

SABPO Metodolojisi Kullanılarak FIPA Uyumlu Çok-Etmenli bir Otel Rezervasyon Sisteminin Tasarımı ve Gerçekleştirilmesi

Ayşegül Alaybeyoğlu¹, Geylani Kardaş², Rıza Cenk Erdur¹, Oğuz Dikenelli¹

¹ Ege Üniversitesi, Bilgisayar Mühendisliği Bölümü, İzmir

² Ege Üniversitesi, Uluslararası Bilgisayar Enstitüsü, İzmir

aysegul.alaybeyoglu@ege.edu.tr, geylani.kardas@ege.edu.tr, cenk.erdur@ege.edu.tr, oguz.dikenelli@ege.edu.tr

Özet: Bu çalışmada JADE etmen çerçevesini kullanan ve FIPA standartlarına uyan çok-etmenli bir otel rezervasyon sistemi tasarlanmış ve gerçekleştirilmiştir. Tasarım ve gerçekleştirim sırasında SABPO Çok-etmenli sistem geliştirme metodolojisi süreçleri takip edilmiştir. Bildiride, sistem etmenlerine ait rollerin belirlenmesi, bu roller ile etmenlerin eşlenmesi ve çok-etmenli etkileşimlerin belirlenmesi de dahil olmak üzere sistemin baştan tasarımı, hayata geçirilmesi ve test edilmesine dair yerine getirilen çalışmalar yer almaktadır.

Anahtar Sözcükler: Yazılım Etmenleri, Çok-etmenli Sistemler, Yazılım Mimarisi.

Design and Implementation of a FIPA compliant Hotel Reservation Multi-Agent System using SABPO Methodology

Abstract: In this study, a FIPA compliant multi-agent system for hotel reservation is designed and implemented using JADE agent framework. During development of the system, analysis and implementation steps of SABPO Multi-agent development methodology are followed. In this paper, work on design, implementation and test of the system including definition of agent roles, role mapping of agents and determination of multi-agent interactions are discussed.

Keywords: Software Agents, Multi-agent Systems, Software Architecture.

1. Giriş

Bir yazılım etmeni, kullanıcısının adına bir takım görevleri yerine getirmek üzere davranma yeteneği olan özerk (otonom) ve amaç yönelimli bir yapıya sahip yazılım bileşenidir. Tek bir etmenin yalnız başına kendi bilgi ve bireysel yeteneklerini kullanarak çözümediği veya etkin bir biçimde çözemeyeceğini düşündüğü problemleri birbiriyle işbirliği yaparak eşgüdümlü bir biçimde çözmek için bir araya gelen etmenlerin oluşturduğu ağ ise çok-etmenli sistem olarak adlandırılmaktadır [3]. Böyle bir etmen sistemine özellikle İnternet üzerinden iş akışının gerçekleştirildiği bir çok sektörde ihtiyaç duyulabilmektedir ki bu sektörlerden biri de turizmdir.

Özellikle sektör içerisinde otel rezervasyon sistemlerinin etkin ve kullanıcılarını en yüksek düzeyde memnun edecek bir yapıda olması beklenmektedir. Müşteriyi temsil eden ve onun yerine ilgili rezervasyon işlemlerini yürüten yazılım etmenlerinin varolması sistemin daha hızlı işlemlerini ve işlemlerin mümkün olduğunca kullanıcı yararına sonuçlanmasını sağlayacaktır. Bu çalışmada da müşterileri ve otelleri temsil eden etmenlerin oluşturduğu çok-etmenli bir otel rezervasyon sistemi tasarlanmış ve gerçekleştirilmiştir. Geliştirilen sistem FIPA (Foundation For Intelligent Physical Agents) [4] uyumludur ve JADE (Java Agent Development Framework) [1] çerçevesini kullanmaktadır.

FIPA, çok-etmenli sistemler arasındaki birlikte-çalışabilirliği (interoperability) en üst düzeye çıkartmak için evrensel standartlar ortaya koymak amacı ile kurulan, kar amacı gütmeyen bir topluluktur. Günümüzde ortaya konan etmen tabanlı yazılım sistemlerinin büyük bir kısmı bu topluluğa ait soyut mimariye uygun olarak tasarlanmıştır. JADE yazılım çerçevesi ise FIPA standartlarına uyumlu etmen sistemlerinin Java ortamında hazırlanmasına imkan vermektedir.

