İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

DAĞITIK ÇOKLU ETMEN TOPLANTI PLANLAMA SİSTEMİ

YÜKSEK LİSANS TEZİ

Bilgisayar Müh. Ali DURMUŞ
(504991163)

Tezin Enstitüye Verildiği Tarih : 23 Temmuz 2002
Tezin Savunulduğu Tarih : 29 Temmuz 2002

Tez Danışmanı :	**Doç.Dr. Nadia ERDOĞAN**
Diğer Jüri Üyeleri	**Prof. Dr. Emre HARMANCI (İ.T.Ü.)**
	Doç. Dr. B. Tevfik AKGÜN (Y.T.Ü.)

AĞUSTOS 2002

ÖNSÖZ

Bana bu çalışmam sırasında çok değerli katkılarını sunan ve yardımlarını hiç bir zaman esirgemeyen Doç. Dr. Nadia Erdoğan'a çok teşekür ederim.

Ayrıca bu çalışma boyunca beni destekleyen ve bana katlanan ev arkadaşlarıma, yazdığım programı kullanarak görüşlerini belirten iş arkadaşlarıma ve değişik şekillerde beni destekleyen ve yanımda olan arkadaşlarıma teşekür ederim.

Temmuz 2002 Ali Durmuş

İÇİNDEKİLER

KISALTMALAR

ÇES	: Çoklu Etmen Sistemi
JATLite	: Java Agent Template, Lite
SAP	: Sözleşme Ağ Protokolü
EİD	: Etmen İletişim Dili
FIPA	: Foundation for Intelligent Physical Agents
ACL	: Agent Communication Language
TCP/IP	: Transfer Control Protocol, Internet Protocol
KQML	: Knowledge Query and Manipulation Language
BNF	: Backus Naur Form
SDK	: Software Development Kit
SMTP	: Simple Mail Transfer Protocol
POP3	: Post Office Protocol
HTTP	: Hyper Text Transfet Protocol
FTP	: File Transfer Protocol
KIF	: Knowledge Interchange Format
JDK	: Java Development Kit
ASCII	: American Standard Code for Information Interchange
CORBA	: Common Object Request Broker Architecture
SQL	: Structured Query Language
RMI	: Remote Method Invocation
AMR	: Agent Message Router
ANS	: Agent Name Server
CPU	: Central Process Unit

TABLO LİSTESİ

ŞEKİL LİSTESİ

DAĞITIK ÇOKLU ETMEN TOPLANTI PLANLAMA SİSTEMİ

ÖZET

Bu tezde bir dağıtık çoklu etmen toplantı planlama sistemi yapısı, algoritması önerilmekte ve bunun uygulama ayrıntıları anlatılmaktadır. Toplantı planlamak günlük hayatta sıkça karşılaşılan ve zaman tüketici bir işlemdir. Toplantı planlama işlemi düzenleyicinin koordinasyonu ve katılımcıların çabaları ile bir sonuca ulaşır. Günlük hayattaki toplantı planlama işleminin gerçek bir simulasyonunu sağlamak için dağıtık çoklu etmen sistemleri yapısı kullanılmıştır. Bu yapı ile birlikte yeni bir toplantı planlama algoritması önerilmiştir. Böylece kullanıcı için zaman kaybettirici ve sürekli tekrar eden bir işlem olan toplantı planlama işlemi kullanıcı adına işleminin yapan etmenler tarafından yürütülmesi sağlanmıştır.

Bir etmenler ağı olan dağıtık çoklu etmen sistemi toplantı planlama işleminin çözümü için kullanılan mimari yapı olmuştur. Sistemdeki her bir etmen bir kullanıcıyı temsil etmektedir. Ayrıca sistemde toplantı yeri bilgileri de bir etmen tarafından temsil edilmektedir. Toplantı planlama işlemi bir dağıtık arama işlemi olarak düşünülmüştür. Toplantı planlama problemin çözümü için işbirlikçi bir toplantı planlama algoritması önerilmiştir. Toplantı planlama algoritmasında öneri ve karşı öneriler ile çözüme ulaşılmaktadır. Ayrıca sistemde bulunan yer etmeni de bir toplantı davetlisi gibi davranmaktadır.

Sistemi gerçeklerken etmenler arası iletişimi sağlayan, etmen yaratma için şablonlar sunan JATLite paketi kullanılmıştır. JATLite dağıtık çoklu etmen sistemi için bir alt yapı sunmaktadır. JATLite ile etmenler arası haberleşme için çok esnek bir yapı sunmaktadır.

Sistemde bulunan etmenler arası mesajlaşmalarda mesajlaşma dili olarak veri ve bilgi alışverişi için kullanılan dil olan KQML kullanılmıştır. Algoritmanın gerçeklenmesi için yeni KQML performative'leri önerildi.

DISTRIBUTED MULTI-AGENT MEETING SCHEDULING SYSTEM

SUMMARY

In this thesis, distributed multi-agent meeting scheduling system's architecture and algorithm is proposed and its implementation details is explained. A meeting scheduling process is often in daily life and it is time consuming process. A meeting scheduling process can be solved by coordination of scheduler and by effort of attendant. A distributed multi-agent system is used to simulate real life meeting scheduling process. A new algorithm is proposed with this architecture. In this way, meeting scheduling process which is repetitive and time consuming for user is performed by agent that act on behalf of user.

As a solution architecture of meeting scheduling problem a distributed multi-agent system, a network of agents, is used. Each agent in a system represent different individual. Also meeting place information is represented with another type of agent. Meeting scheduling process is thought as an distributed search. A collaborative meeting scheduling algorithm is proposed for the solution. In meeting scheduling algorithm, solution is achieved by proposals and counter- proposals. The location agent, in the system, behave like an invitee.

JATLite, provide communication of agent and provide template to create agent, is used in implementation of this system. JATLite provide infrastructure for the distributed multi-agent system. JATLite also provide very flexible infrastructure for the communication of the agent.

KQML , data and information exchange language, is used as an message language at messaging between agents in the system. A new KQML performatives are proposed for the implementation of algorithms.

1. GİRİŞ

Teknolojik gelişmelerin artması ile birlikte günlük yaşamda ve özellikle de çalışma hayatında bilgisayarlaşma gittikçe artmaktadır. Daha önceden kişiler tarafından takip edilen ve yürütülen bazı işler yazılım uygulamaları tarafından yürütülmektedir. Bu yazılım uygulamalarından en ünlüleri kişi adına özerk olarak hareket eden etmenler kullanılarak yazılan uygulamalardır.

Bir diğer önemli teknolojik gelişme ise bilgisayar ağlarının-en ünlüsü internet-çok gelişmesi ki bu ister istemez yeni yazılımların çok heterojen ortamlar ile iletişim halinde olmasını gerektirmektedir. Ağların gelişmesinin bir diğer sonucu ise dağıtık ortamlarda çalışan, kullanıcıya esneklikler sunan yani bir yere bağımlı olmayan uygulamalara ihtiyaç duyulmasıdır. Bu ise ister istemez bizi çok farklı ortamlarda iletişim halinde olabilen esnek bir varlık olan etmenler kullanılarak gerçekleştirilebilir.

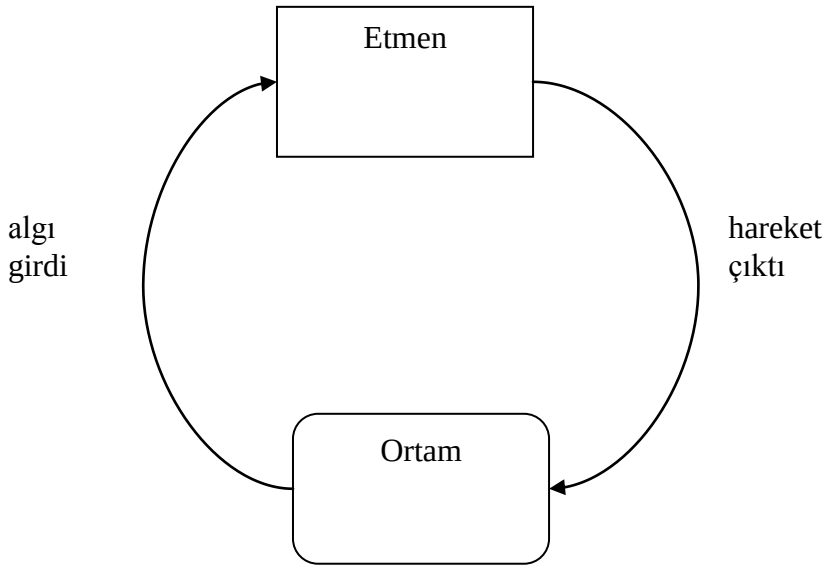
Toplantı planlama işlemi günümüzde çalışma hayatında zaman tüketen, uzun süren ve kendi kendini tekrar eden bir süreçtir. Toplantı planlama işlemin başarı ile sonuçlanabilmesi için bütün katılımcıların katkıları gerekmektedir ki bu işlemin dağıtık özelliğini göstermektedir. Etmenlerin kişi adına bağımsız hareket etme ve dağıtık özelliklerini dikkate alırsak dağıtık çoklu etmen sisteminin bir toplantı planlama işlemi sürecine katılan insanların oluşturduğu topluluğu simüle ettiği görülebilir. Dağıtık Çoklu Etmen Toplantı Planlama Sisteminde kullanıcının tek yapacağı toplantı isteklerini girmektir daha sonra kendi adına hareket eden etmen bütün işlemleri onun adına yürütür. Bu tezde, etmen oluşturulması ve haberleşme için JATLite kullanılarak geliştirilen Dağıtık Çoklu Etmen Toplantı Planlama Sisteminin yapısı ve uygulama ayrıntıları açıklanacaktır.

2. YAZILIM ETMENLERİ

2.1. Etmen Nedir?

Evrensel olarak kabul edilmiş bir etmen tanımı bulunmamaktadır; bu konu üzerindeki tartışmalar devam etmektedir. Aslında özerkliğin etmen tanımının merkezinde olduğu genel olarak kabul görmüştür fakat bunun ötesinde çok az bir uzlaşı vardır. Zorluk etmen ile ilgilendirilmiş değişik özelliklerin değişik alanlarda içersinde farklı önemlere sahip olmalarıdır. Bazı uygulamalar için etmenin tecrübelerinden öğrenme yeteneği çok önemli iken başka uygulamalar için bu pek de önemli olmamaktadır.

Etmen bir ortamda bulunan ve tasarım amacına ulaşmak için ortamda özerk hareket etme yeteneğine sahip bilgisayar sistemidir[1]. Özerklik konusunu biraz daha açarsak. Etmen insan veya diğer sistemlerin müdahalesi olmadan hareket etme yeteneğine sahiptir: kendi iç durumu ve hareketleri üzerinde kontrolü vardır. Bu tanıma uygun en basit ve genel etmen tipi olarak aşağıdaki Şekil 2.1'i alabiliriz. Etmen ortamı etkilemek için bir hareket çıktısı üretmektedir.



Şekil 2.1 Ortam içersindeki etmen

Bu özellikler dışında etmenlerde genelde aşağıdaki özellikler de bulunabilir[2]:

- Reaktif : Ortamda olan değişimleri algılayan ve yanıt verebilen
- Amaç yönelimli : Basit bir şekilde ortama yanıt veren değil amaç doğrultusunda hareket eden.
- İletişim halinde : Diğer etmenler ile ve belki de insan ile iletişim kurabilen.
- Öğrenen : Daha önceki tecrübelerinden yola çıkarak davranışlarını değiştirebilen.
- Hareketli : Kendisini bir makineden başka makineye taşıyabilen.
- Esnek : Hareketleri katı ve kesin değil, değişebilen.
- Akıllı

2.2. Etmenlerin Uygulama Alanları

Etmenlerin çok çeşitli uygulama alanları bulunmaktadır. Bunlardan bir kısmı aşağıda bulunmaktadır:

- Bilgi Toplama
- E-Ticaret
- Gömülü sistemler ve gerçek zamanlı uygulamalar
- Dağıtık hesaplama
- Ağ servisleri
- İş akışları
- İzleme ve uyarma

2.3. Akıllı Etmenler

Tasarım amacına ulaşmak için esnek özerk hareket yeteneğine sahip etmene akıllı etmen denir. Esneklik aşağıdaki 3 anlamdadır;

Reaktivite : Akıllı etmen ortamını algılama ve ortamında olan değişiklere tasarım amacını gerçekleştirmek için zamanında yanıt verme yeteneğine sahiptir.

Aktiflik : Akıllı etmen tasarım amacını gerçekleştirmek için amaç yönelimli davranış şekli ile inisiyatif alma yeteneğine sahiptir.

Sosyal yetenek : Akıllı etmen tasarım amacını gerçekleştirmek için diğer etmenler(ve insanlar) ile etkileşme yeteneğine sahiptir.

Bu özellikler ilk göründüklerinden daha fazla anlam içermektedirler. Önce aktiflik, amaç yönelimli davranış, ele alınırsa, amaç yönelimli sistem kurmak çok zor değildir. JAVA metotları, PASCAL alt yordamları veya C fonksiyonları yazarak böyle bir sistem kurulabilinir. Böyle bir alt yordam yazılınca alt yordamın çalışması için önkoşul ve önkoşul gerçekleştiğinde yaratılan etki tanımlanır. Alt yordamın etkisi onu yazanın alt yordamın başarmasını istediği amacıdır. Ön koşul gerçekleştiğinde alt yordam çağrılır ve çalışma amaç gerçekleşince sonlanmış olur. Bu alt yordamın amaca ulaşmak için bir plan olduğu bir amaç yönelimli davranıştır. Bu sistemde ortamın değişmediği yani amaç ve ön koşulun alt yordam çalışırken değişmediği varsayılır.

Dinamik ortamları, ön koşul ve amacın sürekli değiştiği, düşünürsek kör bir şekilde ortamdaki değişimleri dikkate almadan alt yordamı çalıştırmak çok garip olur. Bu gibi sistemlerde etmen reaktif olmalıdır. Etmenin amacını ve amacına ulaşmak için çağırdığı yordamın önkoşulları etkileyen ve ortamında olan olaylara karşı etmen yanıt vermelidir.

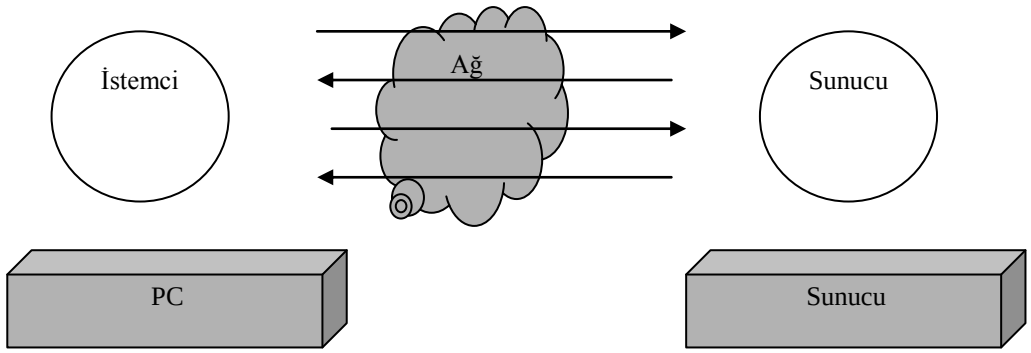
Sadece amaç yönelimli bir sistem kurmak ya da sadece reaktif, sürekli ortamdaki değişimlere yanıt veren, bir sistem kurmak zor değildir. Ama amaç yönelimli davranış ile reaktif davranış arasında etkili bir denge kuran bir sistem kurmak zordur. Etmenler ortama duyarsız sadece amaç yönelimli olmamalıdır ya da hiçbir amaca odaklanmadan sürekli ortama yanıt veren şekilde de olmamalıdır. Sonuç olarak ikisi arasında etkili bir denge kurulmalıdır ki böyle bir denge insan yaşamında bile zor kurulmaktadır.

Esnekliğin bir parçası da sosyal yetenektir. Her gün milyonlarca bilgisayarın bilgilerini değiştirdikleri bir ortamda sosyal yetenekten bahsetmek gereksiz gibi görülebilir ancak bilgilerin değişilmesi sosyal yetenek değildir. İnsandan düşünülürse; çok az kişi her birinin ayrı amacı olan ve özerk olan bireyler ile işbirliği yapmadan kendi amaçlarına ulaşabilir. Bu gibi durumlarda amaca ulaşabilmek için diğerleri ile işbirliği ve pazarlık yapılmalıdır. Diğerlerinin amaçlarını anlayarak, bazen onlara istenilenin yaptırılması veya iş birliği için değişik tekliflerde bulunulur. Benzer şekildeki sosyal yeteneklere akıllı etmenler sahip olmalıdır.

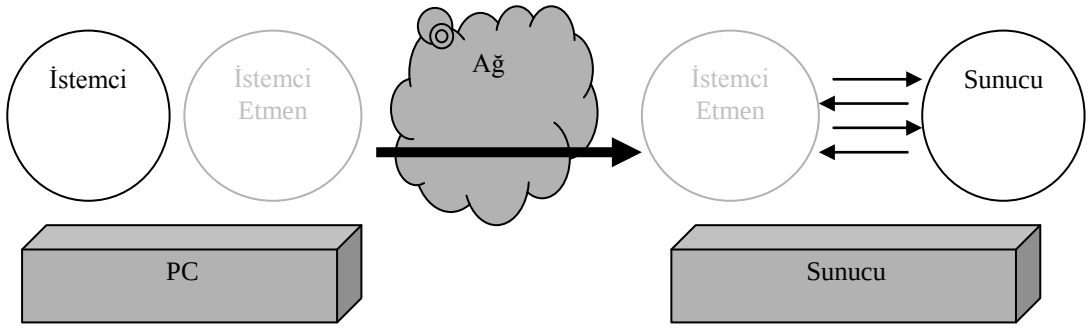
2.4. Hareketli Etmenler

Hareketli etmen bir makineden diğer makineye heterojen ağlarda göç edebilen programdır. Program ne zaman ve nereye göç edeceğini kendisi belirler. Etmen çalışmasına belli bir noktada ara verebilir ve kendisini başka bir makineye taşır ve çalışmasına orada devam eder.

Aşağıdaki şekillerin birincisinde klasik istemci sunucu modeli diğerinde ise hareketli etmen modeli gösterilmektedir. İstemci sunumcu modelinde istemci sunumcu ile ağ üzerinden haberleşir. Hareketli etmen modelinde ise istemci kendisini sunumcu tarafına göç ettirir ve işlemlerini o tarafta yapar.



Klasik istemci-sunucu modeli



Yeni hareketli etmen hesaplama
mantığı

Şekil 2.2 Klasik istemci-sunucu modeli ile hareketli etmen modeli karşılaştırılması

Yaptığımız tanımı biraz daha genişleterek hareketli etmenlerin varlığı üzerine bir tanım yaparsak.

Hareketli etmen, yazılım ortamında bulunan bir yazılım varlığıdır. Bazı etmen karakteristiklerini miras almıştır. Hareketli etmen şu modellerin hepsini kapsamalıdır: etmen modeli, yaşam döngüsü modeli, hesaplama modeli, güvenlik modeli, iletişim modeli ve dolaşma modeli. Bu modeller ayrı ayrı tanımlanacağı gibi iç içe geçmişte olabilir.

Hareketli etmenin içinde bulunduğu ortamı da tanımlamamız gerekir. Bu ortam hareketli etmen ortamı olarak adlandırılabilir. Hareketli etmen ortamı heterojen bilgisayarlar ağları üzerinde dağıtılmış bir bilgisayar sistemidir. Hareketli etmenin yürütülebileceği bir ortam yaratmak amacı vardır. Hareketli etmen ortamı hareketli etmen tanımı içersinde bulunan çoğu modelleri gerçekler. Ayrıca destek servisi de sağlayabilir.

Hareketli etmen tanımı içersinde bulunan modellere kısaca bakar isek

2.4.1. Etmen Modeli

Bu model hareketli etmenin etmen parçasının iç yapısını tanımlar. Etmenin ne olduğu ve ne şekilde davrandığı daha önceki bölümlerde anlatılmıştı. Hareketli etmenin akıllı olma durumuna göre etmen modeli olarak akıllı etmen de kullanılabilinir ya da etmen modeline değişik özellikler eklenebilir.

2.4.2. Yaşam Döngüsü modeli

Bu model hareketli etmenin değişik yürütme durumunu ve bir durumdan diğer bir duruma geçmesine sebep olan olayları tanımlar. Günümüzde iki etkin yaşam döngüsü modeli vardır ki bunlar : sürekli işlem modeli ve görev tabanlı modeldir. Şekilleri koyabilirsin.

Birinci modelde yaşam döngüsü başla durumu ile başlar ve sürekli işlemin yürütüldüğü çalışma durumuna ilerler ve daha sonra işlemin sonlandığı ölüm durumuna geçer. Hareketli etmen bir düğümden diğer düğüme taşınırken, işlem çalışma durumunda işaretlenir ve etmen donma durumuna geçer. Hedef düğüme varıldığı zaman tekrar kaldığı yerden çalışma durumuna girer. Bu tür yaşam döngüsü modeli en güçlü ve esnek çalışma modelidir.

Görev modeli başlama durumu ile başlar. Koşu kümlerini bağlı olarak görev ağlarında ilerler. Her bir görevin kendi durumları vardır. Ama etmen yeni bir

düğüme taşınır ise o anda yürütülen görevin içeriği kaybolur. Taşınmadan önce etmen taşındığı yerde yürüteceği ilk görevi belirler.

2.4.3. Hesaplama Modeli

Hesaplama modeli hareketli etmeninin çalışma durumunda iken nasıl yürüteceğini tanımlar. Hesaplama bir ortam içinde gerçekleşir ve işlemci biçimleri bu hesaplamayı kolaylaştırırlar. İşlemci biçimleri CPU olabileceği gibi Java sanal makinesindeki daha kavramsal yapılar da olabilir. Bu modelin bir parçası olarak ilkel komut kümesi belirlenmelidir. Bu etmenin hesaplama yeteneğini tanımlar. Bu ilkel komutları veri işleme komutları ve işlem parçacığı kontrol komutlarını içerir.

Hareketli etmeni gerçekleyen etmenin diğer modellerine ulaşımı model aracılığı ile gerçekleştirir. Dolayısıyla bu modelin yapısı diğer modelleri de etkilemektedir. Bunun için bu modelin diğer modeller üzerine olan etkilerini de değerlendirmeliyiz.

2.4.4. Güvenlik Modeli

Hareketli etmen güvenliği iki alana ayrılabilir. Birincisi konak düğümün yıkıcı hareketli etmenlerden korunması, ikincisi ise hareketli etmenlerin yıkıcı konaklardan korunmasıdır.

2.4.4.1. Konağın kötü niyetli etmenlerden korunması

Hareketli etmen sistemleri açık bir sistemdir ve bütün açık sistemler gibi konak düğüm değişik saldırılara açıktır. Bu saldırılar 4 kategoride toplanabilir.

Sızıntı : Verilerin yetkisiz taraflarca elde edilmesi

Sıkıştırma : Verilerin yetkisiz taraflarca değiştirilmesi

Kaynak Çalma : Kaynakların yetkisiz taraflarca kullanılması

Yıkıcılık : Kötü niyetli bir şekilde konağın ve verilerinin tahrip edilmesi

Klasik şifreleme, anahtarlama ve diğer teknikler gibi doğrudan sisteme girişi engelleyen teknikler hareketli etmenlerin konakta yürütülmesinin zorunlu olduğunu düşünürsek geçerli olamaz.

Standart yaklaşım olan tanımlanamayan kodların konağı girişinin engellenmesi ise esneklikten taviz verilmesi anlamına gelir. Bu durumda açık sisteme ile kesin kuralları olan sistem arasında bir denge kurulması gerekir.

2.4.4.2. Etmenin kötü niyetli konaklardan korunması

Çalışmak istediği konağın hareketli etmene ve verilerine zarar vermesi ya da hareketli etmeni yok etmesi durumudur. Hareketli etmen çalışmak istediği konağa kodunu ve verilerini yerleştirmek durumundadır ve kötü niyetli konaklar konusunda tamamen korumasızdır.

2.4.5. İletişim modeli

Hareketli etmen, doğal olarak hesaplama ortamındaki kullanıcı, başka etmenler, konak hareketli etmen ortamı ya da başka sistemler gibi varlıklarla iletişim halindedir. Bu varlıklar başka bir etmen, hareketli etmenin konağı ya da dağıtık başka bir sistem olabilir.

Protokol iletişim modelinin bir gerçeklenmesidir. Basitinden karmaşığına kadar değişik protokolü bulunmaktadır. Protokol iletişim halinde olunan varlığa göre değişebilir. Örneğin insan ile KQML kullanarak haberleşemez. Bazen farklı varlıklar ile iletişim kurabilmek için etmenler birden fazla iletişim modülüne sahip olabilirler.

2.4.6. Dolaşma modeli

Dolaşma modeli hareketli etmenin hareket yönüdür. Bu model etmenin bütün hareketlilik yönü ile ilgilidir ki bu etmenin ne şekilde taşınacağıdır. Dolaşma modelinde aşağıdakiler kesinlikle olmalıdır.

- Hareketli etmen sistemindeki bütün varlıkların isimlendirme düzenlemesi (etmenler, konaklar gibi)
- Uzak hareketli etmen ortamındaki bilgiye erişebilme
- Etmeni uzak konağa taşımaya hazırlamak için askıya alınma durumuna çevirme yeteneği
- Etmeni asılı durumda iken uzak hareketli etmen ortamına taşıma yeteneği
- Hareketli etmeni asılı durumda iken uzak konaktan alarak ve onu yeni ortamda yeniden oluşturma yeteneği

2.4.7. Hareketli Etmenlerin Avantajları

- Etkindir. Hareketli etmen çok az ağ kaynağı kullanmaktadır. Hesaplama yapacağı veri yerine kendisini taşımaktadır.

- Ağ trafiğini azaltır. Çoğu iletişim protokolü iletişim sırasında ağ güvenliğini sağlamak için pek çok etkileşim yapar, bu ise ağ trafiğini artırır. Hareketli etmen kendisini bir kere taşır ve etkileşimi gittiği yerde yapar.

- Asenkron etkileşimlidir. Görev hareketli etmen içersine kodlandığı ve hareketli etmen dağıtıldığı zaman hareketli etmen kendisini gönderen programdan bağımsız olarak asenkron çalışır.

- Heterojen ortamları destekler. Hareketli etmen sisteminin üzerine kurulduğu bilgisayarlar ya da ağlar heterojen özelliktedir.

- Uygun bir geliştirme paradigmasıdır. Hareketli etmen kullanarak dağıtık sistem kurmak çok kolaylaşmıştır. Çünkü hareketli etmen doğası gereği dağıtıktır.

- Sağlamdır ve aksaklığa dayanıklıdır. Özellikle çok dağıtık sistemlerde ters durumlara dinamik tepki verebilme yeteneği aksaklığa dayanaklı bir davranış geliştirmesine olanak tanımaktadır.

2.5. Çoklu Etmen Sistemi

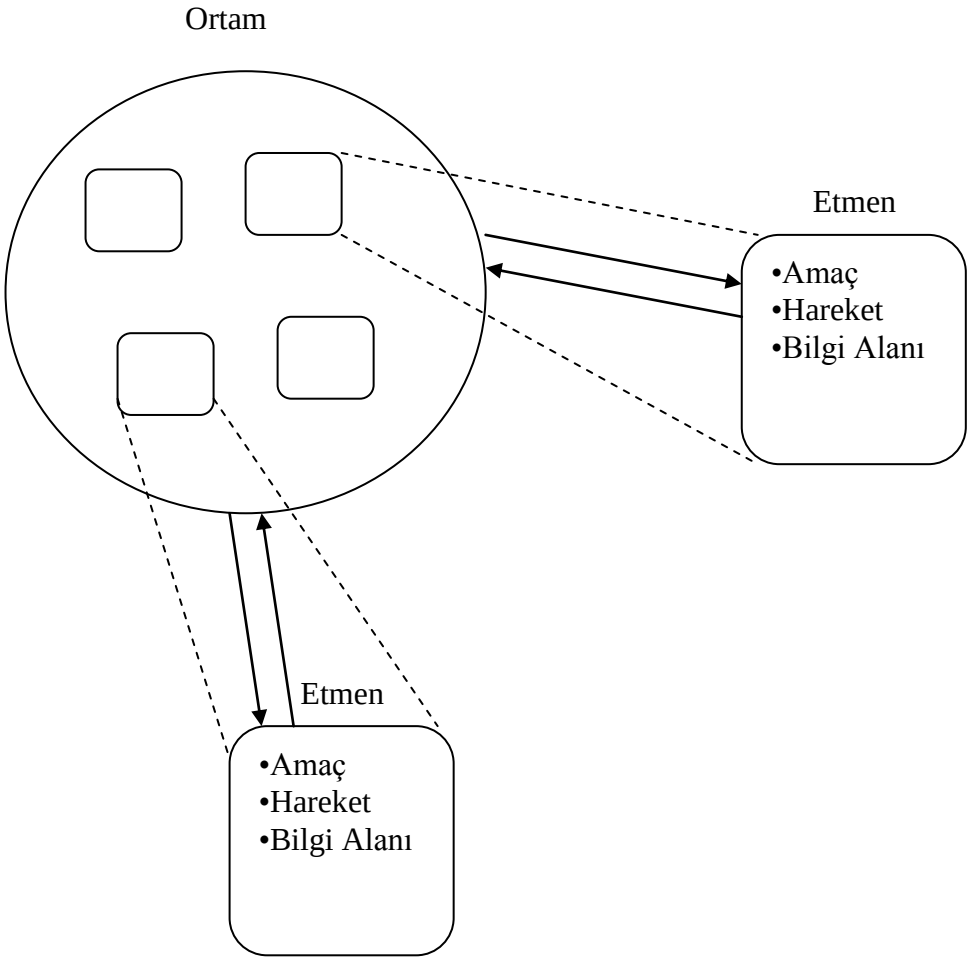
Sahip olunan bilginin sınırlılığı düşünüldüğünde tek bir etmenin her türlü problemi çözmesi beklenemez. Bazı problemlerinin çözümü için çok fazla bilgi gerekmektedir. Ayrıca bazı problemlerin çözümü için farklı uzmanlıkların ortak çalışarak sonuca ulaşması gerekmektedir. Bu ise uzmanlık alanı sınırlı bir etmen ile başarılamaz. Bunlara ek olarak günlük hayatta karşılaşılan çoğu problem doğası gereği dağıtık olmaktadır. Bu problemleri çözmek için açıktır ki çoklu etmen içeren sistemler kurmamız gerekmektedir.

