen UML 2.0'a çok fazla bağlıdır.
- Diğer geliştiricilerle bilgi alışverişini sağlayacak hiçbir metinsel notasyonu bulunmamaktadır.
- Anlam bilgisi de yine UML 2.0'a dayanmaktadır. Bu nedenle yarı-formaldir.
- Sunduğu spesifikasyonları tanımlama ve kullanma adına hiçbir araç şu an için mevcut değildir.



Şekil 1: AUML'de "FIPA Contract Net" protokolünün gösterimi [7]

Kayda değer bir diğer çok-etmenli sistem modelleme dili çalışmasının FIPA ile OMG'nin (Object Management Group) Etmen Özel İlgi Grubu'nun (Agent SIG) ortaklaşa yürüttükleri *Agent Class Superstructre Metamodel* [10] adı verilen çalışma olduğu söylenebilir. Ancak çalışma, bu grupların kendilerinin de belirttiği gibi başlangıç (preliminary) safhasındadır.

Önerilen metamodel etmenleri, etmen rollerini ve etmen gruplarını ve bunların birbirleri ile olan ilişkilerini ortaya çıkarma gayretindedir [11]. Ortaya çıkan belirtim UML'e dayalıdır ve aynı zamanda onun bir uzantısıdır. Önerilen soyut sözdizimi Şekil 2'de verilmiştir.

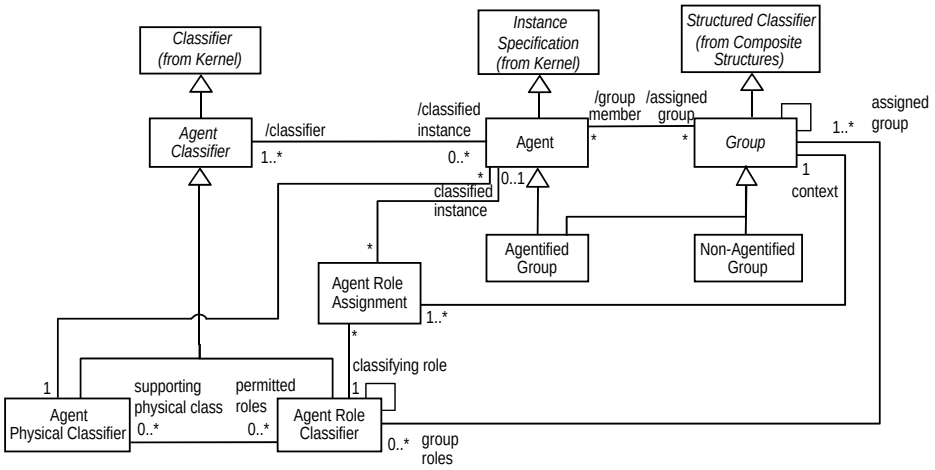
Agent Classifier etmenlerin ne şekilde sınıflandırılacağını tanımlar ve iki altsınıfa sahiptir: *Agent Physical Classifier* ve *Agent Role Classifier*. Agent Physical Classifier bir etmenin çekirdek ihtiyaçlarını karşılayacak temel (ya da ilkel) sınıfları tanımlar. Bu sınıflar daha çok etmen yapılarının hayata geçirildiği fiziksel platform ile alakalıdır. Doğal olarak etmenler de fiziksel platforma bağlı olarak belirli özellikler ve yetenekler kazanırlar. Bu özellikler etmenin yaşamı boyunca devam eder.

Öte yandan *Agent Role Classifier* etmenleri ortamlarda oynayacakları rollere göre sınıflar. Dolaysıyla etmenlerin yeteneklerini ve aktivitelerini içerir. Etmenlerin oynadıkları roller zamanla değişebilir.

Şekil 2'de ortada görünen örnekler (instances) Agent 'dır. *Agent* sınıfı bir sistemin bireylerini oluşturan tüm etmen kümelerini tanımlar. Her bir Agent örneği kendisinin gerekli özelliklerini tanımlayan bir ya da daha fazla Agent Classifier ile ilişkilidir.

Group' lar ise belli bir neden için toplanmış etmen sınıflarını temsil eder. Group içerisinde etmenler kendi rollerine göre birbirleri ile etkileşimde bulunurlar. Bu

nedenle bir Group bir dizi rolle (ya da geçişlilik özelliğini düşünürsek etmenle) tanımlanır. Group'lar bir etmen olarak ya da kendi adlarına etmen gibi davranan yapılar olup olmamalarına göre ikiye ayrılırlar: *Agentified Group* ve *Non-Agentified Group*.