Çok-etmenli sistemlerin hazırlanması sırasında diğer yazılım sistemlerinin hazırlanmasında olduğu gibi belli yazılım metodolojileri kullanılmaktadır. Bu çalışma kapsamında geliştirilen çok-etmenli sistemin hazırlanması sırasında SABPO (Standards Based and Pattern Oriented) [2] çok-etmenli sistem geliştirme metodolojisinin tanımladığı süreçler takip edilmiştir. SABPO, etmen sistemleri için “organizasyon” metaforunu temel almakta ve bu metaforu FIPA standartları ve bilinen etkileşim protokolleri ile sistematik bir biçimde bütünleştirmektedir.

Bildiride, hazırlanan etmen sisteminin SABPO'nun yinelemeli (iterative) süreçlerine dayanan geliştirme aşamaları ele alınmıştır. Sistem etmenlerine ait rollerin belirlenmesi, bu roller ile etmenlerin eşlenmesi ve çok-etmenli etkileşimlerin belirlenmesi de dahil olmak üzere sistemin baştan tasarımı, hayata geçirilmesi ve test edilmesine dair yerine getirilen çalışmalar yer almaktadır.

Bildirinin ikinci bölümünde sistem tasarımından bahsedilmektedir. Üçüncü bölümde sistemin çalıştırılması ve testi yer almaktadır. Dördüncü bölüm olan sonuç bölümünde ise sistem hazırlanırken elde edilen deneyimler ve gözlemler yer almaktadır.

2. Sistemin SABPO Adımları ile Tasarlanması

Çok-etmenli yazılım sisteminin hayata geçirilmesi sırasında SABPO metodolojisinin tanımladığı sistem geliştirme adımları

uygulanmıştır. İzleyen altbölümlerde bu adımlarda gerçekleştirilen çalışmalar anlatılmıştır.

2.1 Sistemde Yer Alan Etmenlerin Üstleneceği Rollerin Belirlenmesi

Çok-etmenli otel rezervasyon sisteminde etmenler farklı kullanıcıları temsil etmektedirler. Buna göre her etmenin üstlendiği roller daha çok temsil ettiği kullanıcının sistemde gerçekleştirmek istediği işlemlere bağlıdır. Üstlenilecek roller şu başlıklar altında toplanabilir:

Kullanıcı Rolü: Kullanıcının çok-etmenli sistemle etkileşimde bulunabilmesini sağlayacak bir ara yüzü, kullanıcının istediği işlemleri gerçekleştirmeyi sağlayan diğer etmenlerin sunduğu servislerin etmen platformundaki yerlerinin belirlenmesini ve gerektiğinde kullanımını içermektedir.

Servis Belirleme Rolü: Bileşen kaynaklarının yani etmenlerin sunduğu sisteme özgü servislerin neler olduğu ve hangi etmenler tarafından sunulduğu bilgisinin tutulmasını içermektedir.

Servis ya da Bileşen Sunucu Rolü: Sunduğu servise veya bileşene ait üst verilerin tutulmasını, servislere (veya bileşenlere) diğer etmenlerin platform bağımlı olarak güvenli erişiminin sağlanmasını içermektedir.

2.2 Rollerin Etmenlere Eşlenmesi ve Ontolojilerin Belirlenmesi

Bir önceki alt bölümde verilen roller ve uygulama alanı (domain) göz önüne alınarak etmen tasarımları gerçekleştirilmiş ve kullanılan etmen çatısına bağlı olarak bir çok-etmenli sistem hazırlanmıştır.

Otel rezervasyon sisteminde hem otel müşterileri hem de otellerin kendisi birer kullanıcı olacak şekilde düşünülebilir ve bu şekilde bir tasarıma gidilebilir. Ancak sistemdeki etkileşimler daha çok müşteri kaynaklı olmakta ve servisleri daha çok oteller sunmaktadır. Bu nedenle müşterileri temsil eden etmenlerin kullanıcı rolünü üstlenmesi daha gerçekçi bir yaklaşım olarak

görülmüştür. Buna karşılık servis sunucu rolünü otelleri temsil eden etmenlerin üstlenmesi düşünülmüştür.

Servis belirleme rolünü üstlenmesi amacıyla bir servis eşleyici (matchmaker) etmenin tasarımı başlangıçta düşünülse de FIPA uyumlu JADE çatısının sunduğu çok-etmenli platformda böyle bir rolü üstlenecek bir etmenin halihazırda olması nedeniyle aynı görevi yerine getirecek yeni bir etmenin gerçekleştirilmesinden vazgeçilmiştir.