Çoklu Etmen Sistemleri (ÇES) bir problemi çözmek için etkileşen gevşek bağlı problem çözücüler ağıdır ki ortaya çıkan sistem problem çözücülerin bireysel yetenekleri ya da her birinin bilgileri toplamından daha fazladır. Şekil 2.3'de de görüldüğü gibi bu problem çözücülerin her biri birer etmendir.

ÇES birimsel bir yapı sunmaktadır. Problemin büyüklüğü ve ihtiyaç duyulan etmen sayısı tahmin edilemeden tasarlanan sistemler yeni ihtiyaçlar doğrultusunda yeni etmenler eklenerek rahat bir şekilde genişletilebilir. Eğer ihtiyaç. duyuluyorsa ÇES'e ihtiyaç duyulan uzmanlığa sahip yeni etmenler eklenebilir. Sistemde kendi uzmanlığı ve bilgisi olan etmenler topluluğundan oluşur.

ÇES ile problem çözme işlemi Dağıtık Problem Çözme olarak adlandırılır. ÇES'in genel problemleri uyumlu çözebilmesi için etmenlerin kendi aralarında iletişimli,

kendi aktivitelerini yürütürken eşgüdümlü ve pazarlık yapabiliyor olmaları gerekir. Bu üç kavram çoklu etmen sistemleri için çok önemlidir.



Şekil 2.3 Çoklu etmen sistemi genel yapısı

2.5.1. Çoklu Etmen Sistemlerinde Eşgüdüm

Eşgüdüm ÇES için merkezi öneme sahiptir, eşgüdüm olmadan etkileşimin kazandırdığı yarar ortadan kalkar ve etmen grubu karmakarışık davranışlar gösteren bireyler toplamına dönüşür. Çoklu etmenlerin eşgüdümü ihtiyaç duymasına değişik sebepler vardır. Bunların bir kısmı aşağıda belirtilmiştir.

- Karmakarışıklığı engellemek. Hiçbir etmen dahil olduğu sistemde global bir bakış açısına sahip değildir. Zaten çok karmaşık ve büyük sistemlerde de global bakış açısına sahip olmak gerçekleşebilir değildir.

- ÇES içindeki her etmenin farklı yetenekleri ve uzmanlıkları vardır. Dolayısıyla nasıl ki bir hastayı iyileştirmek için cerrahlar, hemşireler ve uzman doktorlar

eşgüdüm içinde hareket ediyorsa etmenlerin de eşgüdüm içinde harekete ihtiyacı vardır.

- Etmenlerin hareketleri çoğunlukla birbirine bağlıdır ve bunun için bazen etmenler kendi görevlerini yürütmeden önce diğer etmenin görevini bitirmesini beklemek ihtiyacında olabilir. Birbirine bağlı hareketler eşgüdüm içinde yapılmalıdır.

Eşgüdüm problemini çözmek için çok farklı yaklaşımlar bulunmaktadır. Aşağıda eşgüdüm problemini çözmek için varolan yaklaşımlardan bir kısmı belirtilmiştir.

2.5.1.1. Örgütsel Eşgüdüm

Uyumlu etmenlerin uyumlu davranışlarını sağlamanın en basit yolu herhalde etmen grubundan bir etmenin sistemin genel bilgisine sahip olması ve sisteme daha geniş bakabilmesidir ki bu örgütsel, sıradüzensel yapıyı gerçekleştirmektir. Bu basit bir eşgüdüm yöntemidir, bu görevleri ve kaynakları efendi etmenlerin dağıttığı klasik efendi/köle yapısıdır. Efendi ya da kontrolcü etmen bireysel uyumu sağlamak için etmen gruplarının bilgilerini toplar, plan yapar, etmenlere görevleri dağıtır.

Ancak böyle bir sistem yaratmak gerçek uygulamalarda çokda pratik değildir. Çünkü sistemde bütün etmenlerin niyetlerine, inançlarına ve bilgilerine sahip bir kontrolcü yaratmak çok zordur.

2.5.1.2. Eşgüdüm için Sözleşme – Sözleşme Ağ Protokolü

Etmenlere görev ve kaynak atama için kullanılan en ünlü eşgüdüm tekniğidir. Belirlenmiş örgütsel yapısı ise Sözleşme Ağ Protokolüdür(SAP).

Bu yaklaşımda, merkezi olmayan pazar yapısı varsayılmıştır ve etmenlerin iki rolü vardır; yönetici ve yüklenici. Temel biçimi şu şekildedir. Eğer etmen kendisine yüklenilen problemi kendi yerel kaynakları ve uzmanlığı ile çözemez ise, problemi alt problemlere parçalar ve gerekli kaynak ve uzmanlığa sahip ve bu problemleri çözmeye niyetli etmenleri bulmaya çalışır. Alt problemleri atama problemi sözleşme mekanizması ile çözülür. Sözleşme mekanizması ise şu şekilde işler; yönetici etmen tarafından sözleşme duyurusu yapılır. Bu duyuruya karşılık yüklenici etmenlerden teklifler gelir. Daha sonra yönetici etmen gelen teklifleri değerlendir ve en uygun teklif sahibi ile alt problemin sözleşmesini yapar.

SAP pek çok uygulama tarafından kullanılmaktadır. Bu eşgüdüm yönteminin avantajı dinamik görev dağılımını ve doğal olarak iş yükü dengesini sağlamaktır.

Dezavantaj olarak sistemdeki karmaşıklığı fark edememesidir. Ayrıca sistemdeki etmenler iletişime çok istekli olarak düşünülmektedir.

2.5.1.3. Eşgüdüm için Çoklu Etmen Planlama

Daha farklı bir yaklaşım olarak, çoklu düğüm eşgüdümü problemi planlama problemi olarak da düşünülebilir. Hava trafik kontrol gibi çok kritik uygulamalarda çoklu etmen planlama çelişkili ve karmaşık durumlarından kaçınmaya yoğunlaşır. Çoklu etmen planı yaparak, düğümler bütün hareketlerini ve etkileşimlerini daha önceden belirlerler. İki tip çoklu-etmen planlama vardır

Merkezi çoklu etmen planlamada ayrı etmenler kendi bireysel planlarını oluştururlar ve bu planları merkezi eşgüdümcüye gönderirler. Merkezi eşgüdümcü bütün planları analiz eder ve potansiyel karmaşıklıkları tespit eder. Merkezi eşgüdümcü eşzamanlı hareket edilecek kritik yerleri ve mesaj gönderip beklenerek yapılacak eşzamanlı hareketleri belirler. Daha sonra bu planları birleştirerek çoklu-etmen planı oluşturulur.

Dağıtık çoklu etmen planlama tekniğinde merkezi eşgüdümcü kullanılmaz. Onun yerine bu teknikte etmenler birbirlerinin planlarını modellerler. Etmenler bütün karmaşıklıklar ortadan kalkasıya kadar diğer etmenler ile haberleşirler ve kendi planlarını ve diğerlerinin modellerini güncellerler.

2.5.1.4. Eşgüdüm için Toplumsal Kanunlar

Hava kontrol ve şehir trafiği gibi alanlarda akıllı etmenlerin eşgüdümünü sağlamaya çalışan araştırmacılar, sıradan durumlarda alışılmamış durumlara göre eşgüdümü sağlamanın daha kolay olduğunu fark ettiler.

Eğer bütün etmenler kendi üyelerinin amaçlarını, hareketlerini ve etkileşimlerini tam olarak bilirlerse, şu anda her bir etmenin ne yaptığını ve ilerde ne yapacağını kesin olarak bilebilir. Ve bu durumda etmen karmaşıklıklardan ve gereksiz çabalardan uzak durur ve etmenlerin eşgüdümü mükemmel olarak sağlanmış olur.

Ama böyle kesin hareket ve tepki bilgilerini bilmek ancak sıradan durumlarda mümkündür. Bir etmen diğer etmenin hareket bilgilerini ancak trafik kuralları gibi toplumsal kurallara göre hareket ettiğinde bilir.

2.5.2. Pazarlık Teknikleri

Pazarlık üzerine tam uzlaşılmış bir tanım olmamak ile birlikte genel olarak şu tanımı kullanabiliriz: pazarlık bir konu üzerinde karşılıklı kabul edilebilir bir anlaşmaya

ulaşmak için yapılan toplu etmen iletişimidir. Anlaşma fiyat ya da zaman gibi değişik konular üzerine olabilir. Pazarlıktaki temel fikir anlaşmaya ulaşmaktır.

Pazarlık yarışmacı ya da işbirlikçi olabilir. Yarışmacı pazarlık etmenlerin farklı ve bağımsız amaçlarının olduğu durumlarda kullanılır. İşbirlikçi pazarlık ise merkezi olarak tasarlanmış bir amaç olduğu zaman kullanılır. Aşağıda bazı pazarlık teknikleri anlatılacak.

2.5.2.1. Oyun Teorisi Kullanarak Yarışmacı Pazarlık

Oyun teorisi kullanılarak akılcı ve özerk etmenlerin eşgüdümü açık bir eşgüdüm mekanizması olmadan sağlanmıştır. Etmenlerin hayırsever olduğu varsayılmıştır. Oyun teorisindeki anahtar kavramlar aşağıdaki gibidir:

Fayda fonksiyonları : Fayda amaca ulaşmanın değeri ile amaca ulaşmak için ödenen miktar arasındaki farktır.

Anlaşma uzayı : Anlaşma etmenin faydaya göre yapacağı harekettir.

Strateji ve Pazarlık Protokolü : Pazarlık protokolü pazarlık nasıl ve ne zaman sonlanacak gibi pazarlığı yöneten kuralları tanımlar.

Gerçek pazarlık ise şu şekilde işler: Her bir etmenin etkileşimlerinin sonuçlarının fayda değerleri ile ödeme matrisi oluşturur. Bu ödeme matrisi pazarlığa giren iki tarafa da açık bir bilgidir. Pazarlık işlemi, teklif ve etmenin faydasını maksimize eden anlaşmayı seçerek önerdiği karşı tekliflerden oluşan bir etkileşimli işlemdir. Her bir etmenin kendi faydasını maksimize edeceği varsayılmaktadır. Etmen pazarlığın her aşamasında diğer etmenlerin tekliflerini kendi pazarlık stratejisine göre değerlendirir.

2.5.2.2. İnsandan Esinlenme Yaklaşımı Kullanarak Yarışmacı Pazarlık

İnsanların etkileşimlerinin çoğunda belli bir dereceye kadar pazarlık gerekmektedir. Böyle bir durumda doğal olarak araştırmacılar insandan etkilenmektedirler. Bu şekilde insandan esinlenen örnek bir sistem PERSUADER'dir. Sistem üç etmen içerir: şirket, sendika ve görevi diğer etmenleri kabul edilebilir bir uzlaşmaya ulaştırmaya çalışmak olan arabulucu. Sistemin girdisi şirket ve sendikanın karmaşık amaçlar kümesi ve tartışma içerikleridir. En son çıktı ise uzlaşı veya belli bir pazarlık döngüsünden sonra başarısızlıktır. Yüksek seviyeli görevleri ise:

- İlk uzlaşıyı önermek

- Ret edilen uzlaşıyı geliştirmek
- Tarafları takip ederek, onların uzlaşıyı değerlendirmelerini değiştirmek. (geçmiş pazarlık tecrübeleri kullanılarak)

2.5.2.3. Döngü Modeli kullanılarak İşbirlikçi Pazarlık

Diğer pazarlık modelleri önerilerindeki sınırlamalara dikkat çekilerek önerilmiştir. Modelin döngülü yapısı ile karmaşıklı çözümü yapılmak istenmiştir.

Genel strateji: pazarlık etmenlerin birinin, bir kısmının veya hepsinin önerilerini yapmaları ile başlar. Etmenler önerileri değerlendirirler ve önerileri kendi önceliklerine göre kontrol ederler ve onları öneriler tarafından ihlal edilen öncelikleri listeleyerek değerlendirirler. Etmenler daha sonra diğer etmenlerin öncelikleri hakkındaki bilgilerini güncellerler ve pazarlık döngüsü kaldığı yerden yeni öneri veya öneriler ile devam eder. Eşzamanlı çalışan karmaşıklık çözüm döngüsü ile karmaşıklık çözülür.

2.5.3. Çoklu Etmen Sistemlerinde İletişim

Yukarda ÇES ortamlarındaki eşgüdüm teknikleri kısaca anlatıldı. Eşgüdümü sağlamak için etmenler pazarlık edebilir ancak kesinlikle etkileşmeleri ve bilgi değiştirmeleri gerekir bu ise iletişim demektir. İletişimin önemi vurgulanmak için şöyle tanımlar yapılmaktadır: "varlık ancak doğru bir şekilde Etmen İletişim Dili(EİD) ile haberleşiyorsa etmendir". EİD speech act theory'iden esinlenerek türetilmiştir. Speech act theory insanın günlük hayatında günlük görevlerini başarmak için dili nasıl kullandığını anlamaya çalışarak geliştirilmiştir

Speech Act Teori: Speech Act teorisi insanın (veya makinenin) telaffuzunu konuşanın niyetine, dinleyicideki etkisine ve diğer fiziksel telaffuz göstergelerine bağlı olarak farklı kategorilere ayırır. Speech act insan için tanımlandığından, insan adına hareket eden etmenlerin iletişiminde de etkin olarak kullanılabilir. Speech act'ın üç bakış açısı vardır:

İfade tarzı : Konuşmacının fiziksel telaffuzu

Söylenmek istenen anlam : Konuşmacının telaffuzunun niyet edilen anlamı

İfade sonucu : İfade tarzının sonucunda ortaya çıkan sonuç hareketi

Örneğin Bülent, Mahire "Lütfen kapıyı kapat" dediğinde bu hareket Bülent tarafından yaratılan bir fiziksel ses, Bülent'in mesaj, emir ya da istek için niyetini ve

her şey yolunda giderse kapının kapanmasını içermektedir. İnsan iletişiminden mesajdaki niyet her zaman açık değildir ancak etmenler arası iletişimde mesaj tipleri çok açıktır. Speech act teorisi belli sınıflardaki telaffuzların ifade tarzlarının gücünü belirtmek için performative terimini kullanır. Örneğin söz, ısrar, söyleme birer performative yüklemdirler.

ÇES'de iletişim problemine olan çözümler , iletişimsizlikten, geçici EİD'ye ve standart EİD'ye kadar uzanan bir çeşitlilik göstermektedir.

2.5.3.1. İletişimsizlik ya da İlkel İletişim

Bazı durumlarda etmenler akıllıca diğer etmenlerin planlarını iletişim olmadan tahmin edebilirler. Bu durum etmenlerin amaçları karışmıyorsa çok iyi işler. Etmenin sonuç almak için sadece genel bilgiyi kullanmasının yeterli olduğu durumlarda da iletişimsizlik yeterli bir çözüm olmaktadır.

İlkel İletişim sınırlı sayıdaki sabit sinyal kümesi ile haberleşen etmenler için düşünülebilir. Bu değerlerin sınırlı sayıda olması etmenler arası eşgüdümü de etkiler. Bu durumda karmaşık niyetler ve bilgiler sinyaller ile ifade edilemez.

2.5.3.2. Geçici Etmen İletişim Dili

Bugünlere kadar pek çok ÇES uygulamaları geçici etmen iletişim dili içersindeki geçici performative'ler kümesini kullanmaktadır. Çoğunluğunun açık bir EİD'si bile bulunmamaktadır ve bilgiyi paylaşılan veri yapıları halinde iletilmektedirler. Geçici yaklaşımların kusuru ise farklı kişiler tarafından geliştirilmiş etmen uygulamalarının etkileşimli çalışmaları mümkün değildir.

Plan Geçirme ve Mesaj Geçirme: Plan geçirme yaklaşımında etmen A1 kendi planını etmen A2'ye iletir ve A2 kendi planını geri A1'e iletir. Hangi plan önce ulaşır ise o kabul edilir. Bu yöntem bir iletişim bütün planın iletilmesi pahalı olduğu için ve başka sınırlamalardan dolayı pek tercih edilmez. Mesaj geçirme yönteminde ise etmenler birbirlerine mesajlar gönderirler. Pek çok sistem kesin ve belli içeriklere sahip protokol ile mesaj geçirme yöntemini kullanmaktadırlar.

Kara Tahta Kullanarak Bilgi Değiştirme : Kara tahta sıkça kullanıldığı gibi paylaşılan bir bellek modelidir. Kara tahta etmenlerin mesajlarını yazdıkları, sonucun bir kısmını gönderdikleri ve bilgi edindikleri bir havuzdur.

2.5.3.2. Standart Etmen İletişim Dili

EİD standardına sahip olmak bir zorlama değil bir gerekliliktir. FIPA (Foundation for Intelligent Physical Agents) standartlar örgütü böyle bir standarda ulaşmaya çalışmaktadır. Şu an için önemli olarak iki standart önerilmiştir: KQML(Knowledge Query and Manipulation Language) ve FIPA-ACL(FIPA-Agent Communication Language). ARPA Knowledge Sharing Effort 'un büyük çabalarının bir parçası olan KQML'i ilerleyen bölümlerde ayrıntılı olarak inceleyeceğiz.

FIPA-ACL FIPA'nın açık etmen mimarisi ile ilişkili bir etmen iletişim dilidir. FIPA-ACL içerik dili ile olan ilişkiyi korumaktadır ve herhangi bir içerik dili ve ontoloji spesifikasyonu yaklaşımı ile çalışmak için tasarlanmıştır. FIPA-ACL'in amaçta ve sözdiziminde varolan benzerliğin dışında KQML ile pek çok önemli ayrımları vardır.

2.5.4. Çoklu Etmen Sistemlerinin Yetenekleri

Kaynak sınırlaması ya da performans sorunu nedeni ile merkezi bir etmen için çok büyük olabilecek problemler çok kolay çözülebilir.

Var olan çoklu sistemlerin bir birine bağlanmasına ya da birlikte işlemesine izin verir. Değişimleri yakalamak için var olan sistemlerin tekrar yazılması çok pahalı ve zordur. Bu durumda böyle bir sistemin işlerliğini koruması için daha geniş bir etmen topluğu ile birleştirmek çok yararlı olur.

Doğal olarak etkileşen özerk parçalar, etmenler, topluluğu olarak değerlendirilebilecek problemlere kolay çözümler sağlar. Örneğin bu tezinde konusu olan toplantı planlaması işinde kullanıcının takvimin yöneten planlama etmeni özerktir ve diğer kullanıcıların takvimlerini yöneten benzer etmenler ile etkileşim halindedir.

Dağıtılmış bilgi kaynaklarını etkili bir şekilde kullanarak çözümler sağlar. Örneğin algılayıcı ağları, internetten bilgi toplamak gibi.

Uzmanların dağıtılmış olduğu durumlarda çözümler sağlar. Örneğin koşut zamanlı mühendislik, sağlık ya da üretim durumlarında.

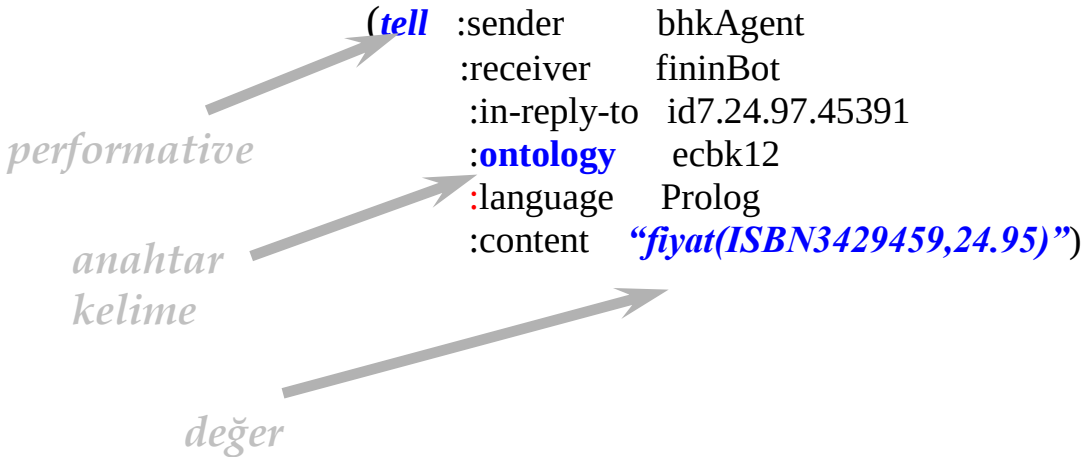
Koşut zamanlı hesaplama olduğu için hesaplama etkinliği, güvenirlik , problem için çalışan etmen sayısı kolayca genişletilebilme, sağlamlık, korunabilirlik ,yanıt verebilirlik , esneklik, yeniden kullanılabilirlik açılarından performansı arttırır.

2.6. Knowledge Query and Manipulation Language (KQML)

Veri ve bilgi alışverişi için kullanılan bir dil ve protokol olan KQML (Knowledges Query and Manipulation Language) yeniden kullanılabilinir ve paylaşılabilinir

büyük tabanlı bilgi tabanları oluşturmayı amaç edinmiş ARPA Knowledge Sharing Effort 'un büyük çabalarının bir parçasıdır[3]. KQML akıllı sistemlerle etkileşim içinde olan uygulama programlarında ya da bir veya daha fazla akıllı sisteminin ortak bir problemi çözmesini desteklemek için bilgi paylaşımları durumunda kullanılabilinir. KQML hem bir mesaj biçimi hem de mesaja dayalı etmenler arası çalışma zamanlı bilgi paylaşımını destekleyen bir protokoldür.

Asenkron ve özerk olmaları açısından KQML etmen tabanlı programların iletişiminde kullanılan en faydalı dildir. Özerkliğin anlamı etmenlerin farklı hatta çakışan gündemlerinin olabilmesidir ki KQML mesajının anlamı mesaj alıcısı yerine göndericisi üzerindeki sınırlamalar baz alınarak tanımlanır. Bu, mesaj alıcısına fonksiyonlarına uygun düşen davranışları seçme olanağı verir. Tabii ki etmenlerin mümkün olduğunca beraber çalışabiliyor olmaları istenir ancak tıpkı insan yaşamında olduğu gibi tam anlamıyla bir birliktelik çoğu zaman mümkün olmaz.



Şekil 2.4. KQML mesajının genel yapısı

Performative olarak adlandırılan KQML mesajları BNF biçimindeki, Şekil 2.5'deki gibi, sözdizimince tanımlanan ASCII dizgilerinden oluşmaktadır. KQML'in sözdizimi dengeli parantez listesine dayanmaktadır. Listenin ilk elamanı performative'dir, kalan elemanlar ise anahtar kelime/değer ikilisi şeklinde performative'in değişkenleridir. Performative içindeki parametreler anahtar kelimeler ile ayırt edilmektedirler, performative içindeki yerlerinden değil. Bu yüzden yerden bağımsızdırlar. Anahtar kelimeler ile başlarlar ve daha sonra anahtar kelimeye karşılık gelen içerikleri(değer) ile devam etmek zorundadırlar. Şekil 2.4'de örnek bir KQML mesajı ve bu mesaj içindeki performative, anahtar kelime ve değerler gösterilmiştir

Performative'ler kümesi dilin çekirdeğini oluşturur. Performative KQML ile iletişim kuran etmen ile olan etkileşimin şeklini belirler. Performative'lerin birincil fonksiyonu mesajların dağıtım protokolünü belirlemektir. Performative :content'in iddia, sorgu, yorum veya konuşma hareketi üzerine karşılıklı anlaşılmış şey olduğunu belirler. Performative ayrıca gönderenin yanıtın ne şekilde gönderilmesini istediğini de tanımlar.

```
<performative>::= (<word> {<whitespace> :<word> <whitespace> <expression>}*)

<expression> ::= <word> | <quotation> | <string> |

(<word> {<whitespace> <expression>}*)

<word> ::= <character><character>*

<character> ::= <alphabetic> | <numeric> | <special>

<special> ::= < | > | = | + | - | * | / | & | ^ | ~ | _ | @ | $ | % | : | . | ! | ?

<quotation> ::= '<expr> | `<comma-expr>

<comma-expr> ::= <word> | <quotation> | <string> | ,<comma-expr> |

(<word> {<whitespace> <comma-expr>}*)

<string> ::= "<stringchar>*" | #<digit><digit>*"<ascii>*

<stringchar> ::= \
```

Şekil 2.5 KQML sözdizimi BNF'de

Tablo 2.1'de performative'lerde kullanılan ayırtılmış anahtar kelimeler ve bunların kullanılış amaçları ve anlamları bulunmaktadır.

Kavramsal olarak ise KQML mesajları performative'leri, onun ilgili olduğu değişkeni (mesajın gerçek içeriğini de içeren) ve taşıma değişkenleri kümesinden(içeriği ve belki de gönderici ve alcıyı da tanımlar) oluşur. Örneğin IBM'in hisse senedi fiyatının sorgusunu sunan bir mesaj aşağıdaki şekilde kodlanabilir:

(ask-one

:content (PRICE IBM ?price)
:receiver stock-server
:language LPROLOG
:ontology NYSE-TICKS)

Tablo 2.1 Performative'lerde ayırtılmış anahtar kelimeler ve anlamları

Anahtar kelime	Anlamı
:content	Performative'in belirttiği tutum hakkında bilgi
:force	Gönderen ilerde herhangi bir zamanda performative'in anlamını ret edecek mi
:in-reply-to	Yanıtta beklenen etiket
:language	:content parametresinde sunulan içeriğin dilinin adı. KIF, LISP, SQL vb.
:ontology	:content parametresinde kullanılan ontolojinin adı
:receiver	Performative'in gerçek alıcısı
:reply-with	Gönderen yanıt bekliyorsa bu yanıtı etiketlemek için gönderilen bilgi
:sender	Performative'in gerçek göndericisi

Bu mesajda KQML performative'i ask-one'dır, content (PRICE IBM ?price)'dır, ontology ise NYSE-TICKS'dir, mesajın alıcısı stock-server şeklinde tanımlanan server'dir ve sorgu LPROLOG dilinde yazılmıştır.

KQML ayırtılmış performative'ler kümesi tanımlamış olsa da bu kapalı ya da en küçük bir set değildir. KQML etmeni belki bir bu performative'lerin birkaç tanesini seçerek kullanıyor olabilir. Bu küme genişleyebilir; etmen topluluğu yorumu ve onunla ilgili protokol konusunda anlaşarak yeni performative'ler ekleyebilirler. Fakat ayırtılmış performative gerçeklenirken, o performative'in standartlarına uygun bir şekilde gerçeklenmelidir.

Tablo 2.2 KQML performative'lerinin kategorileri ve kategorilerdeki performative'ler

Kategori Adı	Performative'ler
Temel Bilgisel Performative'ler	Tell, deny, untell
Veritabanı Performative'leri	Delete, delete-one, delete-all, insert
Temel Yanıt Performative'leri	Error,sorry
Temel Sorgulama Performative'leri	Reply, ask-one, evaulate, ask-if, ask-about, ask-all, sorry
Çoklu-Yanıt Sorgulama Performative'leri	Stream-all, stream-about,eos
Temel Effector Performative'leri	Achieve, unachieve
Generator Performative'ler	Standby, ready, next, rest, discard, generator
Yetenek-Tanımlama Performative'leri	Advertise
Bildirme Performative'leri	Subcribe, monitor
Ağ Performative'leri	Register, unregister, forward, broadcast, pipe, break, transport-address
Kolaylaştırma Performative'leri	Broker-one, broker-all, recommend-one, recommend-all, recruit-one, recruit-all

Tablo 2.2'de ayırtılmış KQML performativeleri ve bunların hangi kategorilerde oldukları gösterilmektedir.

2.7. Java Programlama Dili

Etmen tabanlı uygulamaların dil sınırlaması yoktur. Dolayısıyla etmen yazarken bilinen herhangi bir dil kullanılabilir. Dağıtık ortamlarda etmen tabanlı bir yapı kuruluyor ise ortamdan ve platformdan bağımsız bir mimari kurulmalıdır. Ayrıca seçilen dilin, hareketliliğe ve parelel çalışmaya da destek veriyor olması gerekiyor. Javanın platformdan bağımsız olması , internet ortamında istemci uygulaması için tarayıcı tarafından yorumlanabiliyor olması , uzak makinelerdeki kaynakların kullanımı için hazır fonksiyonlar sunması (RMI) , ağ haberleşmesi için socket programlamayı desteklemesi etmen yazarken en çok kullanılan dil olmasını sağlamıştır.

Ayrıca burada tasarlanan sistemde daha önce de anlatıldığı gibi etmen haberleşme ve yaratma amaçlı olarak JATLite kullanılmaktadır. Bu ise JATLite ile uyumlu çalışabilecek bir programla dilinin seçilmesini gerektiriyor ki bu ise Java programlama dilinden başka bir dil değildir.

Java basit bir programlama dilidir. Java C++ programlama dilinin hiç kullanılmayan ya da etkin kullanılmayan bölümlerinin çıkarılmasından oluşan bir zemin üzerinden geliştirilmiştir. Yazılım geliştirme mühendisliğinde kod geliştirmekten çok kod bakımı daha fazla zaman aldığı için anlaşılabilir bir kod için yapılan değişiklik yazılım fiyatlarını düşürmeye yardımcı olmuştur.

Java nesneye yönelik bir programlama dilidir. Nesneye yönelik programlama dili yeniden kullanılabilen yazılım parçacıkları yaratmaya yardımcı olur.