Şekil 2: FIPA Agent Class Superstructure Metamodel'e ait sentaks [11]

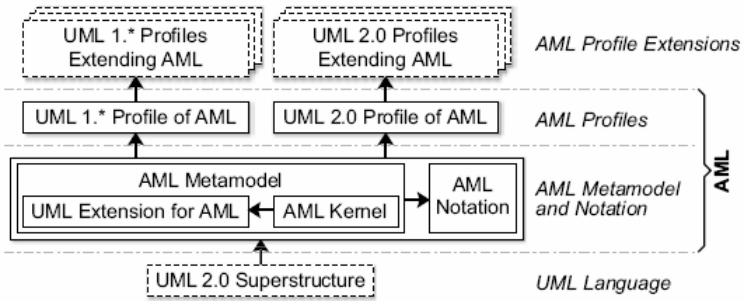
Etmenler ve bunların Agent Role Classifier ve Agent Physical Classifier'ları ile aralarındaki ilişkiyi inceleyecek olursak: Agent ile Agent Physical Classifier arasındaki ilişki etmenin temel yeteneklerini ve ihtiyaçlarını gösterir. Her Agent, bir Agent Physical Classifier ile ilişkilendirilmelidir. Agent ile Agent Role Classifier arasındaki ilişki ise ilgili etmenin ne tür aktivitelerde bulunacağını göstermektedir. Bir etmenin bir Agent Role Classifier ile ilişkilendirilmesine gerek yoktur ama böyle bir etmen bir Group içerisine alınamaz.

Agent Role Assignment, etmenler, roller ve gruplar arasında üçlü bir ilişki olarak tanımlanabilir. Her bir Agent Role Assignment örneği (instance) bir rolü, bir grubu ve bir de etmeni birbiri ile ilişkilendirir. Her bir Agent Role Assignment'in bir rolü ve grubu olması gerekirken her zaman için ilişkilendirilmiş bir etmeni olmayabilir. Bu tip atamalar *position* olarak adlandırılır.

Dikkat edilirse yukarda önerilen metamodel daha önce de belirtildiği gibi UML'i belli açılardan uzatmakta ama özellikle Classifier yapısı ile aynı zamanda UML'i temel almaktadır. Yani bir anlamda UML metamodeli kendi modeline dahil eder. Bu da etmenleri modellerken nesne tabanlı tasarımdan gelen kısıtların (ya da zorlamaların) ortadan kalkmasını sağlamaktadır.

Değinilmesi gereken bir başka etmen modelleme dili ise AML'dir (Agent Modeling Language). Whitestein Technologies şirketi tarafından geliştirilen AML [12] yarı-formal, görsel bir modelleme dilidir. UML 2.0'ın bir uzantısıdır. Şekil 3'te verilen AML yapısı incelendiğinde AML'nin iki ayrı katmanla tanımlandığı görülür: *AML Metamodel and Notation* ve *AML Profiles*.

UML 2.0 Üstyapısı' nın (Superstructure) UML'in soyut sentaks, semantik ve notasyonunu tanımlamasından hareketle *UML Language* katmanı, AML'nin çoklu etmen sistemlerine özel modelleme yapılarını tanımlamasına izin verir.



Şekil 3: AML tanımı katmanları [12]

AML Metamodel and Notation katmanı AML soyut sentaksını, semantiğini ve notasyonunu *AML Kernel* ve *UML Extension for AML* paketleri ile tanımlar. *AML Kernel* paketi AML'nin spesifik modelleme elemanlarının tanımlandığı çekirdektir. Etmen mimarisi, davranışları ve zihinsel bakış açılarına yönelik yapılar bu çekirdekte yer alır. *UML Extension for AML* paketi standart UML elemanları üzerine meta-özellikler ve yapısal kısıtlar ekler.

AML Profiles katmanında, AML Metamodel and Notation katmanı üzerine iki UML profili inşa edilmiştir: UML 1.*'a dayanan *UML 1.* Profile for AML* ve UML 2.0'a dayanan *UML 2.0 Profile for AML*. Bu profiller var olan CASE araçları ile AML'nin hem UML 1.* hem de UML 2.0 için implementasyonunu sağlamayı amaçlamaktadırlar.

AML Profilleri üzerine başka kullanıcılar kendi dil uzantılarını geliştirebilir ve AML'i kendi ihtiyaçları doğrultusunda uzatabilirler. *AML Profile Extensions* adı altında bu uzantılar toplanabilir.

[9]'da da belirtildiği gibi AML, Agent UML'e göre daha yeni bir çalışmadır. Ancak onun da metinsel notasyonunun eksik olduğu ve herhangi bir geliştirme aracı tarafından şu an için desteklenmediği görülmektedir.