Müşteri – otel etkileşimleri için öncelikle (etkileşimi başlatan olarak) müşterilerin otelleri belirlemesi gerekmektedir. Temsilciler yani etmenler bazında düşünüldüğünde müşteri etmeninin, servis sunan otel etmenlerinin tanımlarını etkileşimler öncesi elde etmesi gerekmektedir. Bu hizmeti FIPA uyumlu bir sistemde DF (Directory Facilitator - Dizin Kolaylaştırıcı) sağlamakta [4]; JADE çatısında da çok-etmenli sistem için bu rolü üstlenen ve *DFSservice* adı verilen bir etmen yer almaktadır. Buna göre müşteri etmenleri etkileşimleri öncesi bu etmeden aradıkları servisi sağlayan etmenlerin listesini elde etmekte ve bu etmenlerle iletişime geçmektedirler.

Hazırlanan etmenlerin detaylarına geçmeden önce vurgulanmak istenen son husus ise sistemin kullandığı ontoloji hakkındadır. Her ne kadar bu çalışma kapsamında detaylı bir otel rezervasyon ontolojisi tasarlanmasa da etkileşimlerin gerçek anlamda yürütülmesini sağlayacak deyimlerin oluşturulması yerine getirilmiştir.

FIPA-uyumlu ve JADE çatısına dayanan otel rezervasyon sistemi kütüphanesi “*tourism*” adı verilen Java paketinde yer almaktadır. Bu pakette yer alan sınıfların bulunduğu UML sınıf diyagramı Şekil 1’dedir.

Sınıf modeli incelendiğinde *jade.core.Agent* sınıfının alt sınıfı olan iki sınıf görülmektedir - ki bu sınıflar çok-etmenli otel rezervasyon sistemindeki etmenlerin türetildiği sınıflardır. Bunlardan

tourism.CustomerAgent adından da anlaşılacağı üzere müşterileri temsil eden etmenlerin türetildiği sınıf iken *tourism.HotelAgent* otel etmenlerinin türetildiği sınıftır. Her iki tipteki etmenler de kendilerine özel davranışları uygulamakta ve işlemlerini yürütmektedirler. Tüm etmenler platformdaki faaliyetlerini bir *tourism.LogWriter* nesnesi aracılığı ile kendi adını taşıyan bir dosyaya kaydetmektedir. Aşağıda ilgili etmen sınıfları ve etmen davranışları ile ilgili detaylı bilgiler yer almaktadır:

HotelAgent sınıfı: Çok-etmeli otel rezervasyon sisteminde otelleri temsil eden etmenler bu sınıftan türetilmektedir. Otel etmeni oluşturulurken temsil edeceği otel nesnesine ait özellikler parametre olarak verilir. Bu parametreler, temsil edilen otelin adı, adresi ve oda sayısıdır. Etmenin oluşturulma aşamasında temsil ettiği otel, *tourism.Hotel* nesnesinden türetilerek ilgili parametrelere göre hazırlanır ve etmen *tourism.HotelAgentBehaviour* tipindeki davranışını uygulamaya geçirir.

Etmenin içerdiği *Hotel* nesnesi yukarıda belirtildiği gibi etmenin temsil ettiği otele ait bilgileri barındırmaktadır. Bir *Hotel* nesnesi, *tourism.HotelRoom* sınıfından türetilen nesnelerin bir listesini tutmaktadır. Tasarımda, bir otelin birden fazla odanın bütününden oluştuğu düşünülmektedir ve her bir oda bir *HotelRoom* nesnesi ile temsil edilmektedir. Her odaya ait yatak sayısı, oda ücreti ve rezerve olup olmadığı bilgilerini ilgili *HotelRoom* nesnesi barındırır.