Java dağıtıktır. C++ ve C'den farklı olarak Java özellikle ağ ortamlarında çalışması için tasarlanmıştır. Java TCP/IP protokolleri kullanarak iletişim için geniş bir sınıflar kütüphanesine sahiptir. Java kodları C veya C++ kodlarının yerel dosyaları işlediği gibi URL ile kaynakları kolayca işleyebilir.

Java ikili kodları yorumlanır. Java sınıf kaynak kodları Java derleyicisi tarafından sekiz ikil kodlara çevrilince bu sekiz ikil dosyası Java yorumlayıcısı veya Java uyumlu tarayıcı olan bütün makinelerde çalışabilir. Bu ise Java kodlarının kullanıcının platformundan bağımsız olarak yazılmasını sağlar.

Java sağlam yazılımdır. Sağlam yazılım programlama yanlışları veya mantık hatalarından dolayı kolayca kırılmayan yazılımdır. Sağlam yazılımı özendiren

programlama dili programcıya kod yazarken çok sıkı kısıtlamalar getirir. Java'da tipler çok sıkıdır. Java işaretçi aritmetiğini desteklemez.

Java güvenlidir. Java ağ ortamlarında çalıştığı için yazılım geliştiriciler güvenliği daha çok dikkate almalıdırlar.

Java ağ üzerindeki uygulamaları desteklemek için tasarlanmıştır. Genel olarak ağlar değişik yapıdaki işletim sistemleri ve farklı CPU'lardan oluşmaktadır. Dolayısıyla Java uygulamalarının ağ üzerinde herhangi bir yerde yürütülebilmesi için derleyici yapı bağımsız bir nesne dosyası üretir.

Java yapısal tarafsızdır. Java derleyicisi tarafından yaratılan sekiz ikil kodlar Java uyumlu tarayıcıların veya Java yorumlayıcısının yüklü olduğu makinelere gönderilebilir.

Java taşınabilir. Yapısal tarafsız olmak taşınabilir olmak için büyük bir yardımcı etkendir

Java yüksek performansa sahiptir. Yorumlanmış sekiz ikil kodlarının performansı yeterli değerinin üzerindedir.

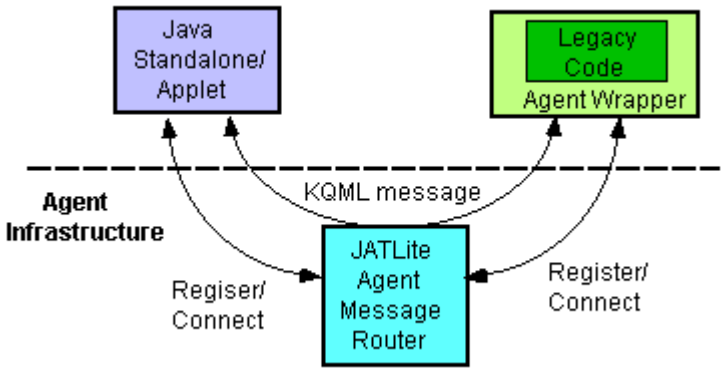
Java çoklu iş parçacıklarını destekler. Sıradan tek iş parçacığı ile çalışan C ve C++ programları tiplerindeki programlar ile aynı anda olması gereken pek çok işlemi yerine getirmek mümkün değildir. Java aynı anda birden fazla iş parçacığı ile çalışabilir. Java etkin bir monitör ve koşul değişkenleri için kullanılan senkron ilkeller kümesine sahiptir.

Java dinamiktir. Java'nın nesneye yönelik özelliği olmasından dolayı sınıflara yeni metot ve değişken eklemek çok kolaydır ve bu kendi istemcilerini etkilemez.

3. JAVA AGENT TEMPLATE,LITE (JATLite)

3.1. JATLite Nedir?

JATLite (Java Agent Template,Lite) internet üzerinde güvenli bir şekilde haberleşebilen yazılım etmenleri yazabilmek için java dilinde geliştirilmiş program paketidir[4]. JATLite aynı zamanda aşağıdaki şekilde gözüktüğü gibi etmenlerin isim ve şifre vererek kayıt yaptırabilecekleri, bağlantı kurup koparabilecekleri, mesaj gönderip alabilecekleri, FTP gibi protokoller ile dosya alışverişinde bulunabilecekleri, birbirleriyle haberleşebilecekleri bir yapı sağlar.



Şekil 3.1 JATLite Etmen Mesaj Yönlendiricisi

3.2. JATLite Ne İçin Faydalıdır?

Etmen sistemleri oluşturma ve bu sistemlerde hata ayıklama zor bir işlemdir. Etmen tabanlı yeni sistemler farklı platformlarda çalışabilsin ve appletlerin de geçici etmenler olarak çalıştırılabilmesi için Java dilinde yazılmaktadır. JATLite, bu tip sistemleri kolayca oluşturabilmek için java etmen alt yapısı ve temel etmen şablonları sunar.

Jatlite, yüksek seviyeli dil ve protokol kullanımını sağlayan etmenleri oluşturabilmek için şablonlar verir. Bu şablonlar programcıya, önceden tanımlı ve etmen oluşturmayı sağlayan bir çok java sınıflarını hazır sunar. Daha da önemlisi bu sınıflar

katmanlı bir yapıda sunulur, böylece programcılar oluşturacakları sistemler için hangi sınıfların gerektiğine kolayca karar verebilirler. Örneğin eğer programcı KQML kullanmak istemezse KQML katmanındaki sınıflarda göz ardı edilir. Bununla birlikte eğer bu katmanlar kullanılırsa buradaki sınıflarda otomatik olarak sisteme dahil edilmiş olur.

3.3. JATLite Neden Tekdir?

JATLite'ı tek yapan en önemli özellik, birlikte gelen etmen altyapısıdır. Geleneksel etmen sistemleri etmenler arasındaki bağlantılar için etmen isim sunucusu kullanırlar. Bir etmen basitçe bağlantı kuracağı etmenin ismini ve IP adresini sunucudan öğrenir ve doğrudan etmen ile bilgi alışverişi için bağlantı kurur.

Böyle bir sistemde bir etmenin IP adresi değiştiğinde ilk etmen bundan bir mesaj gönderdiğinde ve bunun başarısız olduğunda haberdar olacaktır. Eğer ikinci etmen herhangi bir nedenden dolayı çalışamaz duruma gelirse, başarısız gönderilen mesajların tekrar alıcısına ulaştırılması ve güvenli bir haberleşme ortamının sağlanmasıyla sistemdeki her etmenin kendisi meşgul olacaktır. Bu durumlarda dağıtık hesaplamaların ne kadar kolay bir şekilde başarısız kalabileceğini görebiliyoruz.

Jatlite alt yapısı ile, Etmen Mesaj Yönlendiricisine her etmen tek bir bağlantı kurar. Etmen mesaj yönlendiricisi gelen mesajları isimleri ile en son bilinen IP adreslerine yönlendirir. Aynı zamanda aynı bir elektronik mail sistemi gibi ,mesaj yönlendiricisi mesajların hepsini alıcı etmen mesajı alıp "silme" isteği yollayana kadar tutar.

Böylece etmenler mesaj yönlendiricisine kayıt olma isteklerini ve sonrada bağlantı kurma ve koparma isteklerini de mesajla bildirebilirler. Dağıtık hesaplama her zaman mesaj yönlendiricisi tarafından korunur.

3.4. Etmen Mesaj Yönlendiricisi Nedir?

JATLite etmen mesaj yönlendiricisi, kayıtlı olan bir etmenden gelen mesajı doğru alıcısına yönlendiren özel bir uygulamadır. Alınan mesaj dosya sisteminde de kuyruk yapısı ile saklanır. Böyle bir uygulamanın geleneksel etmen isim sunucu kullanmaya göre üç önemli avantajı vardır.

- Applet olarak çalışan bir etmen, popüler tarayıcılardan kaynaklanan güvenlik sınırlamalarına rağmen diğer etmenlerle haberleşebilir.

- Alıcı etmenin mesajı alamaması durumunda asenkron haberleşme mümkündür.
- Bir etmen diğer etmenlerin adresleri veya gönderilemeyen mesajların durumu ile ilgilenmek zorunda değildir.

Yönlendirici, tasarımı ile ilgili olarak 4 farklı konuda anlatılabilir:

Yoklamaya Karşı Kuyruklama: Etmenler başka etmenlere mesaj göndermek istediklerinde, gönderici etmen bağlantının varlığını kontrol ederler. Eğer bağlantı kurulmamış ise, gönderici etmen alıcı etmene bağlanmaya çalışır ve eğer başarılı olursa, mesajı gönderir. Bu mesaj yoklamadır ve alıcı için bağlantı sağlanamamış ise problemlere yol açar. Örneğin alıcı etmen çalışmıyorsa ya da alıcı applet etmeni ise gönderici doğrudan alıcıya mesaj gönderemez. Dolayısıyla göndericinin mesajları saklamak için mesaj emanetçisine(mesaj kuyruğuna) ve alıcı taraf ile bağlantı sağlandığında mesajı göndermek için de yönlendirme mekanizmasına ihtiyacı vardır. Yönlendirici mesaj emanetçisini korur ve alıcı taraf ile bağlantı kurulur kurulmaz mesajları otomatik olarak göndermek için mekanizmaya sahiptir. Yönlendiricinin başka ek avantajları da vardır:

- Yönlendirici rehber hizmeti de olabilir ve diğer etmenler(java olmayan ya da applet olmayan etmenler) ile iletişim mekanizması sağlayabilir.
- Yönlendirici mesaj kuyruğu sağlamak içindir. Dolayısıyla etmen çökse ya da aynı anda pek çok mesaj gönderilse bile mesajlar yok olmaz. Bu sağlamlığı arttırır.

Mesaj emanetçisini merkezileştirmek ile, mesaja bir şey olduğu zaman mesaj kolayca izlenebilir.

Applet Etmeni İletişimi: Eğer Netscape veya IE gibi tarayıcılardan applet çalıştırırsak, applet etmenin sadece kendisini çıkaran sunucu ile iletişim halinde olabileceği gibi bir güvenlik sınırlaması ile karşılaşırız. Bu ise pratik amaçlar için, appletin bütün mesajları HTTP sunucusunda çalışan uygulama programına geçirmesi ve uygulama programının mesajları uygulama veya applet etmeni olduğunu dikkate almadan alıcıya göndermesidir. Dolayısıyla applet etmeni yerine mesajları yönlendirecek bir uygulamaya ihtiyacımız var.

Etmen İsim Hizmeti(ANS): ANS etmen iletişimi için uygun bir hizmettir. Çünkü her bir etmenin sıkça değişen muhtemel bütün alıcıların adreslerini korumak zorunda değildir. Etmen alıcının o anki adresini öğrenmek istiyor ise ANS'ye sorabilir. ANS gerçeklemesi geliştirme yapılan her bir alt yapı için farklı olabilir. Yönlendirici

dinamik olarak değişen etmen adreslerinin tutulduğu uygun bir Agent Name Server'dir. Etmen diğer etmenlerin adresini yönlendiriciye sorabilir ama normal olarak mesaj gönderme için etmenin alıcı adresleri ile ilgilenmesine gerek yoktur.

Ölçekleme Problemi: Eğer etmenler bütün mesajlarını yönlendiriciye gönderir ise, bu yönlendiricinin performansı açısından bir ölçekleme problemine yol açabilir diye düşünülebilinir. Ancak bu problem sistemdeki etmen sayısına göre birden fazla yönlendirici kullanılarak çözülebilir.

3.4.1. Yönlendiricinin Özellikleri

Mesaj Kuyruklama: Yönlendiriciye olan bütün mesajlar dosya sisteminde kuyruklanır ve etmenin isteği doğrultusunda silinir ya da geri çıkarılır.

Mesaj Yönlendirme: Bütün mesajlar alıcı mesajı alabileceği zaman doğru alıcıya yönlendirilir.

Etmen İsim Hizmeti: Etmenin sadece mesaj gönderdiği zaman ihtiyaç duymamasına rağmen yönlendirici isim hizmeti sağlamaktadır.

Kayıt: Yönlendirici kayıt olma fonksiyonu sağlamaktadır. Etmen yönlendiriciyi kullanmak istiyor ise yönlendiriciye kayıt olmalıdır.

Güvenlik: Yönlendirici etmenin isim ve şifresini kullanarak güvenlik kontrolü sağlamaktadır.

List-Agents: Etmen eğer yönlendiriciye kayıtlı etmenlerin listesini ve bağlantı durumlarını öğrenmek istiyorsa list-Agents mesajını yönlendiriciye gönderebilir.

Disconnect: Etmen eğer tekrar bağlanana kadar mesaj almak istiyorsa 'disconnect-agent' mesajını yönlendiriciye gönderebilir. Eğer etmen yönlendiriciye disconnect mesajı göndermemiş ise, yönlendirici mesaj olduğu zaman onu etmene göndermeyi dener. Eğer etmen disconnect mesajı göndermeden kazaren bağlantısını kaybetmiş ise, yönlendiricinin tek başına etmen olup olmadığını kontrol eder. Etmen tek başına etmen ise ve yönlendirici etmen ile bağlantı kuramıyor ise, yönlendirici belli zaman aralıkları ile etmen ile olan bağlantıyı kontrol eder. Eğer bağlantının denemelerde sürekli olmadığı görülüyor ise etmenin otomatik olarak kayıdı iptal olur. Kontrol zaman aralıkları ve en fazla deneme sayısı parametrik olarak belirlenir.

Mesaj Rezervasyonu: Bazı mesajlar belli tarihlerde ve yerde alınması şeklinde rezerve edilebilir. Alıcı etmen, etmenin kendisi ya da başka etmenler olabilir.

3.4.2. Yönlendiricin Varsayımları

- TCP/IP tabanlı bir iletişim
- Her bir kayıtlı etmen için bir bağlantı(aynı anda etmen yönlendirici için birden fazla soket açamaz)
- Mesajlar ayrıştırılabilir KQML mesajı olmalı ve alıcı belli olmalı.
- Bütün alınan mesajları yönlendirici dosyaya kayıt eder. Mesajı dosya sisteminden silmek 'delete-message' mesajı yönlendiriciye gönderilir.
- Etmen yönlendirici hizmetinden yararlanmak istiyorsa yönlendiriciye kayıt olmalı. Etmen yönlendiriciye bağlanmak istediği zaman doğru protokol mesajlarını yönlendiriciye göndermeli.

3.4.3. Yönlendirici Bileşenleri

3.4.3.1. Yönlendirici Sunucusu

Yönlendirici sunucusu yukarda belirtilen bütün özellikleri yerine getiren bir java uygulama programıdır. Eğer applet etmeni kullanılmak isteniyorsa, yönlendirici istemci appletin yükleneceği HTTP sunucu makinesinde çalışmalıdır. Eğer etmenler tek başına uygulama ise, yönlendiricinin HTTP sunucusunda çalışmasına gerek yoktur. Yönlendirici sadece e-mail daemonu gibi çalışır. Yönlendirici aşağıdaki parçaları içerir.

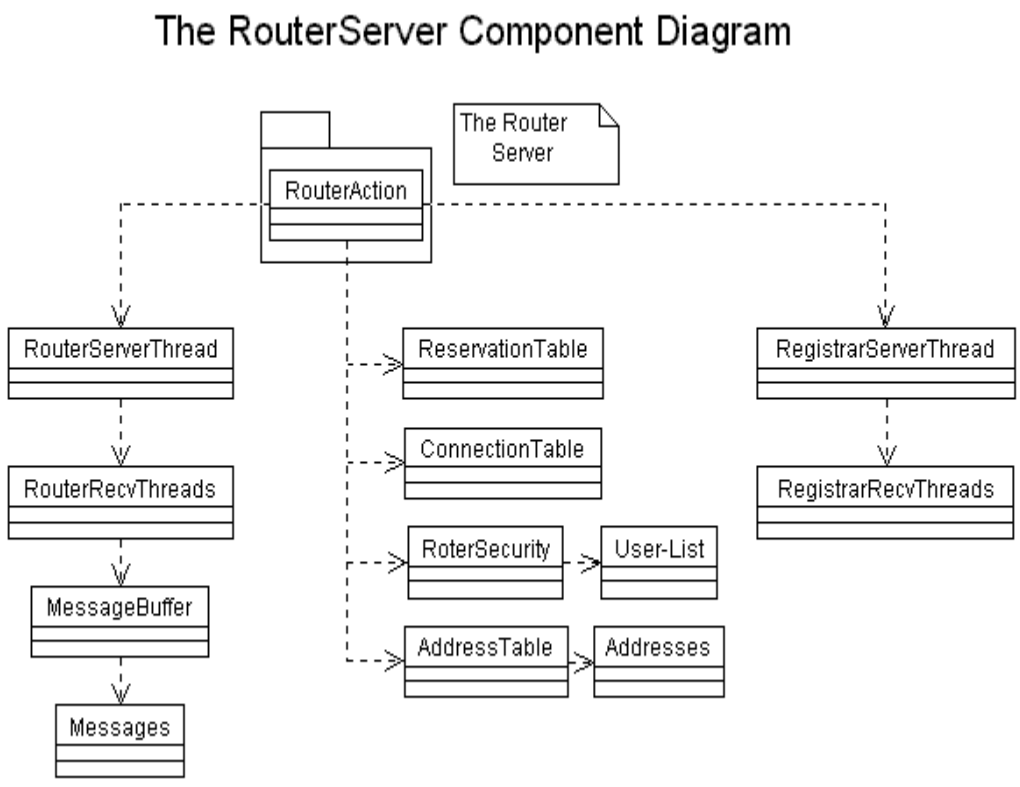
- Bir tane, bütün veri üyelerini ilkleyen ve rezerve edilmiş mesajları yöneten RouterAction
- Bir tane, etmen kayıtlarını almak için çalışan iş parçacığı RegistarAction. RegistrarAction'ın yönlendiriciye kayıt olmak isteyen etmenlerin bağlantılarını kabul eden ve RegistrarRecvThread'ı yaratan bir RegistrarServerThread'ı vardır. RegistrarRecvThread kayıt için gerekli bilgileri alır. Daha sonra kayıtlı etmen listesini günceller. Kayıt sırasında RegistrarRecvThread RegistrarSecurity kontrolü yapar güvenli bağlantı için.
- Bir tane ,bütün kayıtlı etmenlerin adreslerini içeren AddressTable.
- Bir tane, şu anda bağlantılı olan alıcı iş parçacıkları listesini içeren ConncetionTable. Alıcı iş parçacıkları etmen adı ile aynı tek isimleri vardır.
- Bir tane, rezerve edilmiş mesaj bilgilerini içeren ReservationTable.

- Bir tane, güvenli bağlantı için RouterSecurity. Etmen yönlendiriciye bağlanmak istediği zaman, RouterSecurity etmenin bağlantı protokolünü sağlayıp sağlamadığını kontrol eder. Şifre kontrolü zorunlu kılınırsa RouterSecurity şifreyi kontrol eder.

- Bir tane, kayıtlı etmenlerin bağlantı isteklerini kabul etmek ve RouterRecvThread yaratmak için RouterServerThread

- Çok miktarda, diğer etmenler ile haberleşmek isteyen etmenlere atanmış RouterRecvThread. RouterRecvThread bağlantıdan mesajları alır ve alıcı etmene yönlendirir. RouterRecvThread alınan mesajları arabelleklediği kendi mesaj kuyruğu vardır.

Yukarıdaki bileşenler Şekil 3.2'de gösterilmiştir.

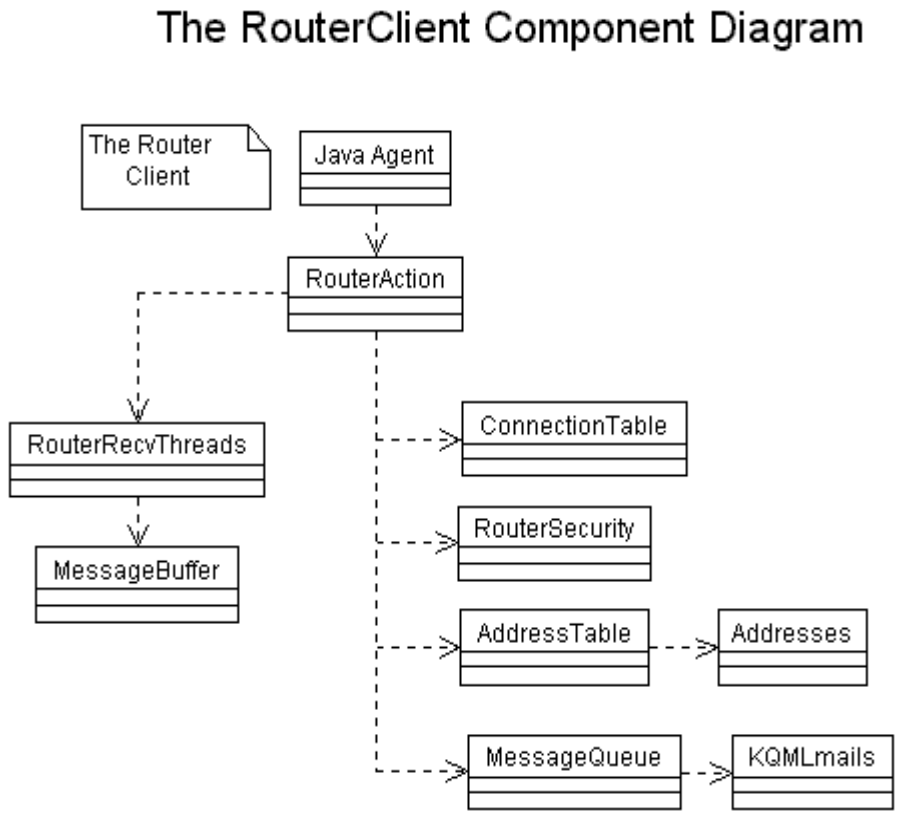


Şekil 3.2 Yönlendirici sunucusunun bileşen diyagramı

3.4.3.2. Yönlendirici İstemcisi

Yönlendirici istemcisi yukarıdaki varsayımlara uyan uygulama etmeni ya da applet etmeni olabilir. JATLite içini doldurmak için Act metoduna sahip sınıf şablonu RouterClientAction'ı kullanıcılara sunar. Eğer etmen yukarıdaki varsayımlara uyar

ise yönlendirici istemcisinin gerçeklenme detayı ile ilgili endişe edilmesine gerek yoktur. Sadece Act() metodu doldurulur. Eğer varsayımlar ihlal edilecekse amaca göre Router Client Object Relation Diagram incelenerek yeni sınıflar eklenebilir ya da varolanlar değiştirilebilir. Şekil 3.3'de en az gerekli yönlendirici istemcisi bileşenleri bulunmaktadır istenirse başka bileşen eklenebilir.

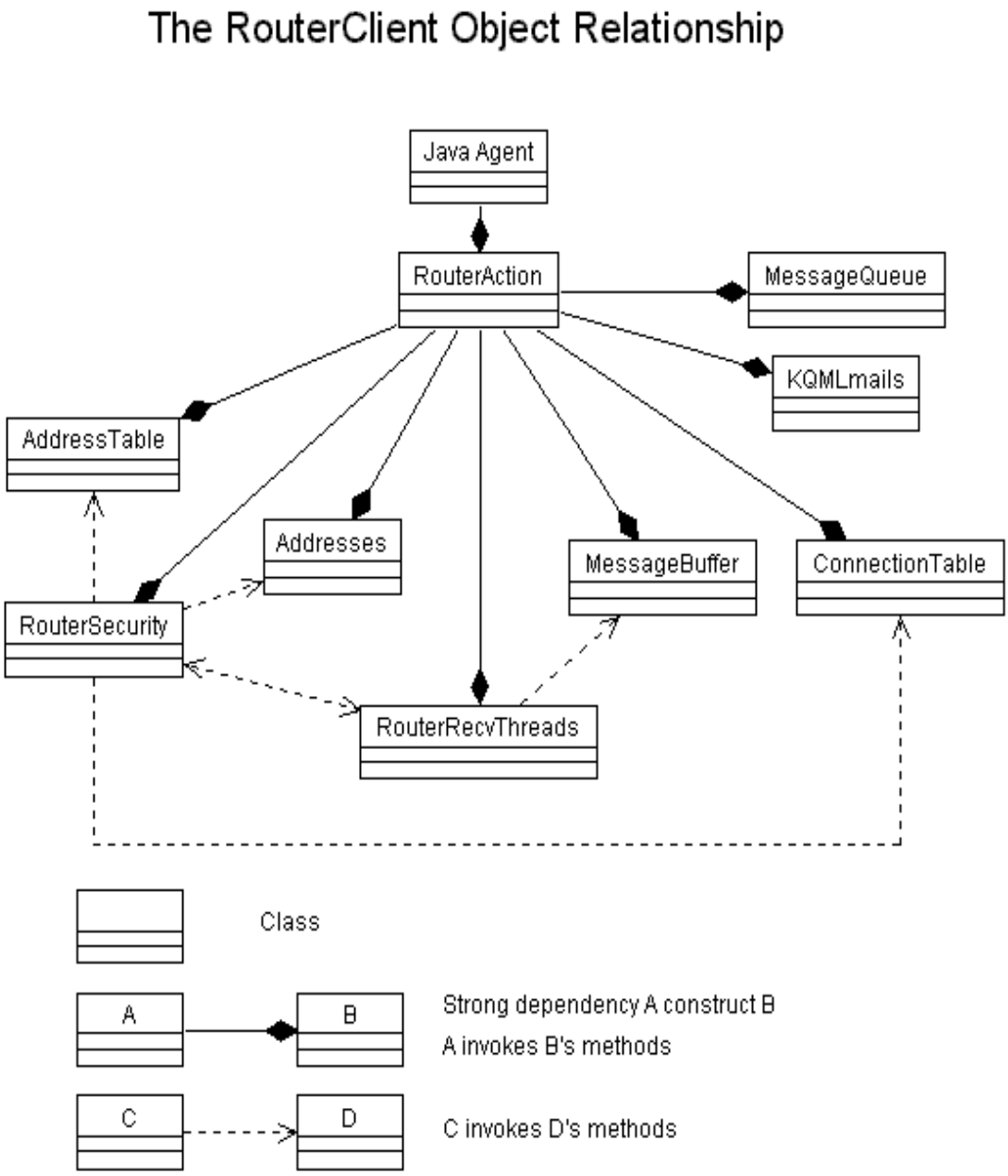


Şekil 3.3 Yönlendirici istemci bileşen diyagramı

- Bir tane, alınan mesaj ayrıştırıldıktan sonra Act() metodunu gerçekleştiren AgentAction. AgentAction bütün veri üyelerini ilkler.
- Bir tane, diğer etmenlerin adreslerini tutan AddressTable. AddressTable kendi ve yönlendirici olmak üzere en azından iki adres tutar.
- Bir tane, bağlantı bilgilerini tutan ConnectionTable. Yönlendirici istemcisi sadece yönlendirici ile haberleşme kanalına sahip olmak zorunda değildir. Amaca göre diğer etmenler ile bağlantı yaratılabilir.
- Bir tane, yönlendiriciye bağlanan ve yönlendiriciden mesaj alan ve gönderen BrecvThread.

- Bir tane, bağlantı/bağlantı kes protokol mesajlarını alan ya da gönderen RouterSecurity.
- Bir tane, yönlendiriciden alınan mesajların arabelleklendiği MessageBuffer.
- Bir tane, MessageBuffer'dan çıkarılan mesajların saklandığı mesaj kuyruk vektörü. Bir tane daha fazladan mesaj kuyruğu olmasının sebebi mesaj silmeyi uygun şekilde gerçekleştirmektir.

Şekil 3.4'de yönlendirici istemcisinin nesneleri arasındaki ilişkiler gösterilmiştir.



Şekil 3.4 Yönlendirici istemcisinin nesneleri arasındaki ilişkiler

3.4.3. Yönlendirici Nasıl Çalışır

3.4.3.1. Mesaj Yönlendirme

Yönlendirici sunucusunun çekirdek sınıfı RouterRecvThread'dir. Etmen tipine göre, yönlendirici etmenden mesaj aldığı zaman aşağıdaki farklı durumlar ortaya çıkar.

- Durum 1. Yönlendiricinin kendisi alıcı. Mesajı gerçekleştirir. Mesaj delete-message, request-address, list-users, reserve-message ve disconnect 'ten birdir.
- Durum 2 . Alıcı bağlı durumda ve kayıtlı bir tek başına bir etmen. Alınan mesaj alıcının RouterRecvThread mesaj kuyruğuna eklenir.
- Durum 3. Alıcı kayıtlı ve şuan bağlı olmayan ama etmene bağlanmak olası tek başına bir etmen. Etmene bağlanarak alıcı için RouterRecvThread yaratılır. Alınan mesaj alıcının RouterRecvThread mesaj kuyruğuna eklenir.
- Durum 4. Alıcı kayıtlı ve şuan bağlı olmayan ama etmene bağlanma mümkün olamayan tek başına bir etmen. Mesaj alıcının gelen dosyasına kayıt edilir.
- Durum 5. Alıcı kayıtlı ve bağlı bir applet etmen. Alınan mesaj alıcının RouterRecvThread mesaj kuyruğuna eklenir.
- Durum 6. Alıcı kayıtlı ama bağlı olmayan bir applet etmen. Mesaj alıcının gelen dosyasına kayıt edilir.
- Durum 7. Alıcı kayıtlı ve hattan düşmüş bir etmen. Mesaj alıcının gelen dosyasına kayıt edilir.
- Durum 8. Alıcı kayıtlı olmayan bir etmen. Göndericiye hata mesajı gönderilir.