3 Modelleme Dillerinin Anlamsal Web'i Destekleme Gerekliliği

Anlamsal Web'in şu anki Web'den farklı değil de onun bir uzantısı olan, bilginin düzgün tanımlanmış bir anlama sahip olduğu ve insanlar ile bilgisayarların beraber çalışabildikleri bir web olarak çalışması hedeflenmektedir [13]. Günümüze kadar Web, veri ve bilgilerin otomatik olarak işlenebildiği bir ortamdan çok insanlar için doküman sağlayan bir medya olacak şekilde geliştirilmiştir. Anlamsal Web ise ilgili otomatik işlemeyi gerçekleştirme amacına sahiptir. Doğal olarak bu otomatik işleme ortamında insan kullanıcıları adına faaliyetlerde bulunması gereken otonom yapıların yani etmenlerin bulunması gerekmektedir.

Günümüzde hem etmen araştırmaları hem de anlamsal web üzerine yapılan çalışmalar birbirini destekleyecek şekilde devam etmektedir. Çünkü yazılım

etmenlerinin anlamsal web ortamlarında çalışacağına ve özellikle yetenekleri anlamsal olarak modellenmiş servislerle etmenlerin etkileşime gireceğine inanılmaktadır. Bu nedenle etmen sistemlerini geliştirmeye yönelik kullanılacak model dillerinin ve onların ürünü olan etmen meta-modellerinin anlamsal web yapılarını da kapsayacak şekilde ortaya çıkarılması gerekmektedir.

Literatürde adı geçen ve bizim de bu bildiride inceleme fırsatı bulduğumuz model dillerinin hiç biri yazılım etmenlerinin anlamsal web ortamlarında çalışmasına yönelik yapıları tanımlamamışlardır. Biz, yazılım etmenlerinin Anlamsal Web ortamında vazgeçilmez bir rol oynayacağına ve bünyelerine özellikle anlamsal web servisleri ile etkileşime ait yapıları da dahil etmeleri gerektiğine inanmaktayız [14].

Çok etmenli sistemler ve anlamsal web entegrasyonu için bizim önerdiğimiz çözüm: *çok etmenli bir sistem mimarisinin geleneksel yazılım mimarilerinden farklı olarak bünyesinde bir ortama (enviroment) da sahip olmasından hareketle; etmen topluluğu haricinde ortamda bağımsız (stand alone) servislerin de bulunmasını ve bu servislerin sahip oldukları anlamsal kimlikleri ile ortamın birer bileşeni olduklarını* öngörmektedir.

Şu an üzerinde çalıştığımız ve SEAGENT [15] adını verdiğimiz etmen tabanlı yazılım geliştirme çatısı, anlamsal web yetenekli çok etmenli uygulamaların geliştirilmesinde kullanılabilecek bütünleşik bir ortamı sağlamaktadır. Sağladığı genişletilebilir mimarisi FIPA tabanlıdır ve etmenlerin, planları dahilinde anlamsal web servisleri ile etkileşime girmesine izin vermektedir. Yazılım çatısı ve çok-etmenli sistemler için çatının sağladığı anlamsal uzantılar bu bildirinin kapsamı dışındadır ancak detaylı bilgiler [15], [16] ve [17]'deki çalışmalarda yer almaktadır.

Anlamsal Web'i bütünleşik olarak ortaya koyan böyle bir çatının tüm bileşenleri bütünüyle şekillendikten sonra bu bileşenleri içine alan bir meta-modelin ortaya çıkarılması gerektiğini ve ancak bu şekilde modelden koda dönüşümün uygulanabileceğini düşünmekteyiz. Bu nedenle şu an için yürütülmekte olan etmen modelleme çalışmalarının bu bildiride söz edilen eksikliklerinin de giderildiği yeni bir model dili üzerinde çalışmalarımız devam etmektedir. Elimizde zaten SEAGENT gibi hazır bir platform olduğundan ortaya konacak meta-modele göre tasarlanan etmen sistemlerinin hayata geçirilmesi kolay olacaktır.

4 Sonuç

Çok-etmenli yazılım sistemlerinin tasarımı ve geliştirilmesine yönelik önerilen modelleme dilleri ile ilgili bir inceleme bu bildiride yer almıştır. Literatürde adları geçen bu model dili çalışmalarının var olan etmen yazılım platformlarına uygunluk ve kabul edilebilirlik açısından daha çok etmen ortamları için UML'i uzatan model yapıları sundukları gözlenmiştir. Bunlar arasından sadece UML uzantısı olmayıp aynı zamanda UML üstyapısını temel alan çalışmaların kabul edilebilirliklerinin biraz daha fazla olduğu söylenebilir. Ancak şu an için model çalışmalarının hiçbiri tamamlanmış ve kullanılabilir bir ürün sunamamışlardır.