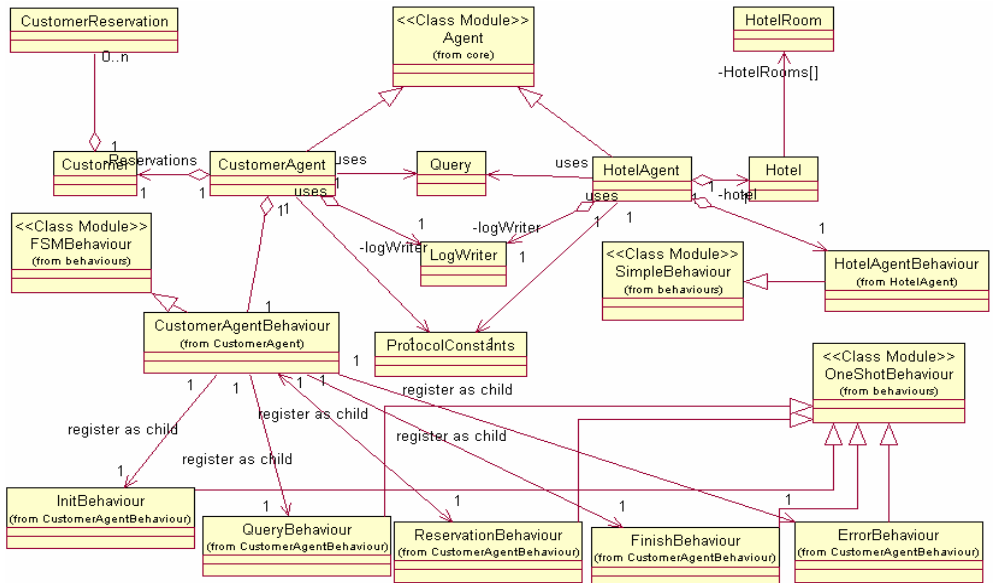
Bir etmen olarak *HotelAgent* da yaşamı süresince davranışlarda bulunmaktadır. Bu etmenin kendi davranış kuyruğuna eklediği ve uygulamaya geçirdiği tek davranış *tourism.HotelAgentBehaviour* sınıfından türetilmektedir. Sunulan servisler ve etkileşimler göz önüne alındığında *HotelAgentBehaviour*’ın JADE platformundaki davranış hiyerarşisindeki *jade.core.behaviours.SimpleBehavior*

ur sınıfının bir alt sınıfı olarak tasarlanması ve gerçekleştirilmesi uygun görülmüştür.

HotelAgent etmeni platformda faaliyete geçer geçmez öncelikle DF'ye kendini otel servisi sağlayan bir etmen olarak DFService etmeni aracılığı ile kayıtlamaktadır. Daha sonra kendine özel davranışını (HotelAgentBehaviour) uygulamaktadır. Buna göre ilk olarak müşteri etmenlerinden gelecek oda sorgusu ACL (Agent Communication Language – Etmen İletişim Dili) mesajlarını beklemekte; gelen mesajların içeriğini sisteme özel ontolojiye uygun olarak değerlendirmekte ve temsilcisi olduğu otelin odalarını sorgu parametrelerine göre değerlendirmektedir. Sorgu kriterlerine uyan ve rezerve olmayan bir oda bulunduğunda veya uygun hiçbir oda bulamadığında sorgulayıcı müşteri etmenine sorgu sonucunu döndürmektedir. Uygun durumda müşteri etmeninden gelen rezervasyon isteğini de mesajın gerçekleştiricisine (performative) bakarak belirlemekte ve oda halihazırda boşsa rezervasyon isteğini gerçekleştirerek müşteri etmenine, onun adına rezerve ettiği odanın numarasını döndürmektedir. Olumsuz rezervasyon sonucu da yine protokole uygun olarak müşteri etmenine bildirilmektedir.

CustomerAgent sınıfı: Çok-etmeli otel rezervasyon sisteminde otel müşterilerini temsil eden etmenler bu sınıftan türetilmektedir. Bir CustomerAgent nesnesi platformda faaliyete geçmeden önce temsil ettiği müşterinin bilgilerini tutan bir tourism.Customer nesnesini ve oda sorgusu ve rezervasyonu için gerekecek protokole uygun deyimleri kendisine verilen argümanlara göre oluşturur. İlgili argümanlar müşterinin adı, oda sorgusu tipi ve parametreleridir. Sorgu tipi tourism.Query sınıfında tanımlanan tiplerden biri olmak zorundadır. Bu tipler, aldıkları argümanlar ve anlamları Tablo 1’de verilmiştir.

Müşteri bilgilerini temsil eden Customer nesnesi CustomerAgent’in müşteri adına yapmış olduğu her bir rezervasyonu kendi içinde bir vektörde tutmaktadır. Bu vektördeki her bir nesne tourism.CustomerReservation tipindedir ve her bir rezervasyona ait bilgileri içermektedir. Bu bilgiler rezervasyonun yapıldığı otelin adı ve rezervasyon oda numarasıdır.