3.4.3.2. Mesaj Rezervasyonu

Yönlendirici mesaj rezervasyon isteği aldığı zaman, RouterRecvThread istekçi etmen için uzantısı 'etmenAdı.rsv.zamandamgası' olan bir yeni dosya yaratır. Zamandamgası java SDK'da Date() veya Calendar() sınıfı kullanarak okunabilir bir zaman damgasına dönüştürülebilecek 12 sekiz ikildir sayıdır. İçeriğin ayrıştırılmasından sonra, rezerve edilmiş mesaj dosyaya saklanır ve zaman damgası ve rezerve edilen dosya adı kullanılarak Reservation nesnesi yaratılır. Reservation nesnesi RouterAction'ın ReservationTable'ına eklenir. RouterAction belli aralıklarla rezerve edilen zamanı kontrol eder ve eğer rezerve edilen zamana ulaşıldı ise RouterAction rezerve edilen mesajı alıcıya göndermeyi dener. Eğer RouterAction

bazı nedenlerden dolayı gönderme işleminde başarılı olamaz ise daha önceden bilinen miktarda göndermeyi dener. Etmen rezerve edilen mesajın alma yerini de belirleyebilir. Alıcı rezerve edilmiş mesajın mesaj göndericisi ile aynı olmak zorunda değildir. Etmen A etmen B için belli yer ve zaman için mesaj rezerve edebilir.

3.4.3.3. List-agents ve Request-Adress

RouterRecvThread ConnectionTable'ı list-agents için veya AddressTable'ı request-address için kontrol eder ve gerekli bilgileri elde eder. Daha sonra RouterRecvThread yanıt olarak istekçiye gönderir.

3.5. JATLite Etmenleri Akıllı mıdır?

JATLite, etmenlerin haberleşmesi ve etkileşimli çalışmalarını sağlamak dışında özel yetenekler sunmaz. Yani JATLite, insan davranışlarını taklit eden, bilgi arayan etmen şablonları sağlamaz. Programcı sistemi için neyin doğru olduğuna kendisi karar verebilsin diye serbest bırakılmıştır. Jatlite bu amaçla özel bir araç sunmaz ama kullanımını destekler.

Jatlite'ın paket alt yapısı etmenlerin bir makineden diğer makineye gitmelerine yani hareketli olmalarına, internet üzerinden otomatik mesaj biriktirme ve kuyruklama yöntemleri ile sisteme girmeye sistemden çıkmaya olanak sağlar. Bu özellikler, etmenlerin bozulabileceği, konumlarını değiştirebileceği sistemlerde gereklidir ve Mesaj Yönlendiricisi alt yapısı ile sağlanmaktadır.

Mesaj Yönlendiricisi aynı zamanda hareketli etmenler tarafından da kullanılabilir. Jatlite özerk etmen taşınması için bir metot sunmasa da yer değiştiren etmenler arasında mesaj geçirimi için Mesaj Yönlendiricisi çok uygundur. Kullanıcının etmeni bir yerden bir yere taşınması için programlamış olup olmamasıyla ilgilenmez ancak etmenin gelecekteki bütün mesajlarını toplamak için uğraşır.

Güzel bir özellik applet olarak çalışan etmenler üzerindeki bazı tarayıcıların sınırlamalarının mesaj yönlendiricisi ile ortadan kalkmasıdır, böylece applet etmenler her tür tarayıcı ile çalışır duruma gelmektedir.

3.6. JATLite Hangi Standartları Kullanır?

JATLite etmen haberleşmesi için KQML dilini kullanır. KQML daha öncede söylendiği gibi hem mesaj formatını hem de mesaj iletim protokolünü kapsayan bir

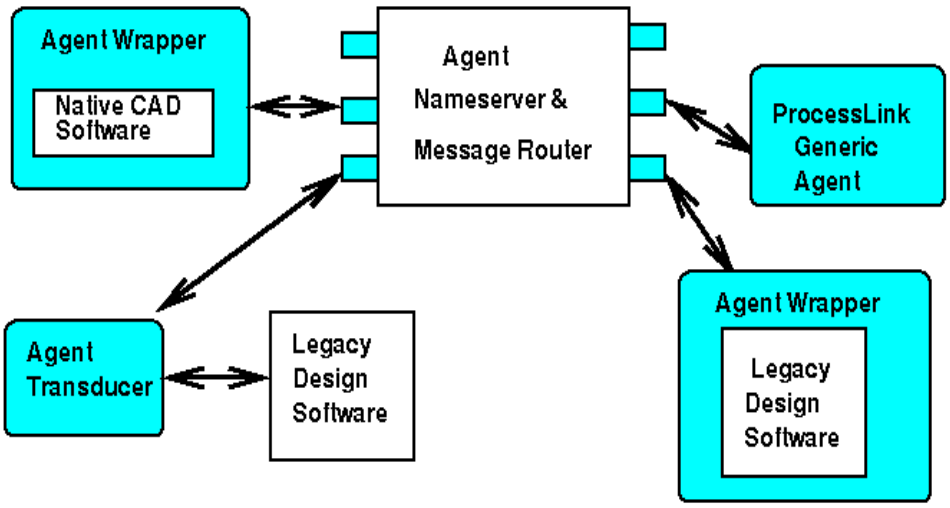
kavramdır. Haberleşme daha alt seviyede TCP/IP,SMTP,FTP gibi internet standartları üzerine oturtulmuştur. Ancak programcı diğer etmen dillerini kullanan etmenler oluşturabilirler, bunun için JATLite'ın katmanlı mimarisinden faydalanır.

JATLite, hiçbir özerk etmen teorisi üzerine yönelmez. Çünkü etmenlerin iç mimarileri hakkında henüz bir ortak anlayış yoktur.

JATLite , Sun'ın JDK 1.1 'i ile kullanılmak zorundadır. Jatlite etmenleri çalıştıran tarayıcılar JDK 1.1 plug'inini kullanmalıdır.

3.7. Neden Etmen Mesajları Kullanılır?

JATLite'ın kullanılan en önemli özelliği, yaşayan uygulamalara, diğer programlarla haberleşme, mesaj gönderip alma için gerekli arayüzü sunmasıdır. Aşağıdaki şekil bu işleyişi göstermektedir.



Şekil 3.5 Jatlite'ın varolan uygulamalar için sağladığı arayüz

Jatlite, etmen haberleşmesi için standart yazılım sağlar. KQML ve diğer etmen haberleşme protokolleri mesaj alışverişi için standartlar sağlarlar.

Anahtar fikir; en olağan şekilde yazılım parçalarını haberleştirebilmektir. Nesne tabanlı bir dil oluşturma zorunluluğu da yoktur. ASCII karakter katarları ile bu sağlanabilmektedir. CORBA gibi nesne platformları gerekli değildir.

Diğer yazılımları JATLite'a entegre etmek çok kolaydır. JATLite etmenleri C,C++,LISP ' ede kolayca entegre edilebilir.

3.8. JATLite'ın Yetenekleri

- Paketin diğer parçalarını etkilemeden diğer teknolojilerle yer değiştirebilen katmanlı bir yapıya sahiptir.
- TCP/IP üzerine kurulu düşük düzeyli haberleşme Unix, Windows, MAC OS gibi bilinen işletim sistemleri tarafından desteklenir, diğer protokollerde kolayca eklenebilir.
- Etmen mesajları KQML dili ve protokolü üzerine kurulmuştur. Mesajların dış katmanları için doğrudan sentaks kontrolü vardır. Mesajların iç yapıları diğer herhangi bir dilde olabilir (SQL, Express, KIF vb.)
- Çoklu iş parçacığı işlemleri ve çoklu sunucu ve mesaj alma soketleri. Soket bağlantıları kalıcıdır ve zaman aşımına sahiptir.
- Etmenlerin kaydolması, bağlanması, bağlantıyı kesmeleri isim ve şifre servisleri için Etmen Mesaj Yönlendiricisi sunar.
- Mesaj Yönlendiricisi hareketli etmenler için mesaj biriktirme ve kuyruklama yeteneğine sahiptir.
- Java ve C++ ile yazılmış etmenleri ve tarayıcılar tarafından desteklenen applet etmenlerini destekler.
- FTP ile dosya transferi olanağı.

3.9. JATLite Varsayımları

JATLite Sun'ın JDK1.1'ini destekleyen herhangi bir platformda çalışabilir. Windows95, WindowsNT, Solaris, MAC OS gibi işletim sistemleri buna dahildir. Diğer java ortamları için değişiklik yapmak gerekebilir. Applet etmenler, Netscape, Internet Explorer gibi tarayıcılar ya da Sun'ın applet görüntüleyicisi ile çalışabilir. JATLite etmenleri standart TCP/IP haberleşmesi ve prosesler arası haberleşme için de soketler üzerine kurulmuştur. Bütün mesajların, internete bağlı bilgisayarlarda java uygulaması olarak çalışan Etmen Mesaj Yönlendiricileri aracılığı ile gönderildiği kabul edilir .

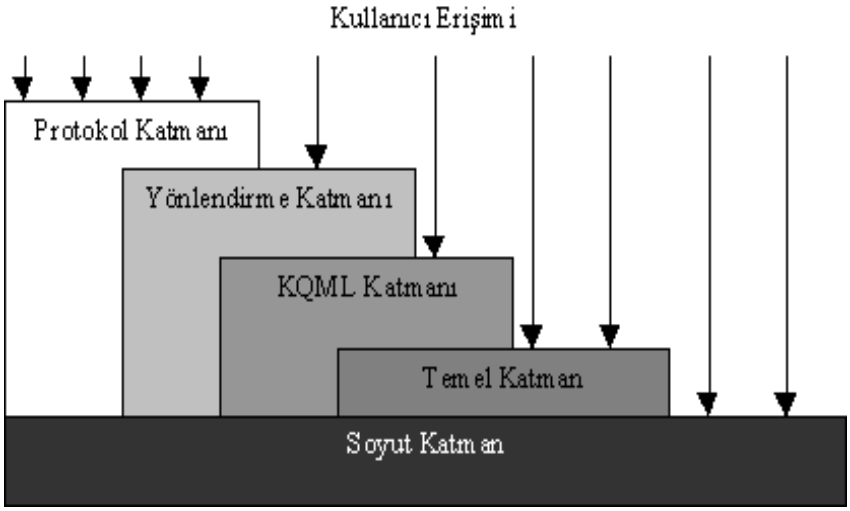
Bir uygulamanın JATLite uyumlu olabilmesi için KQML standardına göre ASCII mesajları gönderebiliyor olması ve mesaj yönlendiricisi bağlantı protokolü ile yönlendirici ile bağlantı kurabiliyor olması gerekmektedir.

Genel çalışma varsayımları;

- TCP/IP tabanlı haberleşme.
- Mesaj alışverişi ile etmen haberleşmesi.
- Bağlantılı her etmen için bir bağlantı.
- Her etmene tek bir adres(isim,makine,port,mesaj-metodu,açıklama). Eğer etmen appletlerdeki gibi sunucu soketine sahip değilse host'a bir değer atanmaz ve port numarası da ,"-1" e eşitlenir.
- Mesaj sonu karakteri "\004" 'e set edilmeli.

3.10. JATLite Katmanları

JATLite mimarisi aşağıdaki şekildeki gibi her biri özel olarak tasarlanmış katmanlı bir yapıya sahiptir. Böylece programcı, kuracağı yapının ihtiyaçlarına göre istediği katmandan başlayarak programını geliştirebilecektir. Mesela TCP/IP üzerine haberleşme isteyen ancak KQML'yi kullanmak istemeyen bir programcı aşağıda açıklandığı gibi sadece Soyut ve Temel katmanları kullanacaktır.



Şekil 3.6 JATLite her biri büyük ölçüde özelleştirilmiş katmanlı bir yapıdan oluşur.

Soyut Katman, JATLite'ın kullanımı için gerekli olan soyut sınıfları sağlar. JATLite bütün haberleşmelerin TCP/IP olduğunu farzetmesine rağmen, programcı istersen bu katmanı değiştirerek başka protokol mesela UDP kullanımını da sağlayabilir.

Temel Katman, soyut katman ve TCP/IP'ye dayanan temel haberleşme rutinlerini sağlar. Temel katman mesela soketten okumayı sağlayacak ve çıkışları bir dosyaya yazacak şekilde geliştirilebilir. Çoklu mesaj portlu etmenleri destekleyecek şekilde de geliştirilebilir.

KQML Katmanı, KQML mesajlarının ayrıştırılması ve saklanmasını sağlar. Kayıt olma, bağlanma, bağlantıyı kesme gibi işlemlerde "Center for Design Research"'[5] un önerdiği KQML standardı biraz daha geliştirilerek standartlaştırılmıştır.

Yönlendirme Katmanı, etmenler için isim kaydetme, mesaj yönlendirmesi ve kuyruk mekanizması olanağı sağlar. Bütün etmenler diğer herhangi bir etmene etmen mesaj yönlendiricisi aracılığı ile mesajlarını yollarlar. Bir etmenin çalışması sonaerer ya da bozulursa yönlendirici mesajları etmen geri gelene kadar saklar. Java ve World Wide Web güvenlik kısıtlamaları düşünülünce özellikle appletler için yönlendirici oldukça önem kazanmaktadır.

Protokol katmanı, SMTP, FTP, POP3, HTTP gibi internet standartlarını, hem java uygulamaları hem de appletleri için sağlar. Projede kullanılan versiyon SMTP ve FTP'yi destekliyor ancak kısa zamanda diğer protokoller de ele alınıp geliştirilebilir. Eğer bir etmen farklı bir koddaki veriyi veya uzun bir veriyi ya da KQML mesajlarını e-mail ile göndermeyi istiyorsa protokol katmanı oldukça yardımcı olacaktır.

3.11. JATLite'ın Geleceği

Gelecekte JATLite'a e-mail haberleşmesi için bir katman daha eklenmesi düşünülmektedir. Böylece JATLite etmenleri diğer etmenlerle e-mail aracılığı ile haberleşebilecektir. Bu türden bir haberleşme firewall içeren yapılar için olumlu olacaktır.

JATLite Etmen Mesaj Yönlendiricisine DNS'in çalışma mantığı gibi bir yapı düşünülmektedir. Böylece JATLite etmenleri daha ölçeklenebilir olacaktır.

Mesaj Yönlendiricisi test edilmesine rağmen şuanki yapıda sistem mesaj yönlendiricisinin bozulması ile çalışamaz hale gelmektedir. Mesaj yönlendiricisini kontrol eden ve bozulduğu anda yenisini yaratabilen yapılar üzerinde çalışmalar sürmektedir.

KQML dili desteklenmeye devam edilecektir ancak bir çoğu KQML katmanını kullanmayarak kendi sistemleri için diğer dilleri kullanma yoluna gidebilecektir, veritabanı işlemleri için SQL olabileceği gibi.

3.12. Neden CORBA veya Java RMI Değil?

CORBA ve RMI JATLite ile uyumludur ve bir çok açıdan beraberce kullanılabilir. Eğer etmeninizin dili java arayüzüne sahip değilse, örneğin, CORBA'yı çevirici olarak kullanabilirsiniz.

JATLite'ın yerini tamamen alabilmek açısından iki dezavantaj vardır. Bütün yazılım parçaları CORBA nesneleri yollamalıdır ve diğeri, Mesaj Yönlendiricisi yeniden yazılmalıdır.

3.13. Katman Seçimi

Dil ve protokoller üzerindeki kısıtlamalar karşın JATLite beş farklı katman sunar.

- Eğer KQML kullanacaksanız, temel haberleşme metotları üzerinde düşünmenize gerek kalmayacak ve doğrudan KQML katmanından başlayabilirsiniz. Alınan mesaj KQML sentaksına göre kontrol edilecek ve KQMLmesaj objesi içine saklanacaktır

- Eğer Etmen İsim Sunucusu ve Mesaj Yönlendiricisi kullanacaksanız, O zaman Yönlendirme katmanından başlayabilirsiniz. Bu katman mesaj yönlendirme, alma, saklama için güzel metotlar sağlar.

- Eğer etmenlerinizin veri alışverişinde bulunmalarını istiyorsanız protokol katmanından başlayabilirsiniz. Bu katman FTP ile veri alışverişi için metotlar sağlar.

3.14. Şablonlar

JATLite her katman için şablonlar sunar.

BaseLayer: BagentAction , Abstract.AgentAction'nın alt sınıfı.

KQMLLayer: KQMLAgentAction, BaseLayer.BagentAction'ının alt sınıfı.

RuterLayer:RouterClientAction, KQMLLayer.KQMLAgentAction'ının alt sınıfı.

ProtocolLayer: IPRouterClientAction, RouterLayer.AgentClient.RouterClientAction'ının alt sınıfı

Act(Object) ve ProcessMessage(String, Object) metotlarından faydalanmak için şablonları kullanmak ve yeni veri üyeleri ve metotları eklemek isteyebilirsiniz.

3.15. Şablonlar Nasıl Kullanılır?

Şablonları tanımlamak için Yönlendirme Katmanı ve RouterClientAction üzerinden açıklama yapmak daha kolay olacaktır.

Aşağıdaki basit adımlarla, etmeninizin applet ya da uygulama olmasından ve diğer etmenlerin çalışıyor ya da çalışmıyor olmasından bağımsız olarak, etmeninizin JATLite'ın sağladığı çeşitli servisler (mesaj yollama, kuyruklama, isim sunuculuğu, KQMLmessage nesnesine otomatik mesaj dönüştürme, KQML parse işlemi, etmen listesi, vb.) sayesinde diğer etmenler ile bağlantılar kurabilir.

RouterClientAction şablonunu kullanmak için aşağıdaki adımlar kullanılmalıdır

A- AgentAction metodunuz için kurucu fonksiyonları hazırlanır (mesela MyRouterClientAction gibi);

Genelde super class(RouterClientAction)'ını kullanmak yeterli olur. RouterClientAction() kurucusu my adresss, Router Adress ve registrar adresss dışındaki tüm veri üyelerine ilk değerleri atayacaktır. Bu yüzden eğer bu kurucu fonksiyon kullanılacaksa, RouterClientAction nesnesine adresleri vermek için setMyAddress(Address), setRouterAddress(Address), setRegistrarAddress(Address) metotlarını kullanmak gerekir.

RouterClientAction(Address routerAddress, Address registrarAddress, Address myAddress, int maxidleTime) kurucusunu, adresleri set etmek için kullanılabilir. Eğer bağlantı maksimum zaman aşımı süresini kullanılacaksa, maxidleTime değerine -1 verilmelidir ya da eğer değer verilmeyecekse bu değere dakika cinsinden atamalar yapılmalıdır.

B- Act(Object) ve processMessage(String,Object) metotlarını doldurulur;

Eğer diğer etmenlerden bir mesaj alınırsa Act(Object) metodu otomatik olarak canlanır. Bu yüzen mesaj alındığı farzedilip buna göre etmenlerin davranışları burada belirlenir. processMessage(String,Object) özel bir metot değildir ancak grafik arayüzler ve iş parçacıklarıyla etkileşimleri tanımlamak için kullanılabilir. ProcessMessage metodu kullanılmayacaksa boş bırakılmalıdır

Public void processMessage(String cmd, Object obj) {}

C- RouterClientAction nesnesi yaratılır ve gerekliyse kayıt (register) ettirilir sonra da yönlendiriciye bağlanılır;

Şimdi RouterClientAction nesnesi hazır durumdadır. Bir tane yaratıp önce register() metodu ile yönlendiriciye kayıt olunmalı sonrada bağlanmak için connect() metodu çağırılmalıdır.

```
MyRouterClientAction action =
new MyRouterClientAction(routerAddress, registrarAddress ,myAddress,
100);
// Yönlendiriciye bir kez kayıt olmanız gerekiyor.

action.register();
action.connect();
```

Adres bilgileri program içinden ya da kullanıcı giriş ekranından alıp parametre olarak aktarılabilir.

3.16. AgentClientClass Sınıfının Kullanımı

AgentClientClass sınıfının kullanımı iki integer sayı üzerinde işlem (+,-,*,/) yapıp sonucu geri yollayan örnek bir etmen üzerinde anlatılacaktır.

A- AgentAction class :

```
public class CalcServer extends RouterClientAction {
......
}
```

B- Router adresi, registrar adresi ve etmen adresi için bir kurucu yazılmalıdır. Maksimum zaman aşımı için de parametre geçirilebilir.

```
public CalcServer(Address myaddress,
Address routeraddress,
Address registraraddress,
int durationtime) {
//  RouterLayer.AgentClient.RouterClientAction
kurucusunu kullan
super(myaddress,routeraddress,registraraddress,durat
iontime);
}
```

C- Act(Object obj) metodu doldurulur : Act() metodu yönlendiriciden her mesaj geldiğinde canlanacaktır. Sadece diğer etmenlerden mesaj alındığı farzedilip etmenin ne yapması gerekiyorsa o belirlenmelidir.

```
public boolean Act(Object o) {
String stampmsg = (String)o;
try {
//KQMLmail  KQMLmessage'ını kapsar
KQMLmail mail = new KQMLmail(stampmsg,0);

//  mail'i  _mailQueue ya ekle.
// Burası mesaj alındıkan sonra silinmesiyle
ilgilidir

_mailQueue.addElement(mail);

// KQMLmessage objesi için KQMLmail getKQMLmessage()
metodunu kullan.
KQMLmessage kqml = mail.getKQMLmessage();

// performative değerini bul.
String perf = kqml.getValue("performative");

// mesaj içeriğini al.

String content = kqml.getValue("content");

String receiver = kqml.getValue("sender");

//hata kontrolü

if(content == null) {
sendErrorMessage(kqml);
return false;

}

if(perf.equals("ask-one")) {
.....

}
else {
// hata mesajı
```

```
sendErrorMessage(kqml);
return false;
}

addToDeleteBuffer(0);
} catch (Exception e) {
return false;
}

return true;
}
```

D- processMessage(String,Object) metodu doldurulmalıdır : Bu metot genelde GUI ya da diğer iş parçacıklarında kullanılır. Bu örnek için uygulanmamıştır.

AgentAction 'ı yaratılır, kayıt olunur, bağlanılır.

```
public static void main(String args[]) {
// RCAddressHelper kullanılacaksa
if(args.length == 1) {
try {
....
// dosyadan adres bilgilerini oku.
...
CalcServer server = new
CalcServer(myAddress,routerAddress,
registrarAddress,idletime);

// Uygulama olduğu için ,sunucu iş parçacığını
yaratın
createServerThread(myaddress.getID(),Thread.NORM_PRI
ORITY);

if(registerRequest == true)
server.register(registraraddress,myaddress);
// etmen adresi ile yönlendiriciye bağlanın

server.connect(myaddress);

// mesaj almak için beklemeye geç
server.start();
```

```
} catch (Exception e) {
System.out.println(e.toString());
System.exit(0);
}
}
}
```

E- Uygulama dosyası CalcServer.java olarak saklanılır ve derleyip çalıştırılır. RCApplet (ya da IPRcApplet)'i kullanarak CalcServer'a mesaj yollanır.

CalcAddressFile dosyasını değiştirilir;

(RouterLayer/Resource/CalcAddressFile):
CalcServer port numarası değiştirilmeli (Yönlendirici ve CalcServer aynı makinede çalışıyorlarsa port numaraları farklı olmalı)

CalcServer 'ı çalıştırılır;
java RouterLayer.AgentClient.Example.CalcServer.CalcServer
RouterLayer/Resource/CalcAddressFile
Ya da java RouterLayer.AgentClient.Example.CalcServer.CalcServer

F- Mesaj yollanır

Performative'i 'ask-one' ve receiver'ı 'CalcServer' yapıp ve aşağıdaki mesajlar yollanılır.
+ 1 4
- 4 235
* 134 346
/ 345 5

3.17. Yüksek Seviyeli public RouterClientAction metodları

- createServerThread(String id, int priority) - Eğer etmeniniz bir java uygulaması olarak çalışacak ise diğer etmenlerin bağlanması için bu metodu kullanmalısınız
- setRouterAddress(Address) - Yönlendiricinin adresinin set edilmesi.
- setRegistrarAddress(Address) - Registrar'ın adresinin set edilmesi.
- setMyAddress(Address) - Etmen adresinin set edilmesi.

- register(Address registrarAddress,Address myAddress) throws ConnectionException - Yönlendiriciye kayıt olma, adresler gönderilmeli.
- register() throws ConnectionException - Yönlendiriciye kayıt olma. Etmen adresi ve regirtrar adresi önceden set edilmeli.
- connect() throws ConnectionException - Yönlendiriciye bağlanma. Etmen adresi ve yönlendirici adresi önceden set edilmeli.
- connect(Address myAddress) throws ConnectionException - Yönlendiriciye bağlanma. Yönlendirici adresi önceden set edilmeli.
- connect(String myName) throws ConnectionException - Yönlendiriciye kaydolma. Etmen adresi ve yönlendirici adresi önceden set edilmeli.
- sendMessage(String receiver,String msg) throws ConnectionException - Diğer etmenlere mesaj yollama.
- sendMessage(String msg) throws ConnectionException - Yönlendiriciye mesaj yollama.
- sendKQMLMessage(String msg) throws ConnectionException, ParseException - Yönlendiriciye KQML sentaks kontrolü yaparak mesaj yolla.
- addToDeleteBuffer(int) - Mesajı sil..'delete-message' yönlendiriciye nihayi durumda yollanmalıdır.
- disconnect() - Yönlendirici ile bağlantıyı kes.
- unregister() - Yönlendiriciden kaydını sil.
- endAction() - Etmeni temizle ve durdur.

3.18. Protokol Katmanının Kullanımı

Eğer SMTP/FTP gibi protokol katmanı servislerinden faydalanılmak isteniyorsa; ProtocolLayer.IPRouterAction sınıfı kullanılmalıdır.

IPRouterClientAction sınıfının kullanımı RouerClientAction sınıfının kullanımı ile neredeyse aynıdır. IPRouterClientAction processmessage metodunu SMTP/FTP kullanımını sağlamak için kullanır. Bunu aşağıdaki kodda belirtildiği gibi kullanmalısınız. Bununla beraber sendEmailMessage(), FTP()'yi uygulamalar için

CreateEmailConnection() ve CreateDataTransferConnection metotlarını da appletler için kullanabilirsiniz.

```
public void processMessage(String cmd,Object obj) {
super.processMessage(cmd,obj);
///// Buraya uygulamanızı yazın
.....
}
```

'cmd' parametresi "SMTP" ve "FTP" ile başlamamalıdır.

Yalnız başına çalışan java uygulamaları ve appletler farklı güvenlik kısıtlamalarına sahip oldukları için SMTP ve FTP için durum aşağıdaki gibidir.

3.19. Java Uygulamaları

- Java uygulamaları sunucu soketi yaratmada ve diğer makinelere bağlanmada serbest olduğu için SMTP ve FTP kullanan uygulamalardaki gibi doğrudan kullanılır.

- SMTP: SMTP kullanabilmek için kurucu fonksiyona parametre olarak geçen IPRouterClientAction nesnesi içinde etmeninizin e-mail adresini belirtmeniz gerekir. KQML mesajlarını e-mail olarak yollamak için sendEmailMessage(Address sender, Address receiver, String msg) ya da sendEmailMessage(String kqmlstr) metodları kullanılabilir. Eğer etmeninizin e-mail adresi "Address sender" parametresiyle belirtiliyorsa ve etmeniniz alıcının e-mail adresini biliyorsa, ilk metot, alıcının adresini bilmiyorsa ikinci metot kullanılabilir. İkinci metot alıcının e-mail adresini yönlendiriciden öğrenecektir. Eğer yönlendiriciye daha önceden e-mail adresi bildirilmemişse ConnectionException hatası oluşacaktır.

- FTP: FTP'yi kullanmanın en kolay yolu, FTPCon nesnesini kullanmak olacaktır. FTPCon nesnesini tanımlayın ve FTP(FTPCon) metodu çağırılır. Bu metot hatalar için ConnectionException hatası oluşturacaktır.

3.20. Appletler

- Appletler soket bağlantılar için bir takım kısıtlamalara sahip oldukları için, SMTP ve FTP sunucuları yönlendiricinin çalıştığı HTTP sunucudan farklı bir yerde ise bu sunuculara doğrudan bağlantı kurulamaz. Appletler için yönlendirici

SMTP ve FTP sunuculara erişimde proxy sunucu görevi görür. Ancak KQML mesajlarında olduğu gibi yönlendirici hiçbir mesajı saklamaz.

- SMTP: SMTP'yi kullanmak için kurucu fonksiyona parametre olarak geçirilen IPRouterClientAction nesnesi içinde etmeninizin e-mail adresini belirtmeniz gerekir. CreateEmailConnection(String kqmlmsg) ya da createEmailConnection(KQMLmessage kqml) metotlarını KQML mesajlarını e-mail olarak yollamak için kullanılabilir. Yönlendirici appletin bağlanacağı yerde sunucu soketi oluşturur ve applete 'port-info' mesajı yollar. Hata durumlarında 'port-info' mesajı yerine hata mesajı yollar. 'port-info' mesajı appletin bağlanacağı sunucu soket numarasını da içerir. Applet sunucu sokete bağlantı kurduktan sonra,IPRouterClientAction otomatik olarak SMTP veri akışını başlatacaktır (sendEmailToRouter(KQMLmessage) metodu kullanılarak) böylece doğrudan sendEmailToRouter(KQMLmessage) metodunu çağırmak zorunda kalmayız. Hata durumlarında bu metot connectionExceptin hatası oluşturacaktır.

- FTP:Bir applet çalıştığı yerel kaynak üzerindeki dosya sistemine erişemez ancak JATLite'ın sağladığı FTP fonksiyonu sayesinde uzak sistemlere dosya transfer edebilir. CreateDataTransferConnection(FTPCon con) ya da CreateDataTransferConnection(String conid, String remotePath, String host, int port, String userName, String password, String localPath, String[] files, int mode) metodlarını çağırabilirsiniz. SMTP de olduğu gibi IPRouerClientAction yönlendiriciden sunucu soketi isteğinde bulunacak ve yönlendirici de 'port-info' mesajı yollayacaktır. Sonra IPRouerClientAction sunucu soketine bağlandıktan sonra transfere başlayacaktır. 'port-info' mesajı, processRouterMessage(KQMLmessage) tarafından işlenecektir.