Etmen ortamlarına anlamsal web desteği getirmeleri açısından model dillerini incelediğimizde henüz hiçbirinin bu desteği verecek model yapılarını oluşturamadıkları ve/veya bünyelerine dahil etmedikleri ortaya çıkmaktadır. Bunun gelecek nesil çok-etmenli yazılım sistemlerinin Anlamsal web ortamlarında çalışması

gerekliliği ve etmenlerin, planları dahilinde yetenekleri anlamsal olarak tanımlanmış web servisleri ile etkileşime girme ihtiyaçları da düşünülürse çok büyük bir dezavantaj olduğu açıktır.

Kaynaklar

1. Russell S. J., Norvig P., 2003, Artificial Intelligence: A Modern Approach Second Edition, Pearson Education, USA, 1080p.
2. Durfee E. H., Lesser V. R., Corkill D. D., 1989, Trends in cooperative distributed problem solving, IEEE Transactions on Knowledge and Data Engineering, KDE-1 pp 63 – 83.
3. Zambonelli F., Omicini A., 2004, Challenges and Research Directions in Agent-Oriented Software Engineering, Journal of Autonomous Agents and Multiagent Systems, 9 (3).
4. Gervais, M., Gomez J., Weiss G., 2004, *A Survey on Agent-Oriented Software Engineering Researches*, Methodologies and Software Engineering for Agent Systems, Kluwer.
5. Bauer, B. and Odell, J. (2005) "UML 2.0 and agents: how to build agent-based systems with the new UML standard", Engineering Applications of Artificial Intelligence, Volume 18, Issue 2 , March 2005, Pages 141-157.
6. Foundation for Intelligent Physical Agents (FIPA): http://www.fipa.org
7. Odell J., Parunak H. V., Bauer B., 2000, *Extending UML for Agents*, Proceedings of Agent Oriented Information Systems Workshop at the 17th National conference on AI.
8. Marc-Philippe Huget J. O. and Bauer B., 2004, The AUML Approach, volume 11 of Methodologies and Software Engineering for Agent Systems, chapter 10, Kluwer.
9. Huget, M. (2005) Modeling Languages for Multiagent Systems, Agent Oriented Software Engineering Workshop 2005, Lecture Notes in Computer Science.
10. FIPA Modeling TC: Agent Class Superstructure Metamodel: www.auml.org/auml/documents/CD2-04-01-21.doc .
11. Odell J., Nodine M. and Levy R.. (2005) A Metamodel for Agents, Roles, and Groups, AOSE V, Lecture Notes in Computer Science.
12. Cervenka, R., Trencansky, I., Calisti, M. and Greenwood, D. (2004) AML: Agent Modeling Language – Toward Industry-Grade Agent-Based Modeling, AOSE 2004, Lecture Notes in Computer Science.
13. Berners-Lee, T., Hendler, J. and Lassila, O.: The Semantic Web, Scientific American, 284(5), pp:34-43.
14. Dikenelli, O., Gümüs, O., Tiryaki, A. M. and Kardas, G. (2005) “Engineering a Multi Agent Platform with Dynamic Semantic Service Discovery and Invocation Capability”, Multiagent System Technologies - MATES 2005, Lecture Notes in Computer Science (Subseries: Lecture Notes in Artificial Intelligence), Springer-Verlag, Vol. 3550, pp. 141-152.
15. Dikenelli, O., Erdur, R. C., Gumus, O., Ekinci, E. E., Gurcan, O., Kardas, G., Seylan, I. and Tiryaki, A. M. (2005) “SEAGENT: A Platform for Developing Semantic Web Based Multi Agent Systems”, AAMAS’05, ACM AAMAS, pp. 1271-1272.
16. Kardas, G., Gümüs, Ö. and Dikenelli, O. (2005) “Applying Semantic Capability Matching into Directory Service Structures of Multi Agent Systems”, Computer and Information Sciences - ISCIS 2005, Lecture Notes in Computer Science, Springer-Verlag, Vol. 3733, pp. 452-461.
17. Dikenelli, O., Erdur, R. C., Kardas, G., Gümüs, O., Seylan, I., Gurcan, O., Tiryaki, A. M. and Ekinci, E. E. (2005), “Developing Multi Agent Systems on Semantic Web Enviroment using SEAGENT Platform”, ESAW’05 (6th International Workshop on Engineering Societies in the Agents’ World), Springer-Verlag’in Lecture Notes in Computer Science