Şekil 1. Çok-etmeli otel rezervasyon sistemi sınıf diyagramı

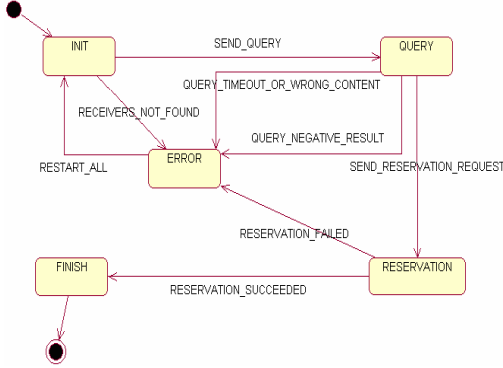
| Sorgu tipi | Argüman(lar) | Sorgu anlamı |
|---------------------|----------------------------|--|
| ANY | - | Herhangi bir otel odasını sorgula |
| BED_COUNT | Yatak sayısı | Yatak sayısı, verilen argümana eşit olan odaları sorgula |
| PRICE | Max. fiyat | Fiyatı, verilen argümana eşit veya daha az olan odaları sorgula |
| BED_COUNT_AND_PRICE | Yatak sayısı ve max. fiyat | Yatak sayısı, verilen yatak sayısına eşit ve fiyatı, en fazla verilen değere eşit olan odaları sorgula |
| BED_COUNT_OR_PRICE | Yatak sayısı ve max. fiyat | Yatak sayısı, verilen yatak sayısına eşit veya fiyatı, en fazla verilen değere eşit olan odaları sorgula |

Tablo 1. Sistemdeki sorgu tipleri ve aldıkları parametreler

CustomerAgent etmeni platformda yaşamaya başlar başlamaz kendisine özgü davranışını uygulamaya geçirir. Bu etmenin kendi davranış kuyruğuna eklediği ve uygulamaya geçirdiği tek davranış `tourism.CustomerAgentBehaviour` sınıfından türetilmektedir.

CustomerAgent etmenlerinin sistemdeki davranışları `HotelAgent`'lerden yapı itibariyle farklılık göstermektedir. Daha önce de bahsedildiği gibi `HotelAgent`'ların uygulamaya koyduğu `HotelAgentBehaviour`'lar JADE çatısı davranış hiyerarşisindeki `jade.core.behaviours.SimpleBehaviour` sınıfının bir alt sınıfı olarak tasarlanmış ve gerçekleştirilmişlerdir. `CustomerAgent`'ların, sistemdeki iletişimleri başlatıcı rolde olmaları ve bu iletişimlerin önceki iletişimlere bağlı olması gibi gerçekler göz önüne alınarak bu etmenlerin uygulamaya koyacağı `CustomerAgentBehaviour`'ın bir FSM (Finite State Machine – Sonlu Durum Makinesi) olarak tasarlanıp gerçekleştirilmesi uygun görülmüştür. Bu nedenle bu nesnelerin türetildiği sınıf `jade.core.behaviours.FSMBehaviour` sınıfının bir alt sınıfı olarak hazırlanmıştır. FSM yapısındaki `CustomerAgentBehaviour`'ın durumlarını oluşturan çocuk davranış sınıflarının her biri `jade.core.behaviours.OneShotBehaviour` sınıfının alt sınıfıdır. Bu sınıflar: `tourism.InitBehaviour`, `tourism.QueryBehaviour`, `tourism.ReservationBehaviour`, `tourism.FinishBehaviour` ve `tourism.ErrorBehaviour`'dir.