4. DAĞITIK ÇOKLU ETMEN TOPLANTI PLANLAMA SİSTEMİ

Kurumlarda, işyerlerinde toplantı planlama işlemi uzun süren ve zaman tüketen bir işlemdir. Çoğu zaman toplantı planlama işlemi toplantıyı düzenleyen tarafından başlatılan bir telefon görüşmeleri veya e-mail zincirleri sonrasında sonlandırılabilinir. Toplantıyı düzenleyen kişi öncelikle kendisi için bir boş zaman daha sonra ise bütün katılımcılar için ortak boş bir zaman aralığı bulmaya çalışır. Ayrıca bulunan ortak zamanda toplantı için uygun olan ve boş bir toplantı yeri de bulması gerekir. Düzenleyici toplantı zamanı için telefon ya da e-mail ile toplantı zamanı konusunda bütün katılımcıların görüşlerini alır. Katılımcılar kendi takvimlerini kontrol ederek yanıt verirler. Bütün bu yanıtların ardından toplantı düzenleyicisi toplantı zamanı için uygun ortak bir zaman aralığı var mı diye kontrol eder. Ve eğer toplantı için uygun ortak bir zaman aralığı bulmuş ise bu zaman aralığında boş bir toplantı yeri var mı diye kontrol eder. Bütün bu işlemler toplantı planlama işlemi başarılı bir şekilde sonuçlanana kadar ya da istenen zaman aralığında başarılı bir sonuç elde edemeyinceye kadar kendini tekrar eder şekilde devam eder. Bu işlemin süresi katılımcıların sayısı ve bunların düzenleyicinin telefonlarına ve e-maillerine yanıt verme sürelerine bağlıdır.

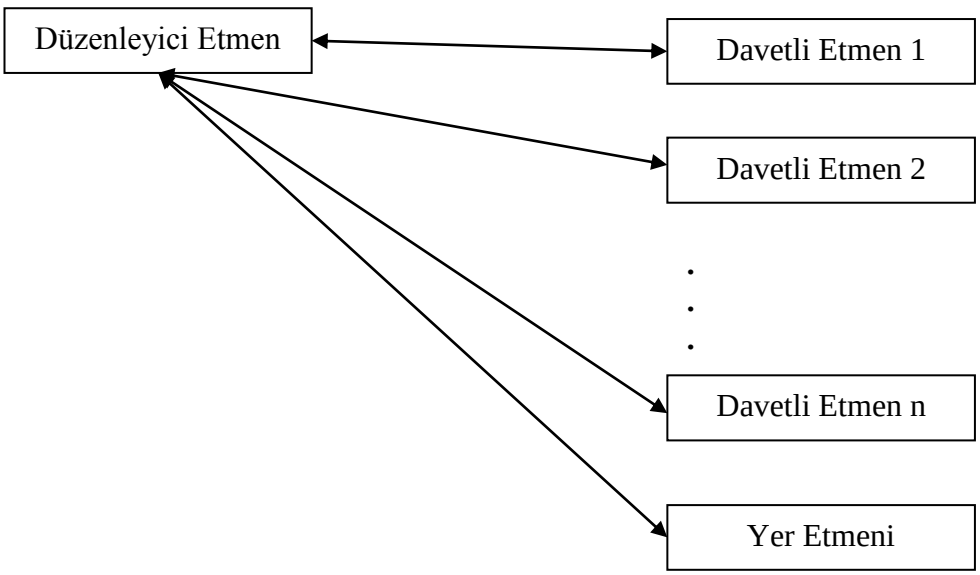
Genelde toplantı planlama aşamasında bazı kişilerin toplantıya kesinlikle katılması istenirken bazı kişiler ise toplantılara bilgi amaçlı ya da sadece orada bulunmaları için çağrılmaktadır. Yani sıradan katılımcı olarak adlandırdığımız bu kişilerin belli sayıdakilerin toplantıya katılamıyor olmaları toplantı planlamasını engellemez. Ancak önemli olarak adlandırdığımız kişilerin toplantıya kesinlikle katılıyor olmaları gerekmektedir.

Problemin karakteristiği, özerk, dağıtık ve bireye bağımlı olması, bizi çözüm için yeni teknolojik olanakları kullanmaya teşvik etmektedir. Temsil ettiği birey adına özerk bir şekilde hareket eden akıllı etmenler böyle bir toplantı düzenleme sisteminde katılımcı veya düzenleyici olabilirler. Bir çok özerk ve akıllı etmenden oluşan çoklu-etmen sistemi ise gerçek yaşamdaki toplantı düzenleme sisteminin bir simülasyonudur. Sistemdeki etmenler toplantı düzenlemek için gerekli hareketlerden sorumludurlar. Sistemdeki her bir etmen temsil ettiği kullanıcının takvim ve öncelik

bilgilerine erişebilir, temsil ettiği birey adına hareket eder, onun adına diğer etmenler ile haberleşir ve onlardan gelen istekleri yanıtlar.

Dağıtık Çoklu Etmen Toplantı Planlama sistemi etmenlerin temsil ettiği bireyin isteği ile ve onun adına diğer etmenler ile birlikte toplantı planlamasını yürütmesi işleminin yapıldığı bir sistemdir. Toplantı planlaması sistemde bulunan ve toplantıya davetli bütün etmenlerin ortak çabası ile sonuçlandırılır. Sistemde ayrıca bir de gerçek yaşama uygun olarak toplantı yerleri bilgilerinin tutulduğu bir yer etmeni de bulunmaktadır. Sisteme giren her etmen öncelikle sisteme kayıt olmak zorundadırlar. Bu ise sistemde mesajlaşmayı sağlayan ve etmen yaratmak için şablonlar sunan JATLite Router'ına kayıt olmaktır. Sistemde etmenler arası mesajlaşmalar daha önce belirtimiz bir bilgi alışveriş dili olan KQML ile gerçekleştirilmiştir.

Sistemde iki tür etmen bulunmaktadır. Bunlar kullanıcısı adına toplantı düzenleme ve toplantılara katılma işlemlerini yürüten planlama etmeni ve sistemdeki toplantı yerleri hakkında bilgilerin tutulduğu ve toplantıların bir katılımcısı gibi davranan yer etmeni. Aşağıdaki Şekil 4.1'de toplantı planlama işleminin kısa bir işleyiş şekli verilmiştir.



Şekil 4.1 Toplantı planlama sisteminin kısa işleyişi

Şekilde görüldüğü gibi planlayıcı etmenin iki rolü bulunmaktadır toplantı düzenlerken iken düzenleyici rolü, başka bir etmen toplantı düzenlerken ise davetli rolüdür. Toplantı düzenleme işlemi düzenleyici etmenin kontrolünde bir mesajlaşma

işlemeleri ile yürümektedir. Yer etmeni de sistemde bütün toplantılara katılan bir davetli gibi yer almaktadır.

4.1. Planlama Etmeni

Planlama etmeni kullanıcı adına hareket eden ve onun bütün bilgilerine sahip olan sistemin temel etmen tipidir. Sistemde yer alan her birey için bir adet planlama etmeni çalışmak zorundadır. Planlama etmeni sistemde iki role sahiptir: düzenleyici ve davetli rolleri. Eğer planlama etmeninden bir toplantı düzenlemesi isteniyor ise planlama etmeni düzenleyici rolü ile hareket etmeye başlar. Bu durumda toplantı planlama işleminin yürütücüsü durumundadır, planlamanın sorumluğu ondadır. Eğer planlama etmenin temsil ettiği bireye bir toplantıya katılma daveti geliyor ise bu durumda planlama etmeni davetli rolü ile hareket eder. Bu durumda temsil ettiği bireyin takvim ve öncelik bilgilerine göre toplantı çağrılarına yanıt verir. Planlama etmeni aynı anda hem düzenleyici hem de davetli rolünde olabilir. Toplantı düzenlerken başka etmenlerden gelen toplantılara katılma davetlerini yanıt verebilir. Aynı şekilde başka bir etmenin düzenlediği toplantının planlamasında davetli olarak yer alırken kullanıcıdan gelen toplantı düzenleme talebi karşısında düzenleyici rolü ile toplantı düzenleme planlamasını başlatabilir. Böylece aynı anda hem düzenleyici hem de davetli rolünde bulunmuş olur.

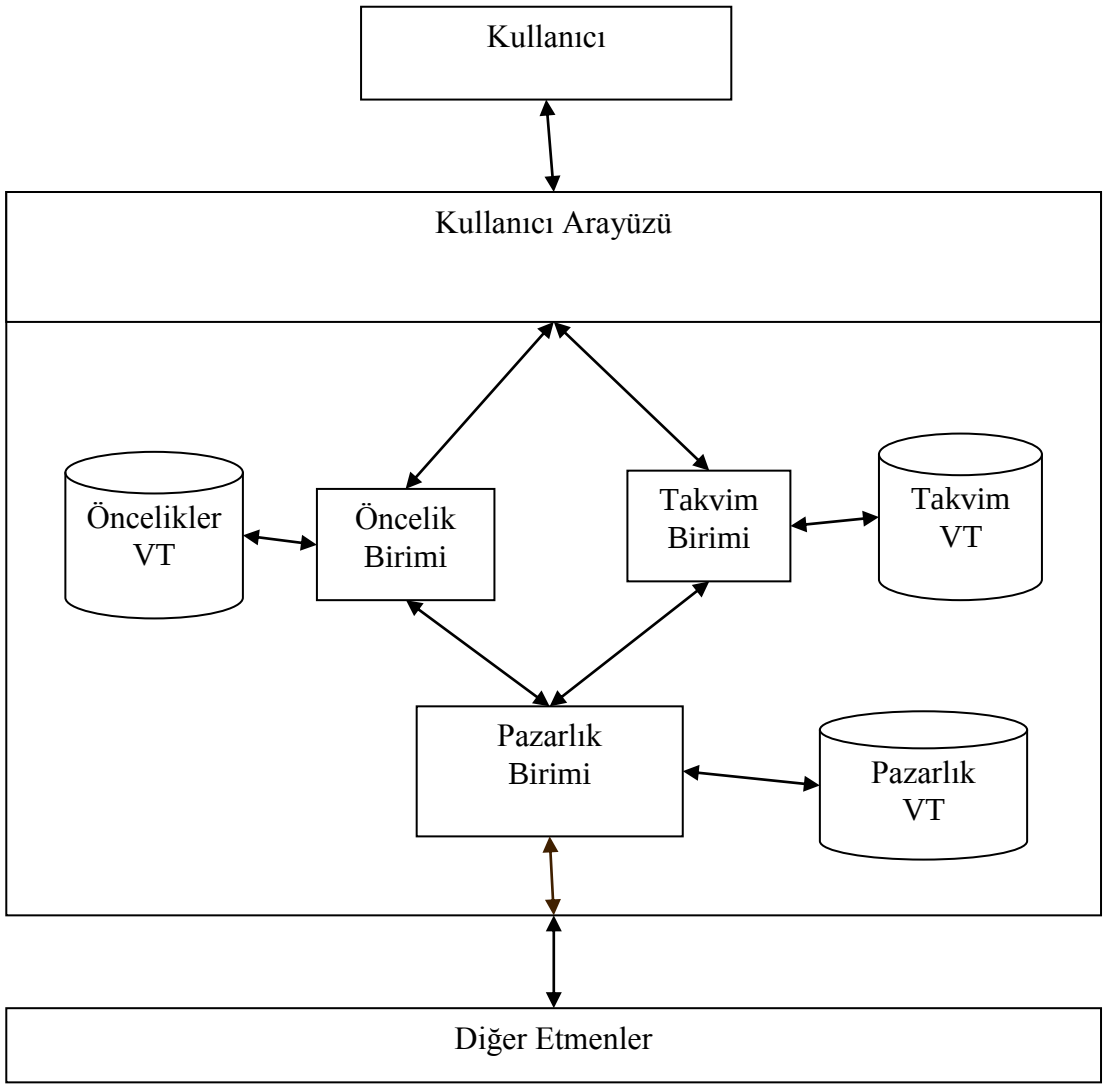
4.1.1. Planlama Etmeni İç Yapısı

Aşağıdaki şekilde planlama etmenin iç yapısı ve ortam ile olan ilişkisi gösterilmektedir.

4.1.1.1. Öncelikler Veritabanı : Bu veritabanında kullanıcının toplantı planlama işlemlerinde kullanılan öncelikleri tutulur. Bu öncelikler gerçek hayattaki toplantı planlama işlemlerinde de sıkça karşılaşılan kişilerin toplantı olmasını istemedikleri günler ya da bu günler içersinde belli zamanlardır. Örneğin Perşembe günleri saat 14:00 ile 15:00 arasında toplantı olmasın ya da cuma günün tamamında toplantı olmasın gibi bilgilerdir. Öncelikler veritabanında tarihler haftanın günü şeklinde tutulur, ve bu günler içersinde toplantı olmasının istenmediği zaman aralığı tutulur. Ayrıca bu veritabanı bireyin takvim bilgilerinin toplantı planlama işlemi sırasında diğer etmenler ile ne kadar paylaşılacağı bilgisini de tutar. Yani takvime dair herhangi bir bilgi diğer etmenlere gönderilebilip gönderilemeyeceği bilgisi veritabınında tutulur.

4.1.1.2. Öncelik Birimi: Öncelikler birimi öncelikler veritabanı üzerindeki işlemleri yapan; silme, ekleme işlemleri ve bu veritabanı ile ilgili gelen sorgulamaları

değerlendiren ve yanıtlayan birimdir. Bu birim pazarlık birimi ve kullanıcı arayüzü ile iletişim halindedir.



Şekil 4.2 Planlama etmeni iç yapısı ve ortam ile iletişimi

Pazarlık biriminden, pazarlık işlemleri sırasında gelen, önerilen, zaman aralığının ya da giden, önerilecek, zaman aralığının etmenin önceliklerine göre uygun olup olmadığının değerlendirilmesi şeklinde istekler alır. Bu istek karşısında öncelik birimi öncelik veritabındaki bilgileri ve pazarlık etmeni tarafından gönderilen zaman aralığını alarak işler ve bu işlemin sonucunda uygundur ya da değildir şeklinde bir yanıt döner. Öncelik birimi, yine pazarlık biriminden etmenin önceliklerine göre diğer etmenlere bilgi gönderimine izin verilip verilmeyeceğinin sorgulanması şeklinde bir istek alabilir. Bu durumda öncelik birimi bu isteğe etmenin öncelik veritabanına göre yanıt verir.

Öncelik biriminin iletişim halinde olduğu diğer bir birim ise kullanıcı arayüzüdür. Kullanıcı arayüzü öncelik veritabanı bilgilerini göstermek, bu bilgilerden bir kısmını silmek ve öncelikler veritabanına yeni bilgiler eklemek için öncelik biriminden isteklerde bulunur. Öncelik birimi öncelik veritabanını kullanarak bu isteklerde belirtilen işlemleri gerçekleştirir ve sonucunu kullanıcı arayüzüne bildirir.

4.1.1.3. Takvim Veritabanı : Takvim veritabanı etmenin temsil ettiği bireyin dolu olduğu zaman aralıkları bilgisini ve bunun yanında düzenlenen ya da davetli olunan toplantıların bütün bilgilerinin tutulduğu veritabanıdır. Düzenlenen toplantı için toplantı isteği sırasında girilen bütün bilgiler ve daha sonra pazarlık aşamasında belirlenen toplantı zamanı ve toplantı yeri bilgisi tutulmaktadır. Davetli olunan toplantı için ise toplantıyı düzenleyen etmen, toplantı zamanı, toplantı yeri gibi toplantı ile ilgili diğer etmenler tarafından gönderilen bilgiler tutulmaktadır. Bu veritabanında ayrıca temsil edilen bireyin istekte bulunduğu ancak başarılı şekilde planlanamayan toplantının bütün bilgileri de tutulmaktadır.

4.1.1.4. Takvim Birimi : Takvim birimi takvim veritabanı üzerinde sorgulama işlemleri yapan, pazarlık işlemleri sırasında pazarlık biriminden gelen istekleri alan ve takvim veritabanını kullanarak yanıt veren ve belli işlemleri gerçekleştiren birimdir. Takvim birimi pazarlık birimi ve kullanıcı arayüzü ile iletişim halindedir.

Pazarlık birimi toplantı pazarlık işlemleri sırasında takvim birimine diğer davetli etmenlere önerilecek ya da diğer davetli etmenlerden önerilen zaman aralığının takvime uygun olup olmadığının belirlenmesi şeklinde istekte bulunur. Takvim birimi takvim bilgilerini kullanarak ve bu bilgiler üzerinde arama yaparak bu isteğe yanıt verir. Takvim birimi yine pazarlık biriminden belli bir tarih aralığında belli bir zaman için takvimde boş yer bulması şeklinde istek alır. Bu istek karşısında takvim birimi takvim veritabanını uygun bir şekilde tarayarak belirtilen zaman aralığı için belirtilen tarihlerde takvim de uygun yer var mı diye daha sonra anlatılacak olan arama tekniği ile arama yapar. Ve uygun bir zaman aralığı bulur ise pazarlık birimine gönderir. Takvim birimi pazarlık biriminin isteği ile bir toplantıya belli zaman aralığı ayrılmak istendiği zaman belli bir zaman aralığı için takvimine blok koyar ve yine pazarlık biriminin isteği ile sonra belli bir toplantı için takviminde var olana blokları kaldırır. Takvim birimi pazarlık biriminden toplantı planlaması başarıldı, toplantıyı kaydet şeklinde bir istek alır. Bu istek karşısında takvim birimi ilgili toplantı ile ilgili bütün bilgileri toplantıyı düzenleyen olunma ya da toplantıya davetli olunma durumuna göre kayıt eder. Yine takvim birimi pazarlık biriminden düzenlenen toplantı planlaması başarılamadı, bilgileri kaydet şeklinde bir istek alır. Bu durumda takvim birimi ise düzenlenen toplantı ile ilgili bilgileri kayıt eder.

Takvim biriminin iletişim halinde olduğu diğer bir birim ise kullanıcı arayüzüdür. Takvim birimi takvim bilgileri ve planlanan ya da planlanamayan toplantı bilgileri ile ilgili değişik istekleri kullanıcı arayüzünden alır ve bu istekleri takvim veritabanını kullanarak yanıtlar.

4.1.1.5. Pazarlık Veritabanı : Pazarlık veritabanı düzenlenen toplantı ve davetli olunan toplantı bilgilerinin ve düzenlenen toplantı için öneri mesajı gönderilen etmen, ve öneriye verdikleri yanıt, davetli etmenlerin bildirdiği karşı öneri ve ret sebepleri bilgilerinin tutulduğu veritabanıdır. Pazarlık veritabanında bilgiler ilgili toplantı planlama oturumu boyunca tutulur ve toplantı planlama oturumu başarılı ya da başarısız sonlanınca silinir. Pazarlık veritabanında düzenlenen toplantı pazarlık işlemi sırasında kullanılan, davetli etmenlerin toplantı zamanı için gönderdikleri karşı öneriler tutulur. Ayrıca düzenlenen toplantılara davetli etmenler tarafından gönderilen herhangi bir öneriyi ret etmelerine sebep olan dolu oldukları zaman aralıkları bilgisi de pazarlık veritabanında tutulur. Düzenlenen toplantı sırasında mesaj gönderilen etmenler ve bunların önerilere verdikleri yanıtlarda da pazarlık veritabanında tutulur. Herhangi bir şekilde sonuçlanmamış ve henüz pazarlık aşamasında olunan etmenin düzenlediği ya da katıldığı toplantılar ile ilgili bütün bilgiler pazarlık veritabanında tutulur.

4.1.1.6. Pazarlık Birimi : Pazarlık birimi planlama etmenin en önemli birimi ve hatta planlama etmeninin beynidir diyebiliriz. Pazarlık birimi diğer birimleri kullanarak toplantı planlama işlemini yürütür, diğer etmenler ile mesajlaşmayı sağlar, diğer etmenlerden gelen toplantı isteklerine diğer birimleri de kullanarak yanıt verir. Pazarlık etmeni kullanıcı arayüzü, takvim birimi ve öncelik birimi ile iletişim halindedir.

Kullanıcı arayüzü yeni toplantı düzenlemesi için pazarlık biriminden istekte bulunur. Pazarlık birimi ise bu isteğe karşılık toplantı bilgilerini kullanarak, takvim ve öncelikler etmeni ile iletişim halinde toplantı düzenleme işlemini yürütmeye çalışır. Pazarlık birimi kullanıcı arayüzünden gelen istek ile JATLite Router'ına kayıt olma ve bağlanma işlemlerini de yürütür. Kullanıcı arayüzünden gelen bu istekler karşısında pazarlık birimi JATLite Router'ına kayıt olmak ya da bağlanmak için gerekli işlemleri yapar. JATLite Router'ından gelen kayıtlı etmen bilgilerini veritabanına yazar.

Yeni öneri bulmak ve etmene gelen toplantı isteğinin etmen için uygun mu bilgisini bulabilmek için takvim ve öncelik birimlerine isteklerde bulunur. Pazarlık sırasında geçici olarak kabul edilen zaman aralıkları için takvimde blok koymak için takvim biriminden istekte bulunur. Yine kabul edilen, başarı ile sonuçlandırılan ya da

etmenin düzenlediği ancak başarısızlıkla sonuçlanan toplantı bilgilerini kayıt etmek için takvim etmeninden istekte bulunur. Kendisine gelen bir toplantı zamanı önerisini ret ettiği zaman toplantı zamanını ret etmesine sebep olan zaman aralığı bilgisini karşı etmene bildirebilir mi sorusu için öncelik biriminden istekte bulunur. Düzenlenen bir toplantı planlama oturumu sırasında tutulan karşı öneri, toplantı önerisi mesajı gönderilen etmenler ve yanıtları, öneri ret sebep bilgilerini, düzenlenen ve davetli olunan toplantı bilgilerini pazarlık veritabanına yazar ve planlama işlemi bittiğinde ise siler. Pazarlık birimi diğer etmenlerden gelen toplantı isteklerine ise diğer birimleri kullanarak yanıt verir: kabul eder, ret eder, karşı öneri getirir. Pazarlık birimi toplantı planlama işlemi ile ilgili bütün mesajları hazırlar ve gönderir. Ayrıca diğer etmenlerden gelen bütün mesajları değerlendirir.

Pazarlık etmeni ayrıca yer etmenine yer listesi mesajı gönderir ve yer etmeninden gelen kayıtlı yer bilgilerini veritabanına yazar.

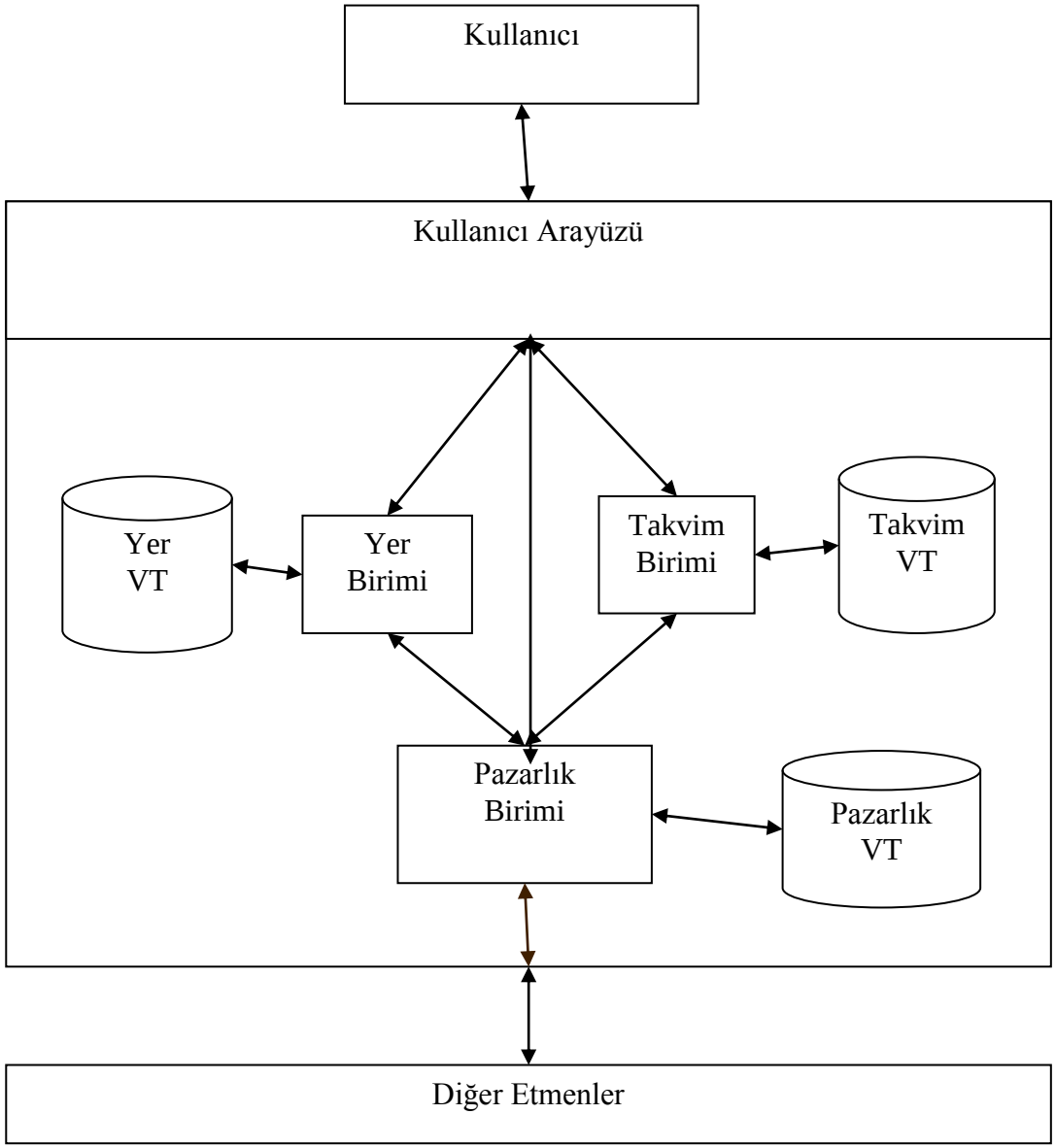
4.1.1.7. Kullanıcı Arayüzü : Kullanıcı arayüzü kullanıcı ile diğer birimler arasındaki iletişimi sağlar. Kullanıcıdan sisteme giriş(JATLite Router'ına bağlanma) isteğini alır ve bunun üzerine pazarlık biriminden istekte bulunur. Toplantı planlanması için kullanıcıdan bütün bilgileri alır ve bu bilgileri pazarlık birimine geçirerek ondan toplantı planlama işlemi için istekte bulunur. Ayrıca kullanıcı arayüzü takvim bilgilerinin izlenebilmesi için takvim biriminden değişik sorgu isteklerinde bulunur. Kullanıcı arayüzü öncelik bilgilerinin izlenebilmesi, silinebilmesi ve yeni öncelik bilgileri eklenebilmesi için öncelik bilgileri biriminden istekte bulunur. Kullanıcı arayüzünün sistemde kayıtlı olan bütün kullanıcıların isimlerinin ve yer etmeninde var olan toplantı yerleri isimlerinin tutulduğu ve pazarlık birimi tarafından JATLite Router'ından ve yer etmeninden gelen mesajlar sonrası güncellenen küçük bir veritabanı da vardır.

4.2. Yer Etmeni

Sistemde yer alan ikinci tip etmen olan yer etmeni toplantı yerleri bilgisinin tutulduğu, gelen toplantı isteklerini toplantı yerlerinin takvim bilgisi ve fiziksel bilgilerini dikkate alarak yanıtlayan etmen tipidir. Yer etmeni sistemde bir adet bulunur. JATLite Router'ının çalıştığı makinede çalışır. Yer sistemde olması istenen toplantı yerleri ve bunların kapasiteleri ve sahip oldukları araç gereçleri bu etmene tanımlanır. Yer etmeni genel olarak toplantı yerleri adına hareket eden bir etmendir. Yer etmeni rol olarak davetli etmen gibi hareket eder. Düzenleyici rolünde olan planlama etmeni davetli etmenlere mesaj gönderirken yer etmenine de mesaj gönderir.

4.2.1. Yer Etmeni İç Yapısı

Şekil 4.3'de yer etmenin iç yapısı ve yer etmenin ortam ile olan ilişkisi gösterilmektedir.



Şekil 4.3 Yer etmeni iç yapısı ve ortam ile iletişimi

4.2.1.1. Yer Veritabanı : Yer veritabanı sistemde tanımlı olan toplantı yerleri bilgilerinin, bunların sahip oldukları, toplantı sırasında kullanılabilecek, araç-gereç bilgilerinin ve toplantı yerinin kapasitelerinin tutulduğu veritabanıdır. Her bir toplantı yeri için eğer varsa toplantı araç gereçleri örneğin video, yansıtıcı, tepegöz gibi bilgiler tutulur. Ayrıca her bir toplantı yeri için o yerin bir toplantı sırasında alabileceği en fazla insan sayısı, kapasite, bilgisi tutulur.

4.2.1.2. Yer Birimi : Yer birimi yer veritabanı üzerinde işlemler yapan; güncelleme ve ekleme işlemleri, toplantı yerleri ile ilgili gelen istekleri alan ve yer veritabanını kullanarak yanıtlayan birimdir. Yer birimi pazarlık birimi ve kullanıcı arayüzü ile iletişim halindedir.

Yer birimine pazarlık biriminden belli bir toplantı yerinin toplantı için istenen fiziksel özelliklere sahip olup olmadığına bildirilmesi şeklinde istek alır. Yer birimi yer veritabanını kullanarak bu isteği yanıtlar. Yine pazarlık birimi yer etmeninden belli özelliklere sahip toplantı için fiziksel olarak uygun olan toplantı yeri bilgilerinin bildirilmesi için istekte bulunur. Yer etmeni toplantı için istenen özellikleri ve toplantı yerlerinin özelliklerini kullanarak bu isteği yanıtlar.

Yer birimin iletişim halinde olduğu diğer birim ise kullanıcı arayüzüdür. Kullanıcı arayüzü toplantı yerleri fiziksel bilgilerini göstermek, yeni toplantı yerleri eklemek ve var olan toplantı yerlerine yeni araç eklemek için yer biriminden istekte bulunur. Yer birimi bu istekleri yer veritabanını kullanarak gerçekleştirir ve sonuçları istekte bulunan kullanıcı arayüzüne bildirir.