Bir `CustomerAgent`, Şekil 2'de verilen bir sonlu durum makinesini davranış olarak uygular: Etmen platformda yaşamaya geçtikten sonra ilk olarak DF'den otel servisi sunan etmenlerin tanımlarını istemektedir. Böylelikle oda sorgusu mesajlarının alıcısı konumundaki otel etmenlerini belirlemiş olur. Bu işlemleri, davranışı INIT durumunda iken gerçekleştirir. İşlemler başarı ile gerçekleştirildiğinde etmen artık QUERY safhasına geçebilir. Bu durumda iken etmen, alıcı konumundaki tüm otellere oda sorgusunu, ontolojiye uygun olarak içeri doldurduğu ACL mesajları ile gönderir ve cevapları beklemeye başlar. Sorgusuna olumlu cevap gönderen ilk oteli belirledikten sonra artık rezervasyon isteğini bildirecektir. Bu nedenle RESERVATION durumuna geçer ve belirlediği otele rezervasyon isteğini bildirir. Olumlu yanıt aldığı anda artık ilgili otelde rezervasyon gerçekleştirilmiştir. `Customer` nesnesinin rezervasyonlar listesine yeni rezervasyon eklenir ve FINISH durumuna geçilir. Normal akış içerisinde olabilecek hatalarda etmen ERROR durumuna geçer ve daha sonra tüm işlemlerine yeniden başlar. Örneğin etmen platformda otel servisi sunan bir etmen bulamazsa INIT durumundan ERROR durumuna düşmektedir. Ya da sorgusuna timeout süresince yanıt alamadığında QUERY safhasında yine ERROR durumuna geçer. ERROR durumunda ilgili loglama işlemi yerine getirildikten sonra uygun geçişle tekrar INIT safhasına dönlür.



Şekil 2. CustomerAgent etmeninin FSM yapısındaki davranış modeli

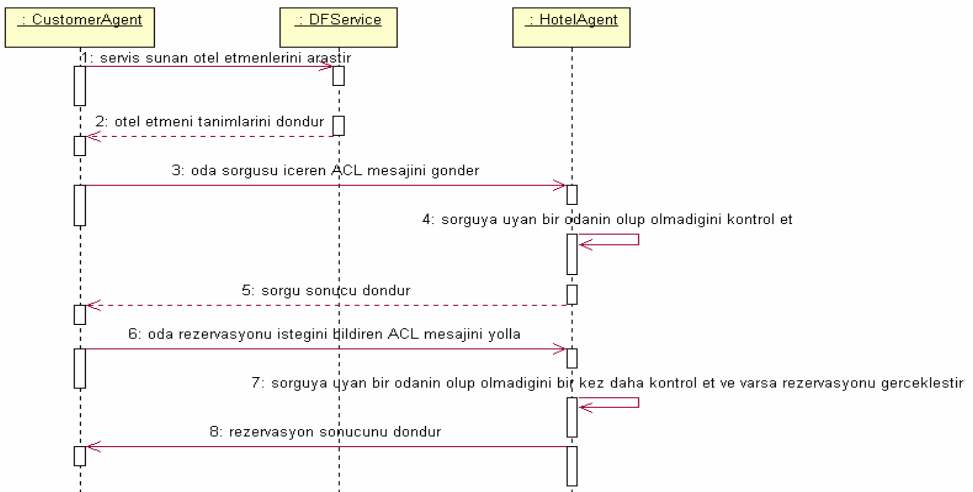
ProtocolConstants sınıfı: Daha önce de belirtildiği gibi sistem hazırlanırken detaylı bir ontoloji hazırlığı yapılmamış ancak etkileşimlerin gerçek anlamda sağlanabilmesi için ACL mesaj içeriklerinin uygun yapıda olması ve tüm etmenler tarafından anlaşılması sağlanmıştır. Hazırlanan deyimlere ait kelimeleri `tourism.ProtocolConstants` barındırmakta ve bir anlamda ontolojiyi temsil etmektedir. Örneğin oda fiyatı 100\$ olan odaları sorgulayacak bir CustomerAgent, sorgu ACL mesajı içeriğini `QUERY_PRICE + QUERY_SEPERATOR + "100"` şeklinde oluşturulan bir `java.lang.String` nesnesi ile

doldurmaktadır. Burada kullanılan `QUERY_PRICE` ve `QUERY_SEPERATOR` `java.lang.String` tipinde olup `ProtocolConstants` sınıfında tanımlanmışlardır.

2.3 Çok-etmenli Etkileşimlerin Belirlenmesi

Bir önceki bölümde rolleri belirlenen etmenlerin otel rezervasyon sistemindeki en önemli etkileşimleri adından da anlaşılacağı üzere bir müşteri etmeninin (CustomerAgent) oda sorgusu ve sonrasındaki rezervasyon isteğidir. Bu senaryoda aktörler bir müşteri etmeni (CustomerAgent), platform DF etmeni ve otel servisi sunan otel etmenleridir (HotelAgent). Şekil 3'te senaryoya ait "sequence" (nesnelere arası etkileşim dizisi) diyagramı yer almaktadır.

Daha önce de belirtildiği gibi bir müşteriyi temsil eden CustomerAgent etmeni DFService etmeninden otel servisi sunan –ki bu servis sabiti HotelAgent sınıfında tanımlanmıştır – etmenlere ait DFAgentDescription'ları (etmen servisi tanımlarını) alır ve bu etmenlere oda sorgusunu içeren ACL mesajlarını gönderir.



Şekil 3. Oda sorgusu ve rezervasyonu senaryosuna ait nesnelere arası etkileşim dizisi diyagramı

Otelleri temsil eden `HotelAgent` etmenleri sorguya uyan otel odalarının olup olmadığını araştırır ve sorgu sonucunu ilgili `CustomerAgent`'a geri döndürür. `CustomerAgent` kendisine olumlu yanıt veren ilk otele kabul mesajı ile birlikte sorgusunu bir kez daha göndererek rezervasyon isteğini bildirir. İlgili `HotelAgent` temsil ettiği otele ait odaları bir kez daha sorguya göre kontrol eder ve uygun odayı bulduğunda rezervasyonu gerçekleştirerek rezervasyon sonucunu oda numarası ile birlikte `CustomerAgent`'a geri döndürür. `CustomerAgent` temsil ettiği müşterinin rezervasyonlar listesine, gelen oda numarasını ve otelin adını barındıran yeni bir rezervasyonu ekleyerek senaryoyu tamamlamış olur.

2.4 Senaryolar İçin HTN'lerin Hazırlanması

Çok-etmenli otel rezervasyon sistemi tasarlanırken etmen görevlerinin planlama katmanları için gerekli HTN (Hierarchical Task Network – Hiyerarşik Görev Ağı) yapıları da oluşturulmuştur. Bunun bir örneği Şekil 4'te gösterilen, rezervasyon senaryosunda `CustomerAgent`'in yerine getirdiği görevlere ait olan ağdır. İlgili HTN incelendiğinde asıl görevin “oda rezervasyonu” olduğu ancak bunun “otel

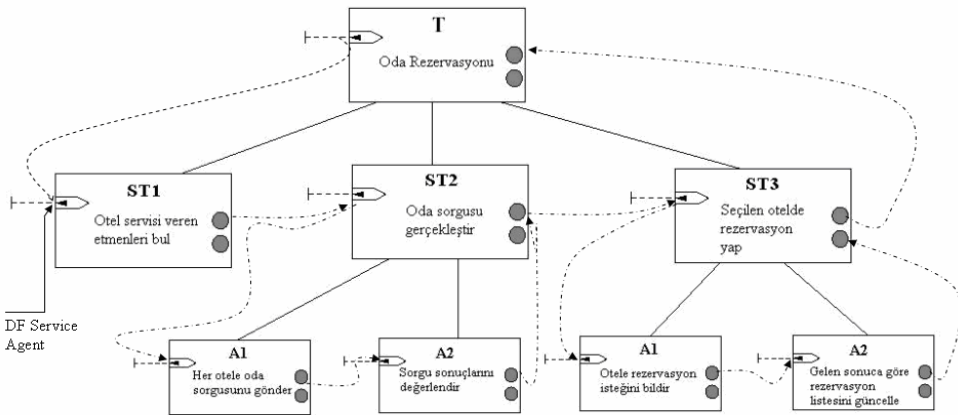
servisi veren etmenlerin bulunması”, “oda sorgusu gerçekleştirilmesi” ve “seçilen otele rezervasyon yapılması” gibi alt görevlere ayrıştırıldığı görülmektedir. Bu alt görevleri de oluşturan çeşitli eylemler (actions) yer almaktadır.