4.2.1.3. Takvim Veritabanı : Takvim veritabanı sistemde bulunan ve yer etmeni tarafından temsil edilen bütün toplantı yerlerinin dolu oldukları zaman aralıkları bilgisinin ve bunun yanında toplantı yerinin kullanılacağı toplantıların bütün bilgilerinin tutulduğu veritabanıdır. Diğer etmenlerden gelen ve toplantı yerinin kullanılacağı toplantıların bilgileri bu veritabanında tutulmaktadır.

4.2.1.4. Takvim Birimi : Takvim birimi takvim veritabanı üzerinde sorgulama işlemleri yapan, pazarlık işlemleri sırasında pazarlık biriminden gelen istekleri alan ve takvim veritabanını kullanarak yanıt veren ve belli işlemleri gerçekleştiren birimdir. Takvim birimi pazarlık birimi ve kullanıcı arayüzü ile iletişim halindedir.

Takvim birimi pazarlık biriminden belli bir zaman aralığında belli bir toplantı yerinin dolu olup olmadığının belirlenmesi şeklinde bir istek ile karşılaşır. Bu durumda takvim birimi takvim veritabanını kullanarak ilgili toplantı yerinin o zaman aralığında dolu olup olmadığına dair bilgiyi pazarlık birimine bildirir. Takvim birimi belli bir toplantı yerinin, belli bir tarih aralığında ve belli bir zaman uzunluğunda uygun olan bilgisinin araştırılması ve sonucun bildirilmesi şeklinde pazarlık biriminden istek alır. Bu durumda takvim birimi takvim veritabanın kullanarak arama yapar ve sonucu pazarlık birimine bildirir. Takvim birimi pazarlık biriminin isteği ile bir toplantıya belli zaman aralığı ve toplantı yeri ayrılmak istendiği zaman belli bir zaman aralığı için takvimine blok koyar ve yine pazarlık biriminin isteği ile sonra belli bir toplantı için takviminde var olan blokları kaldırır. Takvim birimi pazarlık

biriminden toplantı planlaması başarıldı toplantıyı kaydet şeklinde bir istek alır. Bu istek karşısında takvim birimi ilgili toplantı ile ilgili bütün bilgileri kayıt eder.

Takvim biriminin iletişim halinde olduğu diğer bir birim ise kullanıcı arayüzüdür. Takvim birimi takvim bilgileri planlanan toplantı bilgileri ile ilgili değişik istekleri kullanıcı arayüzünden alır ve bu istekleri takvim veritabanını kullanarak yanıtlar.

4.2.1.5. Pazarlık Veritabanı : Pazarlık veritabanı diğer etmenlerden gelen toplantı istekleri bilgilerinin tutulduğu veritabanıdır. Pazarlık veritabanında bilgiler ilgili toplantı planlama oturumu boyunca tutulur ve toplantı planlama oturumu başarılı ya da başarısız sonlanınca silinir.

4.2.1.6. Pazarlık Birimi : Pazarlık birimi yer etmenin en önemli birimidir. Pazarlık birimi diğer birimleri kullanarak diğer etmenlerden gelen toplantı isteklerini yanıtlar ve pazarlık işlemini yürütür, diğer etmenler ile mesajlaşmayı sağlar. Pazarlık birimi kullanıcı arayüzü, takvim birimi ve yer birimi ile iletişim halindedir.

Pazarlık etmeni kullanıcı arayüzünden gelen istek ile JATLite Router'ına kayıt olma ve bağlanma işlemlerini de yürütür. Kullanıcı arayüzünden gelen bu istekler karşısında pazarlık birimi JATLite Router'ına kayıt olmak ya da bağlanmak için gerekli işlemleri yapar.

Etmene gelen toplantı isteğinin etmen için uygun mu bilgisini bulabilmek ve toplantı isteklerine karşı öneri bulmak için takvim ve öncelik birimlerine isteklerde bulunur. Pazarlık sırasında geçici olarak kabul edilen zaman aralıkları için takvimde blok koymak için takvim biriminden istekte bulunur. Yine kabul edilen, başarı ile sonuçlandırılan toplantı bilgilerini kayıt etmek için takvim etmeninden istekte bulunur. Daha önce geçici olarak blok konulmuş toplantılarla ilgili blokların kaldırılması için yine takvim biriminden istekte bulunur. Pazarlık birimi diğer etmenlerden gelen toplantı isteklerine ise diğer birimleri kullanarak yanıt verir: kabul eder, ret eder, karşı öneri getirir. Pazarlık birimi toplantı isteğini kabul ettiği zaman uygun olan yer bilgisini de kabul mesajı ile birlikte istekte bulunan etmene gönderir. İstekte bulunan etmenlere sistemde kayıtlı olan toplantı yerleri bilgisini de pazarlık etmeni gönderir. Pazarlık birimi toplantı isteklerine yanıtlar ile ilgili bütün mesajları hazırlar ve gönderir. Ayrıca diğer etmenlerden gelen bütün mesajları değerlendirir.

4.2.1.7. Kullanıcı Arayüzü : Kullanıcı arayüzü kullanıcı ile diğer birimler arasındaki iletişimi sağlar. Kullanıcıdan sisteme giriş(JATLite Router'ına bağlanma) isteğini alır ve bunun üzerine pazarlık biriminden istekte bulunur. Kullanıcı arayüzü takvim bilgilerinin izlenebilmesi için takvim biriminden değişik sorgu isteklerinde

bulunur ve sonuçlarını kullanıcıya gösterir. Kullanıcı arayüzü toplantı yerleri bilgilerinin izlenebilmesi, güncellenebilmesi ve yeni toplantı yeri bilgileri eklenebilmesi için yer biriminden istekte bulunur.

4.3. Toplantı Planlama Protokolü

Toplantı planlama algoritmasını anlatmadan önce algoritmanın şekillendirilmesinde etkili olan bazı konulara değinmek gerekmektedir. Bu konular algoritmasın etkinliğini ve hızlılığını direkt olarak etkilemektedir.

4.3.1. Bilgi Değişimi : Dağıtılmış toplantı planlama işleminde özel bilginin değişimi ile kazanılan verimlilik ile özel bilginin başkasına verilmesinden kaynaklı olarak ihlal edilen özel bilginin gizliliği ilkesi arasında bir denge vardır. Açıktır ki bütün davetli etmenler kendi takvim bilgilerinin ve öncelik bilgilerinin tamamını düzenleyici etmene gönderselerdi dağıtılmış toplantı planlama işi çok etkili bir şekilde ve tek yinelemeli arama işlemi ile sonlandırılabilinirdi. Ama bu bütün etmenlerin kendi özel bilgilerini sistemdeki diğer etmenler ile paylaşmaları anlamına gelir ki bu durum özel bilginin gizliliği ilkesinin ihlali sonucunu doğurur. Bu durumun karşıtı durum ise etmenlerin kendi takvim ve öncelik bilgilerini hiç bir şekilde diğer etmenler ile paylaşmamaları ve sorulara sadece evet veya hayır şeklinde cevaplamalarıdır. Bu durum bilginin tam olarak gizliliği olmak ile birlikte uygun bir çözüme ulaşmayı zorlaştırır, uygun çözüm bulunsa bile çözüme ulaşmak uzun zaman alır. Yani verimlilik ve çözüm zamanından kaybetmiş oluruz.

Bu tartışmaların sonucu olaraktan biz toplantı önerisi etmen tarafından ret edildiği zaman redde sebep olan zaman aralığını belirtmek için bazı takvim bilgilerini diğer etmenlere göndermelerine karar verdik. Bu şekilde bilgi değişimin sonucu olarak toplantı düzenleyen etmen diğer etmenlerin bir kısım bilgilerini elde etmiş olur, bu ise ona daha iyi öneriler üretmesinde yardımcı olur. Bir miktar bilgi değişimi ve bu bilgiden çıkarsama yapmak çözüm zamanını çok kısaltmaktadır[6]. Toplantı bilgileri sürekli değişebileceği için elde edilen bilgini sürekli geçerli olmayabilir. Dolayısıyla elde edilen bilginin bir toplantı planlama oturumunda tutulmasına karar verdik.

4.3.2. Arama Yönelimi: Dağıtılmış toplantı planlama işlemi etmenlerin kendi takvim ve öncelik bilgilerini kullanarak toplantı zamanı önerileri, karşı önerileri geliştirmek ve toplantı zamanı önerilerinin uygunluğunu bulmak için aramalar yaptıkları için dağıtılmış bir arama işlemi olarak düşünülebilinir. Takvim üzerinde toplantı için uygun zaman aralığının ne şekilde aranacağı ve birden fazla takvimde uygun zaman bulunduğu zaman hangi bilginin öneri ya da karşı öneri olarak

alınacağı sorusunun cevabı olarak üç tip arama yönelimi vardır: doğrusal önce, doğrusal az yoğunluk ve sıradüzensel arama yönelimi[7].

Doğrusal önce arama yöneliminde, etmen toplantı planlama işini takviminde mümkün olduğunca yakın bir zaman içinde yapmaya çalışır. Etmen toplantı planlama için başlangıç zamanından başlayarak takviminde aramaya başlar ve uygun olarak bulduğu ilk zaman dilimini öneri ya da karşı öneri olarak alır ve arama işlemini sonlandırır. Bu arama yönelimi ile yapılan toplantı planlama işlemleri sonucunda etmenin takviminde öne yığılma oluşur.

Doğrusal az yoğunluk arama yöneliminde ise etmen toplantıyı takviminde en az yoğunluk olan bölgeye planlamaya çalışır. Etmen toplantı için öneri ya da karşı öneri geliştirirken takvimini tarar ve takvimindeki bütün uygun zaman aralıklarını bulur ve bulunan bu zaman aralıklarından takviminin en az yoğunluklu bölgesinde bulunan zaman aralığını çözüm olarak alır. Doğal olarak bu tür bir arama yönelimi kullanarak toplantı planlamasına katılan etmeninin takvim bilgisi dolu olan zamanların eşit dağılmış bir takvim ortaya çıkarır.

Sıradüzensel arama yöneliminde ise düzenleyici etmen toplantıyı bütün davetli etmenlerin birleştirilmiş takvim bilgilerinde en az yoğunluğa sahip bölgesinde planlamaya çalışır. Bu arama şu şekilde çalışır ; öncelikle davetli etmenlerin en az uygun zamanları olan en az yoğun oldukları hafta bulunur, daha sonra bu hafta içinde bütün davetli etmenlerin uygun zamanın olduğu en az yoğun oldukları gün bulunur, daha sonra bu gün içindeki bütün davetli etmenlerin uygun zamanları olan en az yoğun oldukları zaman bulunur. İşlem sıradüzensel olarak devam eder. Ve sonuçta bulunan toplantı zamanı bütün davetli etmenlerin takvim bilgilerinin en az yoğunlukta olduğu bölümdür.

Sunulan algoritmada doğrusal önce arama yönelimini kullanılmıştır.

4.3.3. Karşı Öneri: Dağıtılmış toplantı planlama işlemi çözüme bütün katılımcıların katkılarıyla ulaşan bir ortak problem çözme işlemidir. Davetli etmenler soruları basitçe evet veya hayır şeklinde cevaplayarak edilgen olabilirler ya da düzenleyici etmenlerin önerilerini ret ettikleri zaman kendi bilgilerini kullanarak karşı öneri geliştirerek etken de olabilirler. Davetli etmenler eğer edilgen olurlarsa bütün planlama işini, uygun zaman bulma işlemini sadece düzenleyici etmen yapacaktır ve çözüm zamanı uzayacaktır. Bunun aksine davetli etmenlerin karşı öneri geliştirmesi ve planlamaya bu şekilde katkıda bulunmaları toplantı planlama işlem zamanını kısaltmaktadır[8]. Sunulan algoritmada davetli etmenler bir öneriyi ret ettikleri

zaman kendi bilgilerini kullanarak öneri geliştirirler ve düzenleyici etmenlerde bu karşı önerileri değerlendirirler.

4.3.4. Toplantı Planlama Algoritması

Toplantı isteğinde bulunan kullanıcı aşağıdaki bilgileri düzenleyici etmene sağlamalıdır:

- Toplantı ad ve amacı,
- Toplantı uzunluğu, dakika olarak,
- Toplantı planlama işinin olacağı dönem bilgisi. Toplantı planlama için başlangıç ve bitiş zamanı. Örneğin 11.03.2002 ve 15.03.2002 gibi,
- İlk toplantı önerisi için tarih ve zaman bilgisi(seçimlik),
- Önemli ve sıradan davetliler ayrı ayrı olarak katılımcı bilgisi,
- En alt sıradan katılımcı sayısı. Toplantıya katılması zorunlu olan, toplantının başarılı planlanması için olumlu yanıt vermesi gereken en az sıradan katılımcı sayısı,
- Toplantı önceliği:1ile 4 arasında bir sayı,
- Toplantı için gerekli olan araçlar. Örneğin televizyon, video, tepegöz…
- Toplantının olmasını istendiği yer bilgisi (seçimli),

1. Düzenleyici etmen toplantı düzenlenmesi için gereken bilgileri ve toplantı düzenleme isteğini kullanıcısından alır. Eğer başlangıç öneri toplantı zamanı kullanıcı tarafından girilmiş ise bu değer düzenleyici etmen tarafından başlangıç önerisi olarak dikkate alınır. Eğer başlangıç toplantı zaman aralığı uygun değilse ya da başlangıç toplantı zaman aralığı girilmemiş ise etmen kendi takvim, öncelik bilgileri ve toplantı bilgilerini kullanarak boş bir zaman aralığı bulmaya çalışır. Eğer boş bir zaman aralığı bulamaz ise toplantı düzenleme işlemi başarısızlıkla sonuçlanmıştır. Eğer bulmuşsa , bütün davetli etmenlere ve yer etmenine öneri mesajı gönderir. Yer etmenine ve davetli etmenlere gönderilen mesajlar farklı biçimdedir. Düzenleyici etmen önerilen zaman aralığı için takvimine blok koyar.

2. Toplantı önerisini alan davetli etmen, öneriyi kendi takvim ve öncelik bilgilerini kullanarak değerlendirir. Bu değerlendirme sonucunda öneriyi kabul ya da ret eder.

2.a. Eğer toplantı önerisini kabul ediyorsa önerilen zaman aralığı için takvimine blok koyar ve daha önceden bu toplantı için takviminde var olan blokları kaldırır. Daha sonra kabul mesajını düzenleyici etmene gönderir.

2.b. Eğer önerilen zaman aralığı davetli etmen için uygun değilse, davetli etmen toplantının özelliklerini, kendi takvim bilgilerini ve kendi öncelik bilgilerini kullanarak karşı öneri bulmaya çalışır. Karşı öneri bulduysa bu karşı öneriyi içeren bir mesajı düzenleyici etmene gönderir. Mesaj ayrıca düzenleyici etmen tarafından önerilen toplantı zaman aralığını ret etmesine sebep olan, takviminin dolu olduğu zaman dilimini de içerir eğer önceliklerinde karşı etmenlere ret sebeplerini bildirmesine izin verilmiş ise. Karşı öneri bulamadı ise ret mesajını düzenleyici etmene gönderir. Davetli etmen bu toplantı için takviminde var olan önceki blokları kaldırır ve karşı öneri gönderdi ise karşı öneri olarak gönderdiği zaman aralığı için takvime blok koyar.

3. Toplantı önerisini alan yer etmeni kendi takvimini, toplantı için istenen toplantı yer özelliklerini ve sahip olduğu toplantı yerlerinin özelliklerini dikkate alarak öneriyi değerlendirir. Eğer değerlendirme sonucunda uygun yer bulduysa bu durumda düzenleyici etmene uygun olan toplantı yeri bilgisi ile birlikte kabul mesajı gönderir. Eğer toplantı için istenen fiziksel özelliklere sahip herhangi bir toplantı yeri bulamadı ise düzenleyici etmene ret mesajı gönderir. Eğer önerilen zamanda uygun yer bulamadı ise, karşı öneri bulmaya çalışır. Daha sonra yer etmeni ret mesajını, eğer bulduysa karşı öneri olarak bulduğu zaman aralığını ve yer bilgisini düzenleyici etmene mesaj olarak gönderir. Yer etmeni bu toplantı için takviminde var olan blokları kaldırır ve kabul ettiği zaman aralığı ya da karşı öneride bulunduğu zaman aralığı ve toplantı yeri için takvimine blok koyar.

4. Yanıtları alan düzenleyici etmen yanıtları değerlendirir:

4a. Eğer düzenleyici etmen bütün davetli etmenlerden ve yer etmeninden kabul mesajı aldıysa, bütün davetli etmenlere ve yer etmenine toplantı yeri bilgisi ile birlikte başarılı planlama mesajı gönderir ve takviminde bu toplantı için var olan blok işlemini sonlandırır.

4b. Eğer düzenleyici etmen bütün önemli davetli etmenlerden, yer etmeninden ve daha önceden belirlenmiş sayıdaki sıradan davetli etmenlerden kabul mesajı almış ise, düzenleyici etmen önemli davetli etmenlere, yer etmenine ve toplantı için kabul mesajı gönderen sıradan davetli etmenlere toplantı yeri bilgisi ile birlikte başarılı düzenleme mesajı gönderir.

Bu toplantı için ret mesajı gönderen sıradan davetli etmenlere ise toplantı iptal mesajı gönderir. Takviminde bu toplantı için var olan blok işlemini sonlandırır.

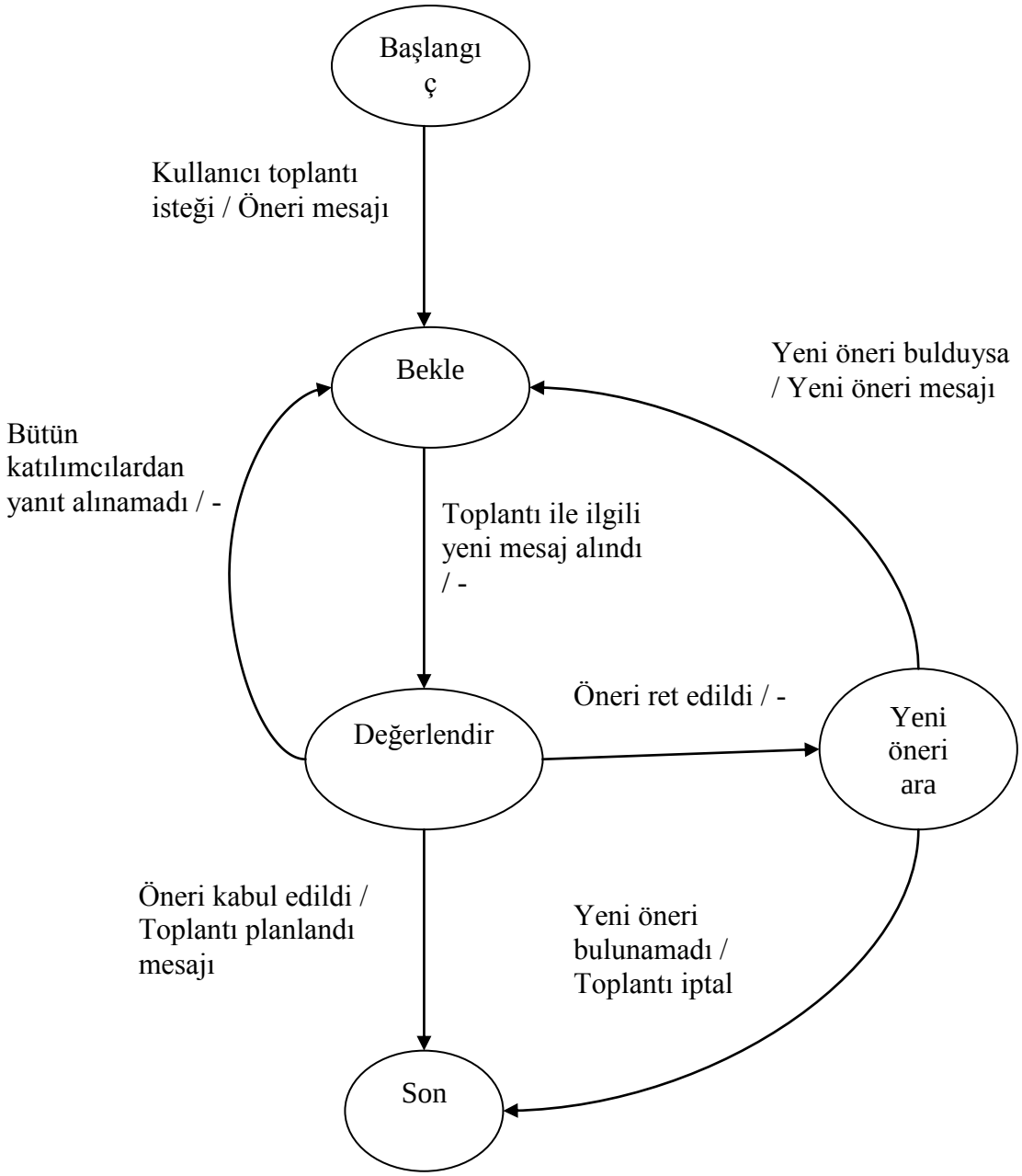
4c. Eğer düzenleyici etmen yer etmeninden, bütün önemli davetli etmenlerden ya da daha önceden belirlenmiş sayının altında sıradan davetli etmenden kabul mesajı alamadı ise, yer etmeni ve davetli etmenlerden gelen karşı önerileri 4a ve 4b adımında belirttiğimiz şekilde değerlendirir. Eğer 4a ve 4b adımında belirttiğimiz kriterlere göre ortak bir karşı öneri bulundu ise bulunan bu zaman aralığı bütün etmenlere yeni öneri olarak gönderilir ve 2. adıma geçilir ve bu yeni toplantı zaman aralığı için takvime blok konulur ve daha önce bu toplantı için takvimde var olan bloklar kaldırılır. Eğer 4a ve 4b adımlarında belirttiğimiz kriterlere uygun karşı öneri bulunamaz ise 6. adıma geçilir ve daha önce bu toplantı için takvimde var olan bloklar kaldırılır.

5. Başarılı planlama mesajını alan davetli etmenler ve yer etmeni toplantı yeri bilgisini güncellerler ve takvimlerindeki bu toplantı için var olan bloklarını sonlandırılar. İptal mesajını alan davetli etmenler ve yer etmeni ise takvimlerinde bu toplantı için var olan blokları kaldırırlar ve toplantı için var olan bilgileri silerler.

6.Düzenleyici etmen davetli etmenlerden ve yer etmeninden toplantı öneri zamanları için aldığı ret sebeplerini pazarlık veritabanında saklarlar. Daha sonra toplantı için bu bilgileri, takvim bilgilerini, öncelik bilgilerini ve toplantının bilgilerini kullanarak yeni bir öneri bulmaya çalışır. Eğer bu işlem sonucunda toplantı planlama başlangıç ve bitiş tarih aralığında yeni bir öneri, zaman aralığı, bulamadı ise toplantı planlama işlemini iptal eder ve bütün davetli etmenler ve yer etmenine toplantı planlama iptal mesajı gönderir ve kendi takviminde bu toplantı için var olan blokları kaldırır. Eğer bu toplantı için toplantı planlama başlangıç ve bitiş tarih aralığında yeni bir öneri, zaman aralığı buldu ise bu zaman aralığını davetli etmenlere ve yer etmenine gönderir ve bulunan toplantı zaman aralığı için takvimine blok koyar ve 2. adımdan devam eder.

4.3.4.1. Etmenlerin Planlama İşlemi Sırasındaki Durum Diyagramı

Şekil 4.4'de düzenleyici etmenlerin algoritmaya göre durum diyagramları görülmektedir. İki durum arasındaki geçiş için a/b şeklinde bir gösterim kullanıldı. a etmene gelen mesajı veya koşulu gösterirken b etmen tarafından gönderilen mesajı göstermektedir. b yerine bazen konan "-" ise a koşulu gerçekleştiğinde ya da a mesajı geldiğinde herhangi bir mesaj gönderilmeyeceğini belirtmektedir.



Şekil 4.4 Düzenleyicinin toplantı planlama işlemi sırasındaki durum diyagramı

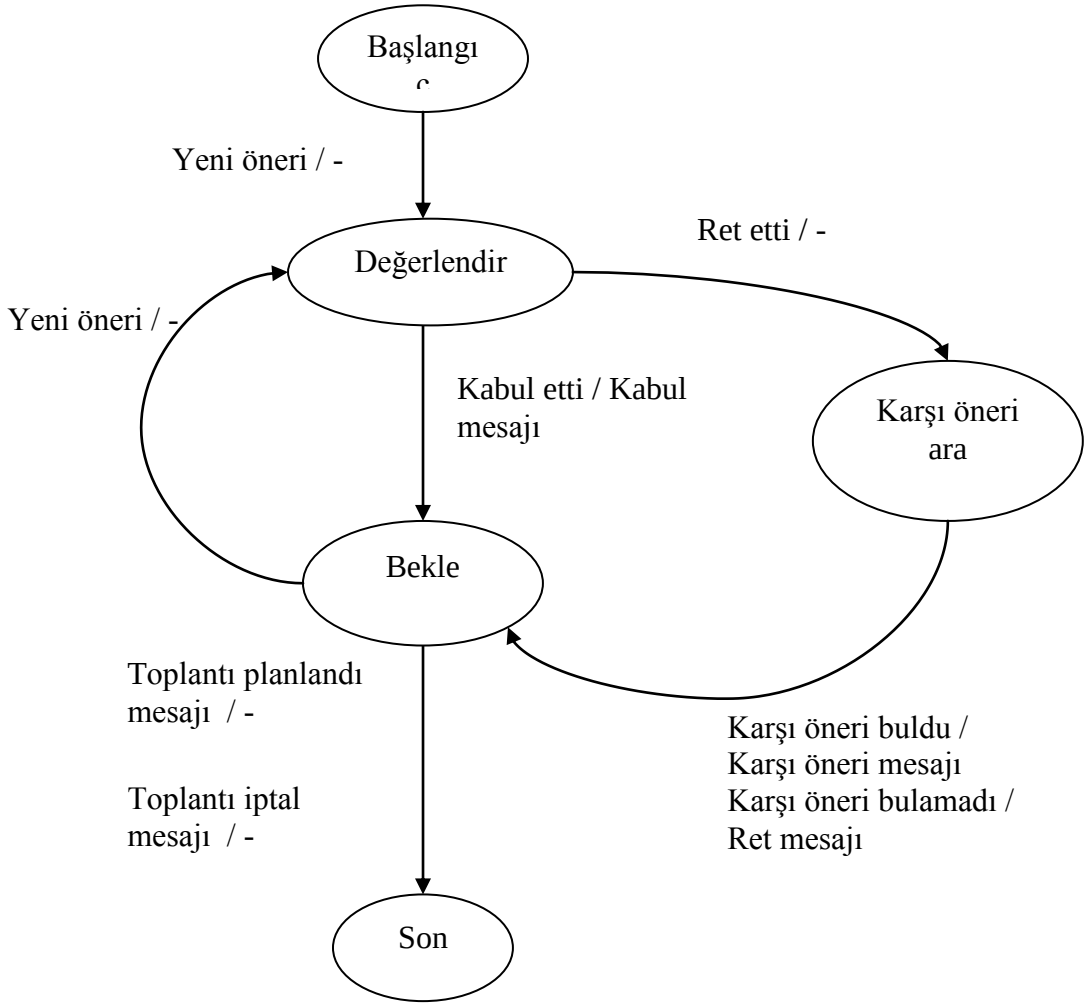
Düzenleyicinin durum diyagramına kısaca bakacak olursak; başlangıç durumunda iken kullanıcıdan toplantı isteği gelir. Bu durumda davetli etmenlere ve yer etmenine toplantı önerisi mesajı gönderir ve bekleme durumuna geçer. Daha sonra her yeni mesaj geldiğinde düzenleyici etmen değerlendirme durumuna geçer ve gelen mesajları değerlendirir. Eğer bütün davetli etmenlerden ve yer etmeninden ilgili toplantı için mesaj gelmemiş ise düzenleme etmeni tekrar bekleme durumuna geçer. Eğer bütün davetli etmenlerden ve yer etmeninden ilgili toplantı için mesaj gelmiş ve öneri kabul edilmiş ise davetli etmenlere ve yer etmenine toplantı planlandı mesajı göndererek düzenleyici etmen son durumuna geçer. Eğer bütün davetli etmenlerden

ve yer etmeninden yanıt gelmiş ve öneri kabul edilmemiş ise düzenleyici etmen yeni öneri ara durumuna geçer. Bu durumdan, yeni öneri bulmuş ise bulunan yeni öneriyi davetli etmenlere göndererek düzenleyici etmen bekleme durumuna geçer. Eğer yeni öneri bulamamış ise bütün davetli etmenlere ve yer etmenine toplantı iptal mesajı gönderir ve düzenleyici etmen son durumuna geçer.

Şekil 4.5 davetli etmenlerin algoritmaya göre durum diyagramı görülmektedir. Davetli etmenin ve toplantı planlama işleminde davetli etmen olarak değerlendirilebilecek yer etmenin durum diyagramına kısaca bakacak olursak; başlangıç durumunda iken düzenleyici etmenden gelen toplantı önerisi mesajı ile davetli etmen değerlenir durumuna geçer.

Bu durumda iken eğer toplantı önerisini kabul ederse düzenleyici etmen kabul mesajı göndererek davetli etmen bekleme durumuna geçer. Değerlendir durumunda iken eğer toplantı önerisini ret ederse davetli etmen karşı öneri ara durumuna geçer. Karşı öneri bulur ise düzenleyici etmene karşı öneri mesajı göndererek davetli etmen bekle durumuna geçer. Karşı öneri ara durumunda iken eğer karşı öneri bulamaz ise düzenleyici etmene ret mesajı göndererek davetli etmen bekle durumuna geçer. Bekle durumunda iken eğer düzenleyici etmenden yeni öneri mesajı alır ise davetli etmen tekrar değerlendir durumuna geçer. Eğer bekleme durumunda iken düzenleyici etmenden toplantı planlandı ya da toplantı iptal mesajı alır ise davetli etmen son durumuna geçer.

Davetli etmenin ve toplantı planlama işleminde davetli etmen olarak değerlendirilebilecek yer etmenin durum diyagramına kısaca bakacak olursak; başlangıç durumunda iken düzenleyici etmenden gelen toplantı önerisi mesajı ile davetli etmen değerlenir durumuna geçer. Bu durumda iken eğer toplantı önerisini kabul ederse düzenleyici etmen kabul mesajı göndererek davetli etmen bekleme durumuna geçer. Değerlendir durumunda iken eğer toplantı önerisini ret ederse davetli etmen karşı öneri ara durumuna geçer. Karşı öneri bulur ise düzenleyici etmene karşı öneri mesajı göndererek davetli etmen bekle durumuna geçer. Karşı öneri ara durumunda iken eğer karşı öneri bulamaz ise düzenleyici etmene ret mesajı göndererek davetli etmen bekle durumuna geçer. Bekle durumunda iken eğer düzenleyici etmenden yeni öneri mesajı alır ise davetli etmen tekrar değerlendir durumuna geçer. Eğer bekleme durumunda iken düzenleyici etmenden toplantı planlandı ya da toplantı iptal mesajı alır ise davetli etmen son durumuna geçer.



Şekil 4.5 Davetlinin toplantı planlama işlemi sırasındaki durum diyagramı

4.4. Sistemde Alınan ve Gönderilen Mesaj Tipleri ve Açıklamaları:

Sistemde kullanılan etmenler arası haberleşme için KQML kullanıldı. KQML daha önce bahsettiğimiz gibi etmenler arası bilgi alış verişi kullanılan bir dildir. Standart KQML performative'lerine ek olarak sistemi gerçeklenirken propose, counter-propose, accept, list-location, location-list, reject performative'leri önerildi. Aşağıda sistemde kullanılan ve önerilen performative'lerin biçimi ve kullanılış amacı bulunmaktadır. Sistemde herhangi bir toplantı meetingId ile ayrılmaktadır. MeetingID bütün sistemde tekdir ve etmenler arasındaki haberleşme sırasında :content bölümünde mutlaka bulunur.

(propose
:sender <sözcük>
:receiver <sözcük>
:reply-with <sözcük>
:language <sözcük>

:ontology <sözcük>
:content <bilgi>)

Toplantı düzenleyen etmen tarafından davetli etmenlere ve yer etmenine toplantı bilgilerini ve toplantı zaman önerilerini bildirmek amacıyla gönderilir. Bilgi olarak ilk toplantı önerisi ise toplantı hakkında bütün bilgiler değilse toplantı için önerilen zaman bilgisi bulunur.

(counter-propose
:sender <sözcük>
:receiver <sözcük>
:in-reply-to <sözcük>
:language <sözcük>
:ontology <sözcük>
:content <bilgi>)

Toplantıya davetli etmenler ya da yer etmeni tarafından toplantı düzenleyen etmene, toplantı düzenleyen etmenin daha önce göndermiş olduğu toplantı önerisini kabul etmediklerini ve kendilerinin karşı önerilerini bildirmek amacıyla gönderilir.

(reject
:sender <sözcük>
:receiver <sözcük>
:in-reply-to <sözcük>
:language <sözcük>
:ontology <sözcük>
:content <bilgi>)

Toplantıya davetli etmenler ya da yer etmeni tarafından toplantı düzenleyen etmene, toplantı düzenleyen etmenin daha önce göndermiş olduğu toplantı önerisini kabul etmediklerini bildirmek amacıyla gönderilir.

(accept
:sender <sözcük>
:receiver <sözcük>
:in-reply-to <sözcük>
:language <sözcük>
:ontology <sözcük>
:content <bilgi>)

Toplantıya davetli etmenler ya da yer etmeni tarafından toplantı düzenleyen etmene, toplantı düzenleyen etmenin daha önce göndermiş olduğu toplantı önerisini kabul ettiklerini bildirmek amacıyla gönderilir.

(insert

:sender <sözcük>
:receiver <sözcük>
:reply-with <sözcük>
:language <sözcük>
:ontology <sözcük>
:content <bilgi>)

Toplantı düzenleyen etmen tarafından davetli etmenlere ve yer etmenine daha önce accept mesajı ile kabul etmiş oldukları toplantı zamanın ilgili toplantı için kesinleştiğini bildirmek amacıyla gönderilir.

(delete

:sender <sözcük>
:receiver <sözcük>
:reply-with <sözcük>
:language <sözcük>
:ontology <sözcük>
:content <bilgi>)

Toplantı düzenleyen etmen tarafından davetli etmenlere ve yer etmenine ilgili toplantının planlanamadığını bildirmek amacıyla gönderilir.

(list-location

:sender <sözcük>
:receiver <sözcük>
:reply-with <sözcük>
:language <sözcük>
:ontology <sözcük>
:content <null>)

Sistemdeki planlama etmenleri tarafından sistemde kayıtlı toplantı yerleri bilgisini almak için yer etmenine gönderilir.

(location-list

:sender <sözcük>
:receiver <sözcük>
:in-reply-to <sözcük>
:language <sözcük>
:ontology <sözcük>
:content <bilgi>)

Yer etmeni tarafından list-location mesajı gönderen planlama etmenine sistemde kayıtlı bulunan yer bilgilerini bildirmek amacıyla gönderilir.

(list-agent
:sender <sözcük>
:receiver <sözcük>)

Planlama etmeni tarafından JATLiteRouter'ına sistemde kayıtlı olan etmen bilgilerini öğrenmek amacıyla gönderilir.

(registered-agent
:sender <sözcük>
:receiver <sözcük>
:content <bilgi>)

JATLiteRouter'ı tarafından list-agent mesajı gönderen planlama etmenine sistemde kayıtlı olan etmen bilgilerini bildirmek amacıyla gönderilir.

(error
:sender <sözcük>
:receiver <sözcük>
:language <sözcük>
:ontology <sözcük>
:content <bilgi>)

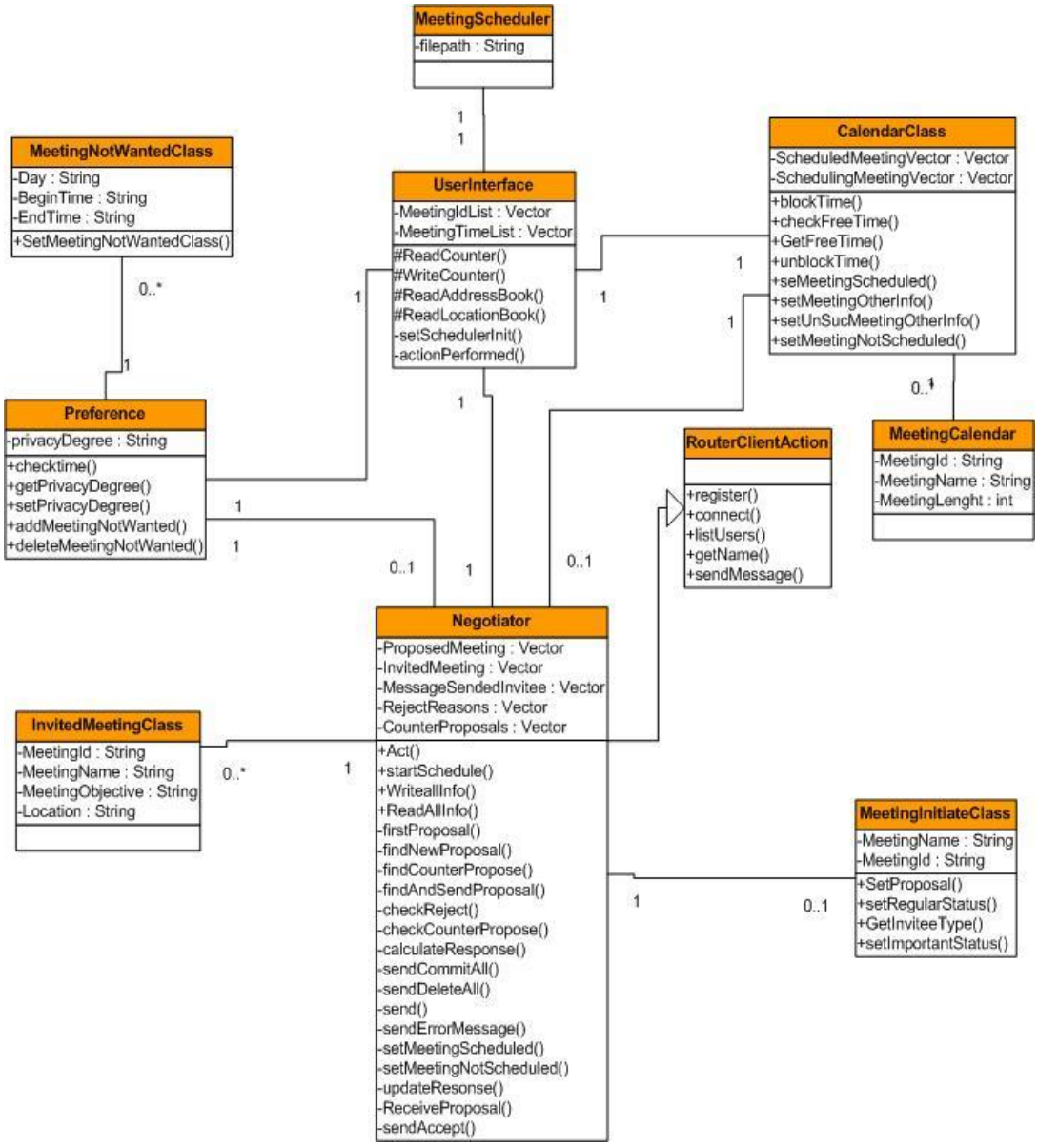
Herhangi bir mesajda belirtilen meetingID bilgisi mesajı alan etmenin veritabanında bulunamadığı zaman mesaj gönderen etmene gönderilir.

4.5. Uygulama Sınıfları Mimarisi

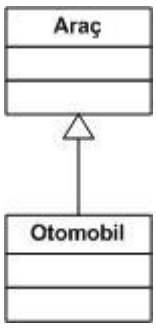
Sistemde daha önceki bölümlerde de aktarıldığı gibi temel olarak iki tip etmen bulunmaktadır: planlama etmeni ve yer etmeni. Dolayısıyla gerçekleme işlemi sırasında iki adet paket yazılmıştır: scheduler package ve location package.

4.5.1. Scheduler Paketi Detayı

Scheduler paketi kişiler adına işlem yapan planlama etmeninin gerçeklenmesidir. Şekil 4.6' da scheduler paketi sınıf diyagramı gözükmektedir. UserInterface sınıfı kullanıcı arayüzünün gerçeklenmesidir. Kullanıcı işlemlerinin gerçekleştirilmesi için gerekli olan işlemler bu sınıf tarafından sağlanır. Kullanıcı isteklerini yorumlar ve işlemleri yerine getirmek için gerekli metotları çağırır. CalendarClass, Preference, Negotiator sınıflarından nesneler bu sınıf tarafından yaratılır.



Şekil 4.6 scheduler paketi sınıf diyagramı

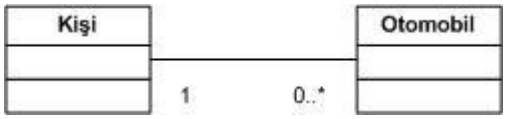


Otomobil sınıfının Araç sınıfından miras aldığını gösterir.

\+ public anlamına gelir

\- private anlamına gelir

protected anlamına gelir



Her otomobilin bir sahibi vardır, her kişinin 0 ve daha fazla otomobili vardır.

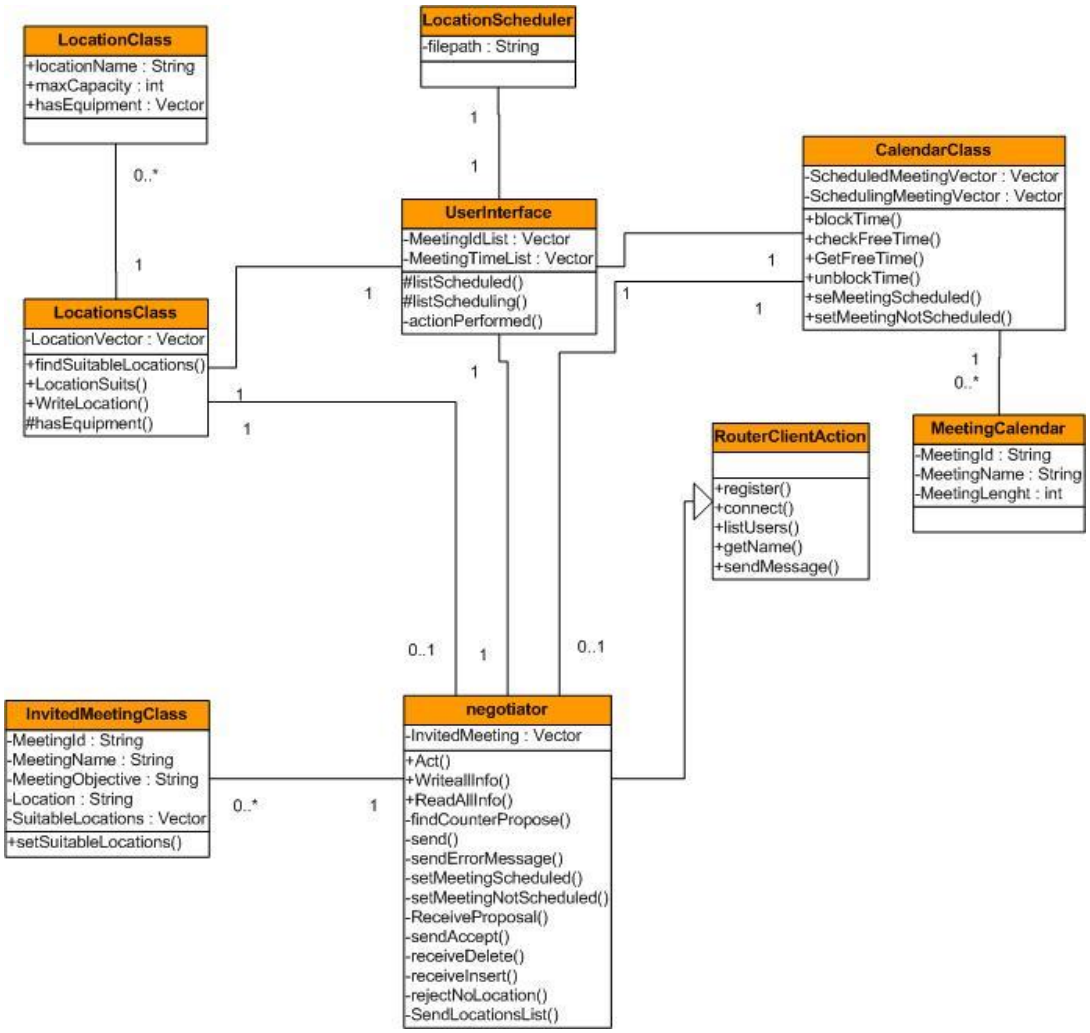
Etmenin takvimini temsil eden CalendarClass sınıfı, etmenin takvimi üzerindeki işlemleri ve sorguları gerçekleştirmek üzere gerekli metotlara sahiptir. Takvimde boş zaman dilimi bulma, takvimdeki belli zaman dilimlerinin boş olup olmadığını kontrol etme, takvimdeki belli zaman dilimlerini işaretleme ve işareti kaldırma işlemleri için metotları vardır. Ayrıca planlanan ya da planlanamayan toplantı bilgilerini yazmak için metotları vardır. Etmenin takvimindeki tek bir toplantıyı temsil eden MeetingCalendar sınıfından nesnelere sahiptir.

Etmenin önceliklerini temsil eden Preference sınıfı, etmenin öncelikleri üzerinde işlemleri ve sorguları yürütmek için gerekli metotlara sahiptir. Preference sınıfı belli bir zaman dilimini öncelik bilgilerine göre uygunluğunu kontrol etmek, yeni öncelik bilgisi eklemek, silmek ve öncelik bilgisi değerini bildirmek için metotlara sahiptir. Preference sınıfı toplantı istenmeyen günler bilgisini temsil eden MeetingNotWanted sınıfından nesnelere sahiptir.

JATLite paketinin sunduğu RouterActionClient sınıfından türetilmiş olan Negotiator sınıfı, etmenler arası iletişimi, mesaj gönderme, mesaj alma işlemlerini yürütür. Ayrıca yukarda anlattığımız algoritmanın yürütücüsü Negotiator sınıfıdır. Negotiator sınıfı kendisine gelen mesajları değerlendirir ve bu mesajlara karşılık mesajlar gönderir. Negotitor sınıfı toplantı planlama işlemi sırasında bütün mesajları yaratır diğer etmenlere gönderir. Kendi gönderdiği mesajlara karşılık gelen mesajları değerlendirir. Negotiator sınıfı davetli olduğu toplantı bilgilerini temsil eden InvitedMeetingClass sınıfından nesnelere, kendi düzenlediği toplantıları temsil eden MeetingIniateClass sınıfından nesnelere sahiptir.

4.5.2. Location Paketi Detayı

Location paketi toplantı yerleri adına işlem yapan yer etmeninin gerçeklenmesidir. UserInterface sınıfı kullanıcı arayüzünün gerçeklenmesidir. Toplantı yerleri işlemlerinin gerçekleştirilmesi için gerekli olan işlemler bu sınıf tarafından sağlanır. Kullanıcı isteklerini yorumlar ve işlemleri yerine getirmek için gerekli metotları çağırır. CalendarClass, LocationsClass, Negotiator sınıflarından nesneler bu sınıf tarafından yaratılır.



Şekil 4.7 location paketi sınıf diyagramı

Yer etmeninin takvimini temsil eden CalendarClass sınıfı, etmenin takvimi üzerindeki işlemleri ve sorguları gerçekleştirmek üzere gerekli metotlara sahiptir. Takvimde boş zaman dilimi bulma, takvimdeki belli zaman dilimlerinin boş olup olmadığını kontrol etme, takvimdeki belli zaman dilimlerini işaretleme ve işareti kaldırma işlemleri için metotları vardır. Ayrıca planlanan ya da planlanamayan toplantı bilgilerini yazmak için metotları vardır. Etmenin takvimindeki tek bir toplantıyı temsil eden MeetingCalendar sınıfından nesnelere sahiptir.

Toplantı yerlerini temsil eden LocationsClass sınıfı, toplantı yerleri üzerinde işlemleri ve sorguları yürütmek için gerekli metotlara sahiptir. LocationsClass sınıfı toplantı için ihtiyaç duyulan özelliklere göre uygun toplantı yeri var mı ya da bir toplantı yeri belli bir toplantı için uygun mu sorularını yanıtlamak için metotlara sahiptir. Ayrıca bu sınıf yeni toplantı yeri eklemek ve silmek için metotlara sahiptir. LocationsClass sınıfı toplantı yerlerini temsil eden LocationClass sınıfından nesnelere sahiptir.

JATLite paketinin sunduğu RouterActionClient sınıfından türetilmiş olan Negotiator sınıfı, etmenler arası iletişimi, mesaj gönderme, mesaj alma işlemlerini yürütür. Negotiator sınıfı kendisine gelen mesajları değerlendirir ve bu mesajlara karşılık mesajlar gönderir. Negotiator sınıfı davetli olduğu toplantı bilgilerini temsil eden InvitedMeetingClass sınıfından nesnelere sahiptir.

4.6. Kullanıcı Arayüzleri

Daha önceki bölümlerde anlatılan sistemi gerçeklemek için temelde yer etmeni ve planlama etmeni olmak üzere iki farklı program yazılmıştır. Bu programlar kullanıcı girişlerini sağlamak ve yürütülen işlemlerin sonuçlarını izlemek için kullanıcı arayüzlerine sahiptirler.

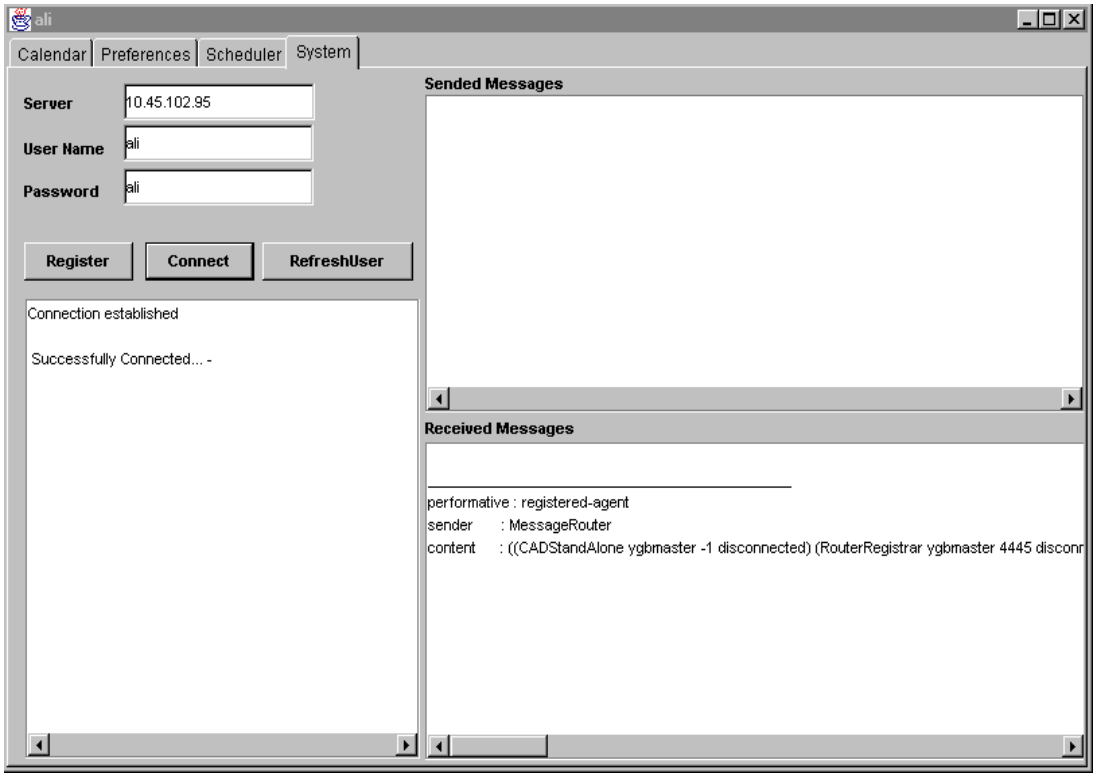
Sistemin çalışabilmesi için JATLite yönlendiricisinin de bir bilgisayarda çalıştırılması gerekmektedir. JATLite yönlendiricisi çalıştıktan sonra sistemde çalışmaya başlayan etmenler(planlama etmeni ve yer etmeni tipinde) JATLite yönlendiricisine register olmalıdırlar. Daha önceden register olmuşlar ise connect olabilirler.

4.6.1. Planlama Etmeni Programı

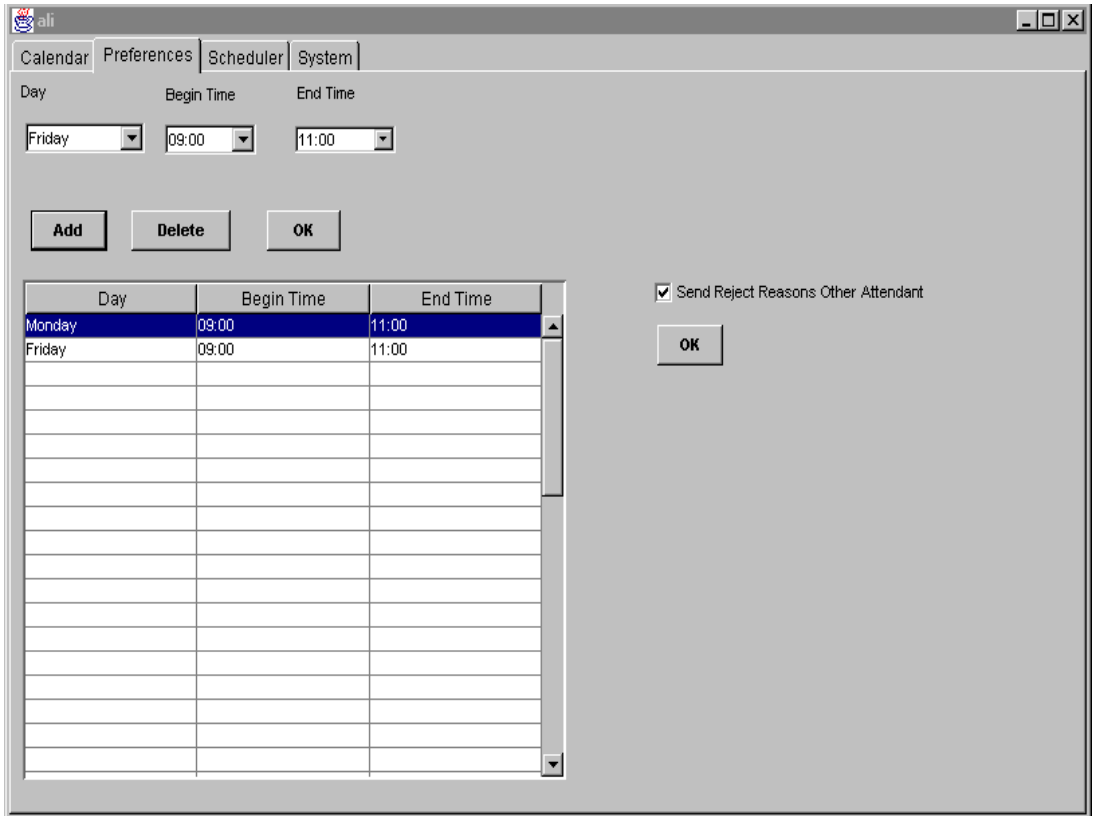
Planlama etmeni ekranı, sisteme kayıt olmak ve etmene gelen giden mesajların izlenmesine için kullanılan “System” menüsü, toplantı isteğinde bulunmak için kullanılan “Scheduler” menüsü, öncelik bilgilerini girmek izelemek için kullanılan “Preference” menüsü ve kullanıcının takvim bilgisinin izlenmesi için kullanılan "Calendar” menüsü olmak üzere 4 alt menüye sahiptir.

Program çalışmaya başladıktan sonra şekil 4.8’de görülen “System” menüsü aracılığı ile server’a JATLite yönlendiricisinin çalıştığı bilgisayarın IP’si , kullanıcı adı ve şifre girilerek sisteme kayıt olunur. Ayrıca bu menü aracılığı ile sistem mesajları, diğer etmenlere ve JATLite yönlendiricisine gönderilen ve alınan mesajlar izlenebilir.

Kullanıcının öncelik bilgileri; toplantı istenmeyen günler ve diğer etmenlere red sebeplerinin gönderilip gönderilmemesi bilgileri şekil 4.9’daki “Preferences” menüsü kullanarak izlenebilir ve girilebilir.