3. Sistemin Çalıştırılması ve Testi

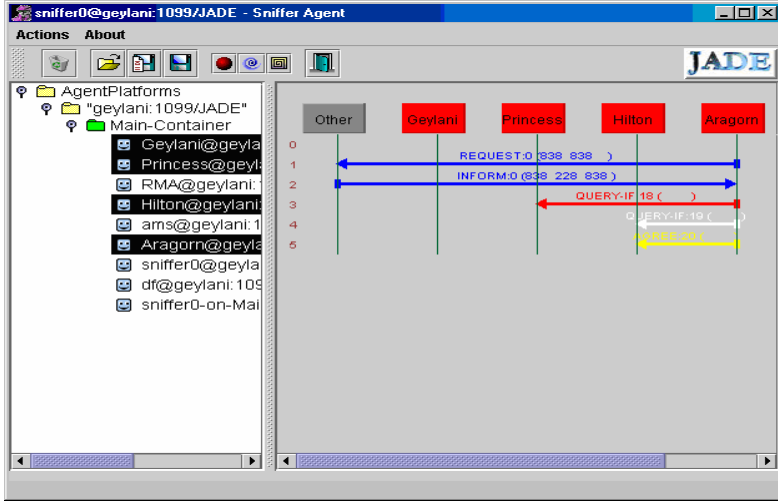
Sistem tüm bileşenleri ile hazırlandıktan sonra platform çalıştırılmış ve etmenlerin davranışları test edilmiştir. Gerek etmenlerin tuttuğu işlem kayıt dosyaları aracılığıyla gerekse de JADE kütüphanesi içerisinde yer alan `Sniffer` (Yoklayıcı) etmeni kullanılarak etmenlerin etkileşimleri gözlenmiştir. Bunların yanı sıra yine JADE kütüphanesinde yer alan “dummy agent” kullanılarak etmenlerin hazırladığı FIPA ACL mesajları kontrol edilmiştir. Şekil 5'te `Sniffer` etmeni vasıtasıyla etmenlerin gerçekleştirdiği etkileşimlerden bir örnek görülmektedir.

4. Sonuç

Sistem yazılımı geliştirme süreci bu kapsamdaki bir çok-etmenli platformun hazırlanması için nispeten kısa bir zaman almıştır. Bunda en önemli paylardan biri JADE çatısının sistem geliştirmede ve gerçekleştirmede kullanılmasıdır.



Şekil 4. Oda Rezervasyonu görevine ait HTN yapısı



Şekil 5. JADE Sniffer (Yoklayıcı) etmeni kullanılarak etmen iletişimlerinin gözlenmesi

Ancak yazılım geliştirme sırasında bazı eksiklikler de göze çarpmıştır. Örneğin etmen mesaj kuyruklarının istenildiğinde temizlenebilmesine JADE çatısındaki “Agent” sınıfının imkan vermediği görülmüştür. Her ne kadar bu ihtiyaç JADE’teki “MessageTemplate” yapısı kullanılarak çözüle de kuyruk temizleme gibi bir imkanın olmasının daha uygun olacağı düşünülmektedir. Ancak bunun geliştirilen platformda istenmeyen durumların oluşmasına neden olabileceği ve bu yüzden böyle bir imkanın sunulmadığı da düşünülmektedir. Bunun yanı sıra gönderilen mesajlara ait zarfların (envelope) içeriğinin bir metin olarak alınmasında –ki bunun için hazır metot JADE çatısında yer almaktadır - nedeni belirlenemeyen hataların oluştuğu gözlenmiştir.

Yazılım geliştirme sürecinde sıkıntısı çekilen bir konu da ACL mesaj içeriklerinin hazırlanmasında yaşanmıştır. Başlangıçta içerik olarak bayt dizisine dönüştürülmüş nesnelere gönderilmesi düşünülmüştür. Fakat bilindiği gibi şu an için FIPA standartlarında içeriğin nesne olarak aktarılması yer almadığından klasik metin

aktarımı tasarlanmış ve gerçekleştirilmiştir. Her ne kadar bu aktarım metin manipülasyonu için ekstra yazılım kodlarının hazırlanmasını gerektirse de bunun, sistemin tamamen FIPA uyumlu olması için gerekli olduğu düşünülmüş ve içerik hazırlanması bu şekilde gerçekleştirilmiştir.

5. Kaynaklar

- [1] Bellifemine, F., Poggi, A. and Rimassa, G., “Developing Multi-agent Systems with a FIPA-compliant Agent Framework”, **Software Practice and Experience**, 31: 103-128 (2001).
- [2] Dikeneli, O. and Erdur, R. C., “SABPO: A Standards Based and Pattern Oriented Multi-agent Development Methodology”, **Lecture Notes in Artificial Intelligence**, 2577:213-226 (2003).
- [3] Erdur, R. C., “Çok-Etmenli Sistemler”, Etmen Tabanlı Yazılım Geliştirme dersi notları, Ege Üniversitesi Bilgisayar Mühendisliği Bölümü, 26 sayfa (2001).
- [4] FIPA Standards, <http://www.fipa.org/>.