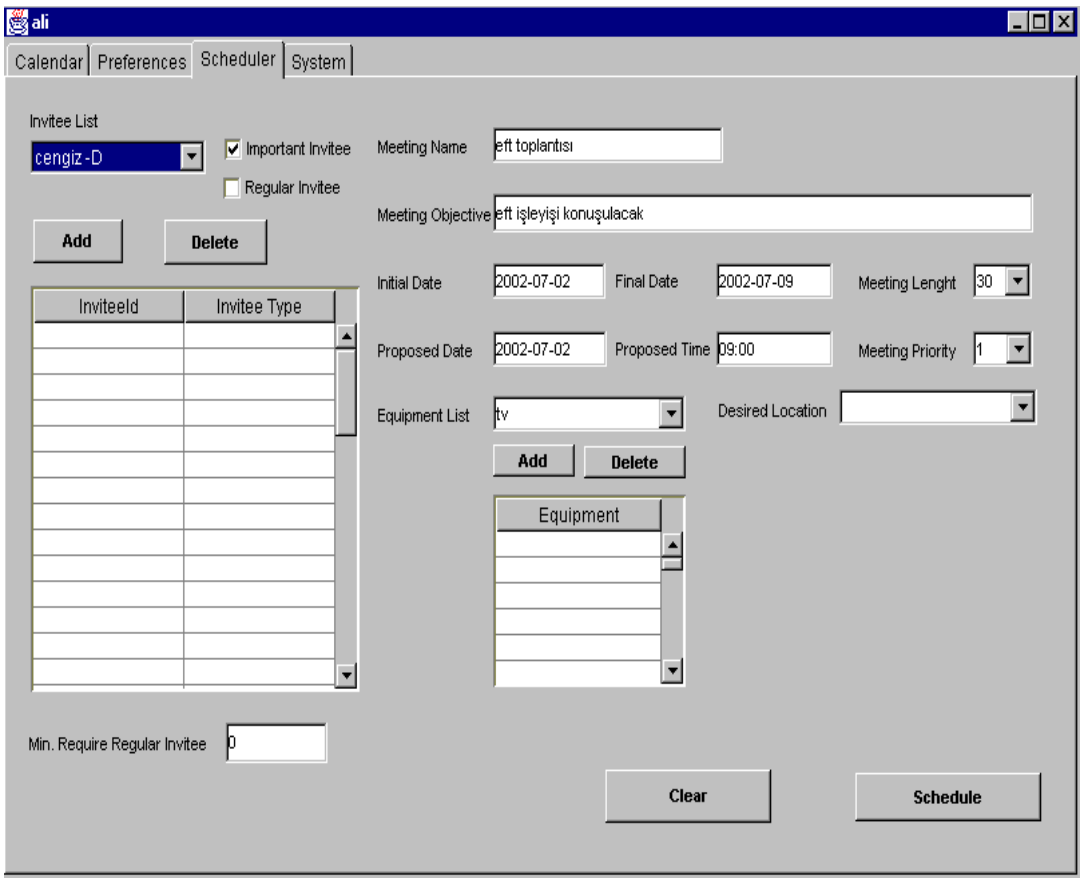


Şekil 4.8 System ekranı menüsü



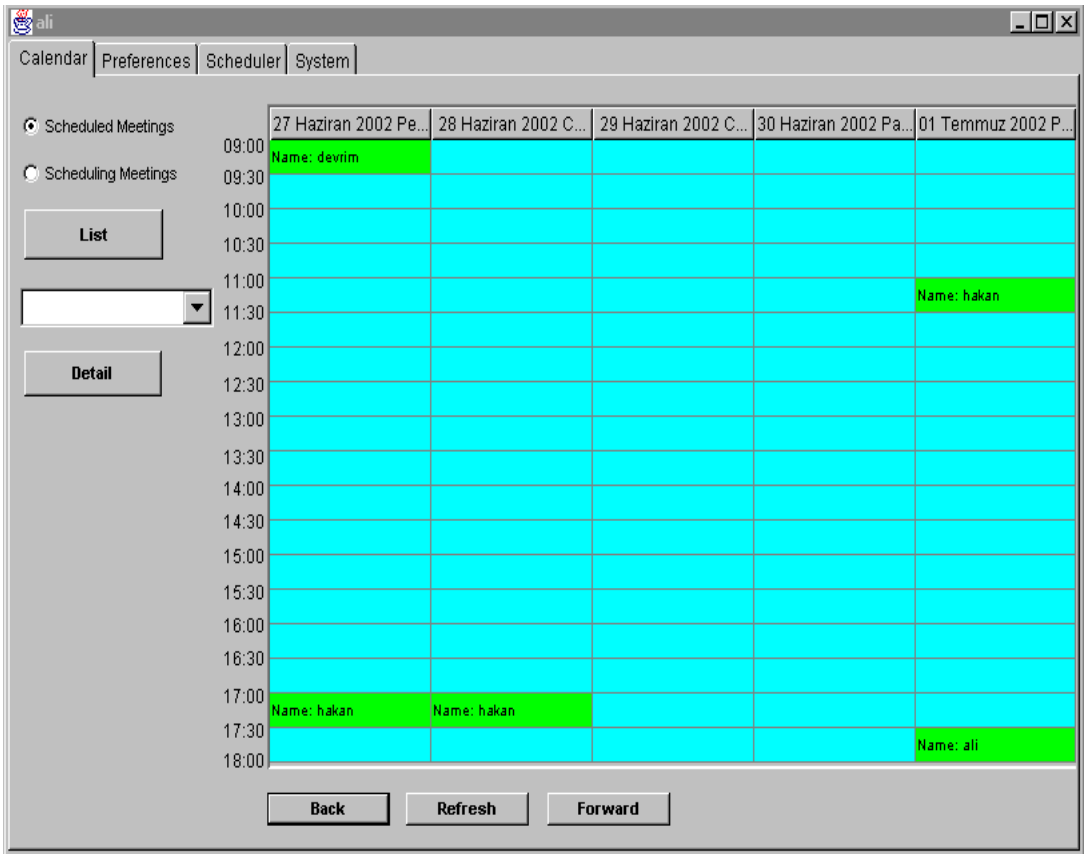
Şekil 4.9 Preference ekranı menüsü

Toplantı isteğinde bulunmak için şekil 4.10'da görülen "Scheduler" menüsü kullanılır. Sistemde kayıtlı olan kişilerin listesinden toplantı katılımcıları seçilir ve bu kişiler önemli ya da sıradan katılımcı olarak listeye eklenir, yine sisteme kayıtlı olan toplantı yeri listesinde eğer isteniyorsa toplantı yeri seçilebilir. Eğer toplantı için ihtiyaç duyuluyorsa gerekli olan araçlar listeden seçilip eklenir. Toplantı başlama ve bitiş tarihi, toplantı süresi, toplantı için önerilen tarih ve diğer bilgiler bu menü aracılığı ile girilir. Daha sonra "Schedule" tuşu kullanılarak toplantı isteği yapılır.

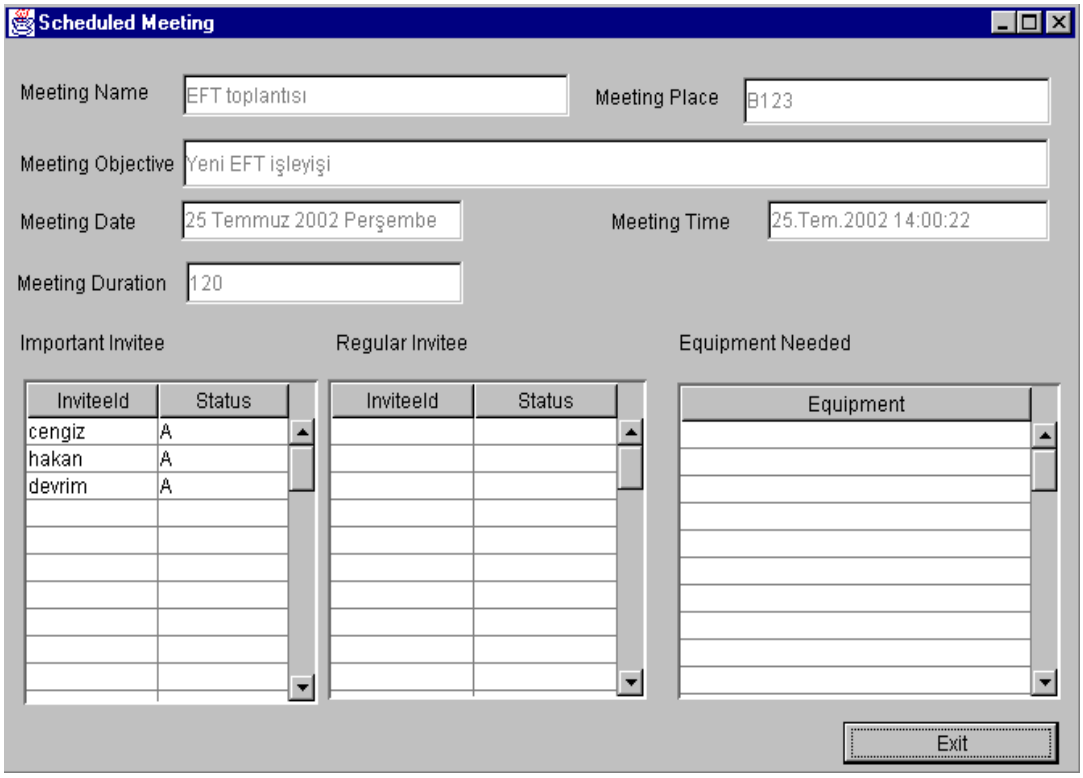


Şekil 4.10 Scheduler ekranı menüsü

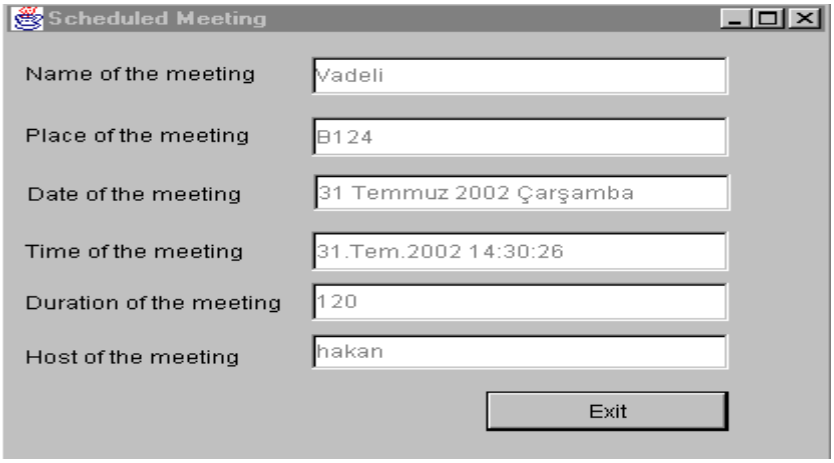
Gerçekleşen toplantılar şekil 4.11'de görülen "Calendar" menüsü aracılığı ile izlenebilir. Bu menüde bügünden itibaren 5 gün ve her gün 30 dakikalık zaman dilimlerine bölünmüş olarak gösterilir. Dolu olan zaman dilimleri yeşil renkte gösterilir. "Forward" tuşu ile daha sonraki 5 gün gösterilir, "Backward" tuşu ile önceki 5 gün gösterilir, "Refresh" tuşu ile ekran yenilenir. Herhangi bir toplantı üzerine basıldığı zaman ise toplantı ile ilgili detaylı bilgi şekil 4.12'deki ve şekil 4.13'deki detay ekranı görüntülenir. İzleme yapılan etmenin düzenlediği toplantının detayı izleniyor ise şekil 4.12'de görülen ekran gösterilir, başka bir etmenin düzenlediği toplantının detayı izleniyor ise şekil 4.13'de görülen ekran gösterilir.



Şekil 4.11 Calendar ekranı menüsü



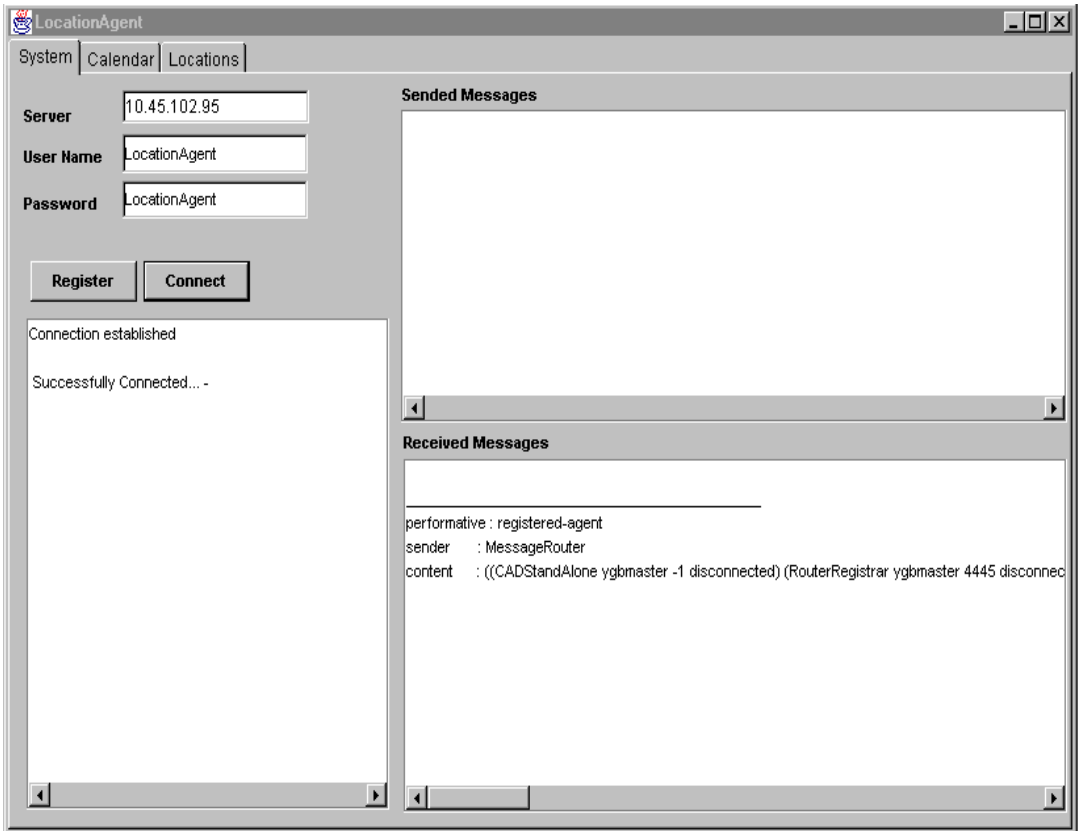
Şekil 4.12 Kendi toplantısı ekranı



Şekil 4.13 Davetli olunan toplantı ekranı

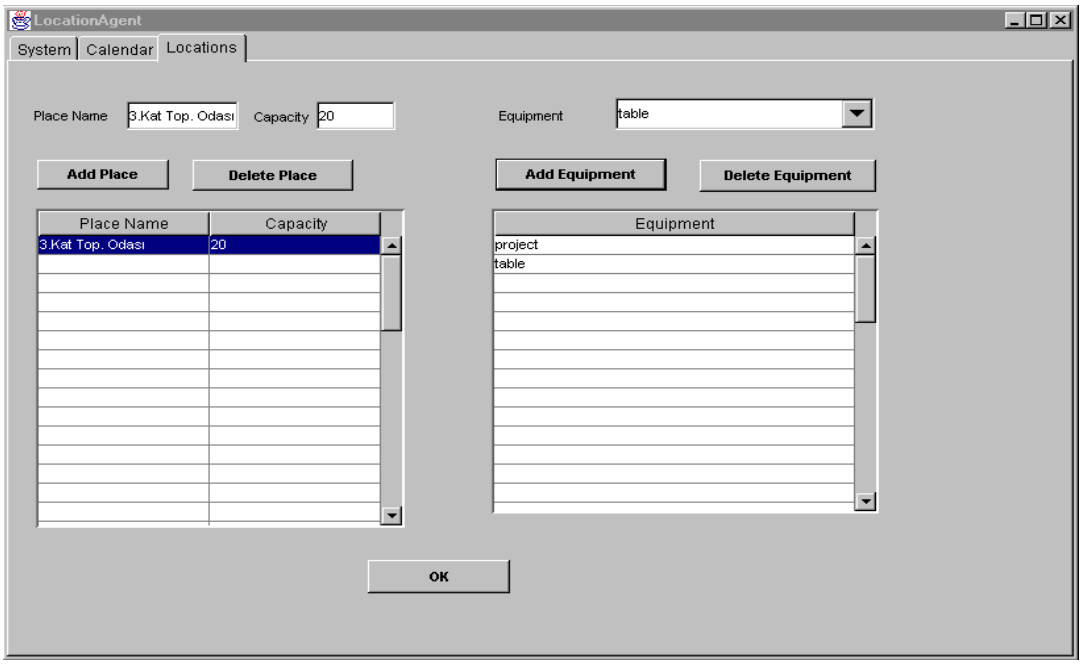
4.6.2. Yer Etmeni Programı

Yer etmeni sistemde bir tane bulunur ve herhangi bir bilgisayarda çalışabilir. Yer etmeni sisteme kayıt olmak ve etmene gelen giden mesajların izlenmesine yarayan "System" menüsü, sistemde kayıtlı olan toplantı yeri bilgilerini girmek için kullanılan "Locations" menüsü ve kullanıcının takvim bilgisinin izlenmesi için kullanılan "Calendar" menüsü olmak üzere 3 alt menüye sahiptir.

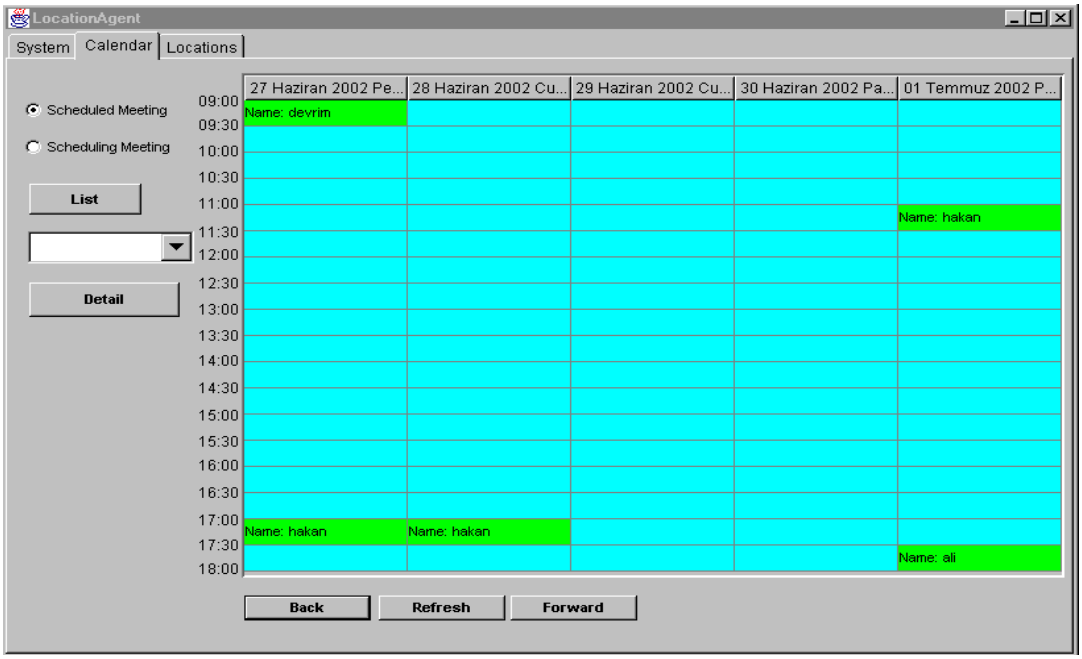


Şekil 4.14 System ekranı menüsü

Program çalışmaya başladıktan sonra şekil 4.14. görülen “System” menüsü aracılığı ile server’a JATLite yönlendiricisinin çalıştığı bilgisayarın IP’si , kullanıcı adı ve şifre girilerek sisteme kayıt olunur. Ayrıca bu menü aracılığı ile sistem mesajları, diğer etmenlere ve JATLite yönlendiricisine gönderilen ve alınan mesajlar izlenebilir.



Şekil 4.15 Locations ekranı menüsü



Şekil 4.16 Calendar ekranı menüsü

Sisteme toplantı yeri bilgisi tanımlamak ve var olan toplantı yeri bilgilerini izlemek için şekil 4.15'de görülen "Locations" menüsü kullanılabilir. Bu menü'den sisteme yeni toplantı yeri, bu toplantı yerinin kapasitesi ve sahip olduğu araçlar girilebilir.

Gerçekleşen toplantılar şekil 4.16'da görülen "Calendar" menüsü aracılığı ile izlenebilir. Bu menüde bugünden itibaren 5 gün ve her gün 30 dakikalık zaman dilimlerine bölünmüş olarak gösterilir. Dolu olan zaman dilimleri yeşil renkte gösterilir. "Forward" tuşu ile daha sonraki 5 gün gösterilir, "Backward" tuşu ile önceki 5 gün gösterilir, "Refresh" tuşu ile ekran yenilenir. Herhangi bir toplantı üzerine basıldığı zaman ise toplantı ile ilgili detaylı bilgi şekil 4.13'deki detay ekranı görüntülenir.

4.7. Deneysel Sonuçlar

Daha önceki bölümlerde anlatılan toplantı düzenleme algoritmasını gerçekleyen program yazıldıktan sonra, bir takım deneysel işlemler yapmak üzere bir sistem kuruldu. Bu sisteme değişik sayıda kullanıcı tanımlandı. Toplantı süresi(uzunluğu), toplantının düzenlenmesi gereken işgünü aralığı, toplantı yeri sayısı ve katılımcı sayısı(aynı zamanda toplantı istek sayısıdır) değiştirilerek toplantı düzenleme işlemleri yapıldı. Daha önceki bölümlerde de anlatıldığı üzere toplantı düzenleme işlemi sırasında kullanıcı takviminde bulunan ilk boş zaman dilimi toplantı için seçildiğinden, ard arda sürekli toplantı düzenlendiği zaman kullanıcı takvimi başlangıçtan itibaren doldu.

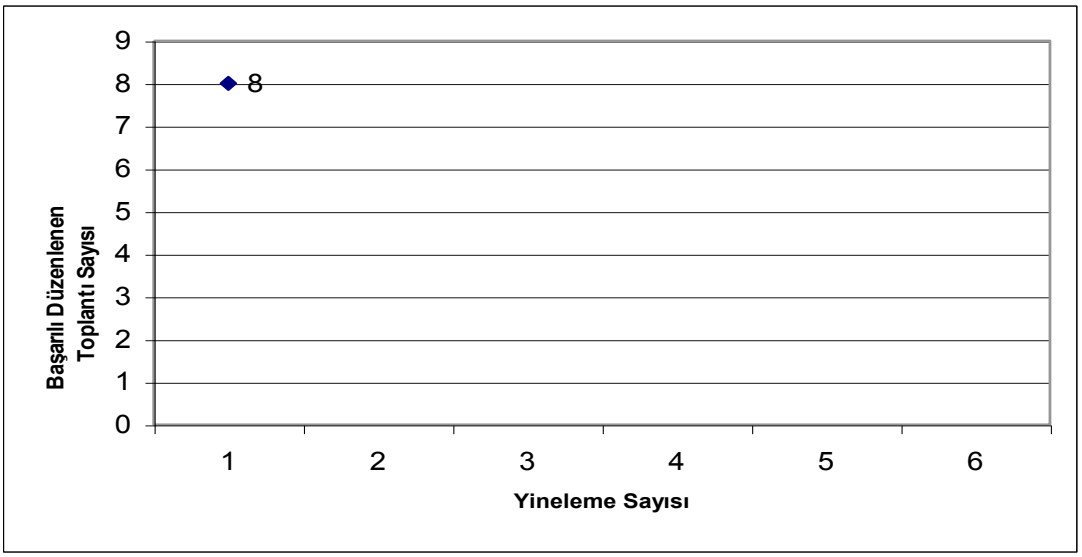
Denemeler sırasında ortaya çıkan ilginç sonuç ise; belli sayıdaki kullanıcı grubu ile bunlardan herbirinin aynı toplantıya katıldığı ve her biri aynı anda bir birlerinden toplantı isteğinde bulunduğu işlem sırasında gerçekleşti. Örnek olarak 8 kişinin 8'ininde katıldığı 8 ayrı toplantı isteğini aynı anda birbirlerinden istedikleri zaman ortaya çıkan durumu verebiliriz. Bu durumu denemek üzere, geçici olarak, programa toplantı düzenleme işlemi başarısızlıkla sonuçlandığı zaman başa dönerek işlemi bir kez daha denemesi için ekleme yapıldı. Bu yeniden deneme işlemi "yineleme" olarak adlandırıldı.

Aşağıda bu şekilde düzenlenmiş program ile yapılan denemelerden elde edilen sonuçlardan oluşturulan değişik grafikler bulunmaktadır.

İlk denemeler sisteme 8 kişi tanımlanarak yapıldı. Toplantı düzenleyen sisteme 8 kişi ve bir adet yer etmeni tanımlandı. Bu 8 kişi aynı zaman dilimi içersinde ve aynı anda karşılıklı, bir birlerinin katılacağı 8 kişilik, 8 adet toplantı düzenlemeyi denediler. Toplantı süresi olarak 30 dakika, toplantı yeri sayısı olarak ise 1 alındı. Kullanıcılar

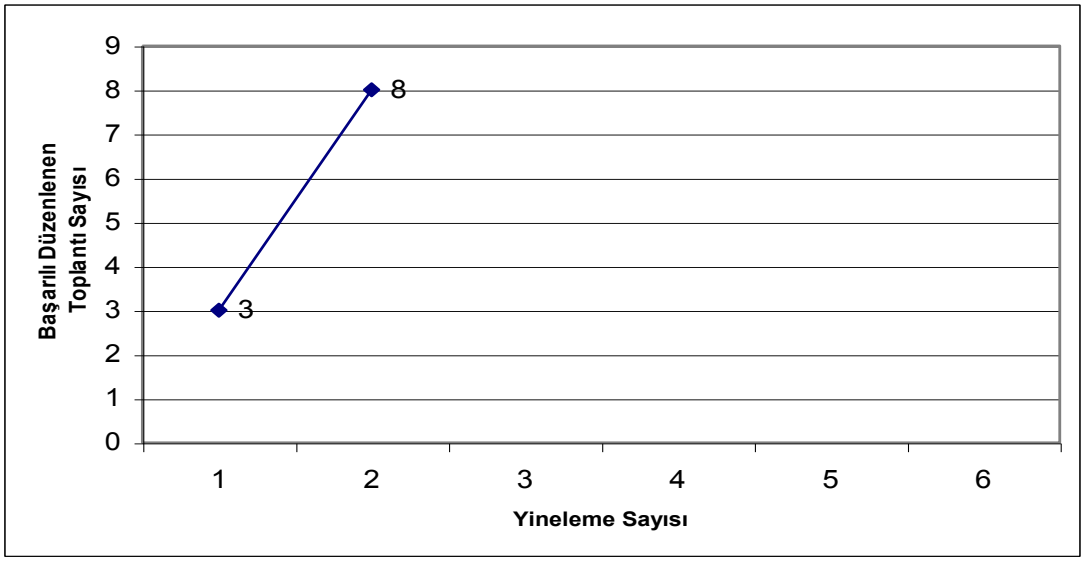
toplantı isteğini birbirlerine aynı anda yaptılar. Toplantının düzenleneceği zaman dilimi olarak ise sırasıyla 5 işgünü, 3 işgünü , 2 işgünü ve 1 işgünü kullanılmıştır.

Şekil 4.17'de yukarıda anlatılan denemenin 5 işgünü için yapılmasından elde edilen sonuç görülmektedir. Şekilde de görüldüğü gibi 5 işgünü içersinden 8 toplantı da yinelemeye gerek duymadan başarılı bir şekilde düzenlenebilmiştir.

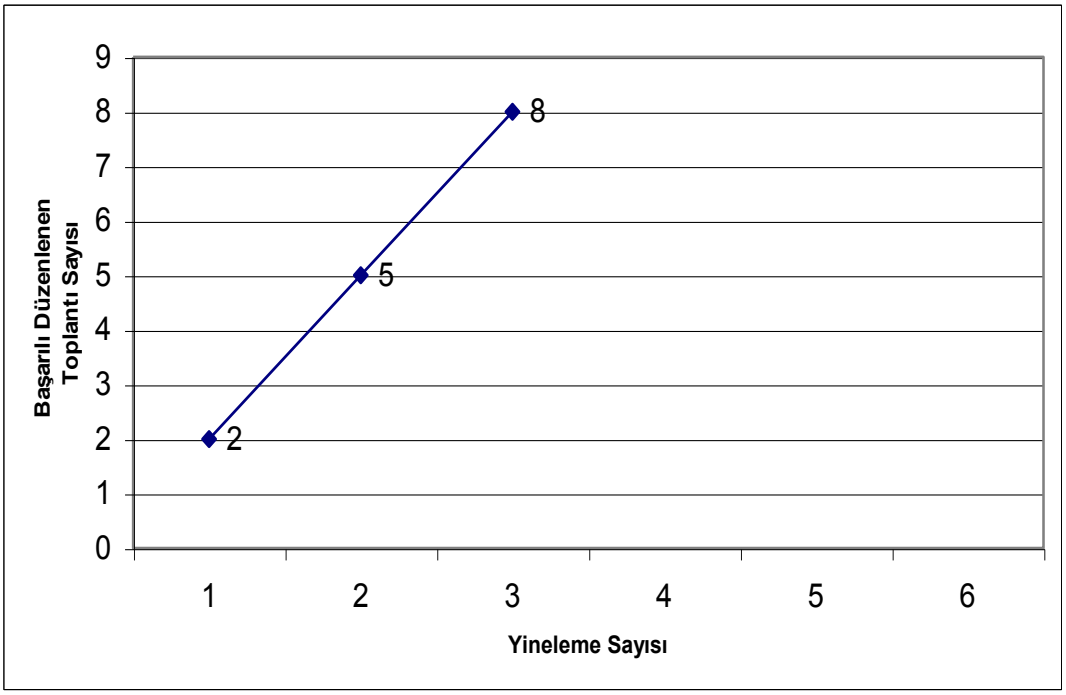


Şekil 4.17 8 kişi 5 işgünü sonuçları

5 işgünü yerine 3 işgünü kullanıldığında ise şekil 4.18'de görüldüğü üzere 1 yinelemede ancak 3 toplantı düzenlenebilmekte 8 toplantının 8'i ancak 2 yinelemeden itibaren düzenlenebilmektedir.

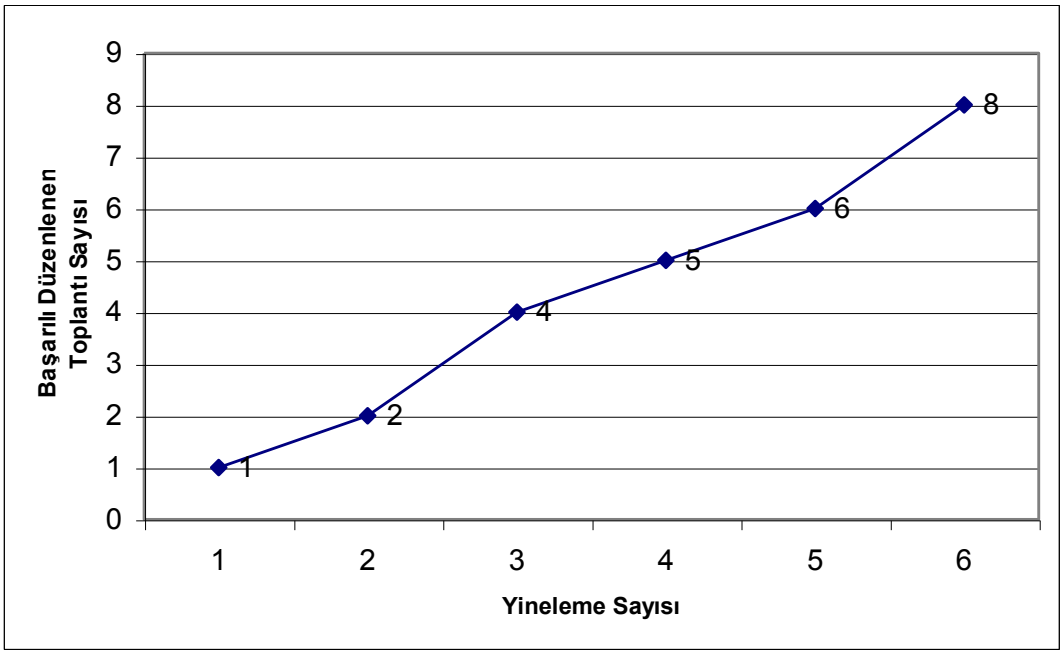


Şekil 4.18 8 kişi 3 işgünü sonuçları



4.19 8 kişi 2 işgünü sonuçları

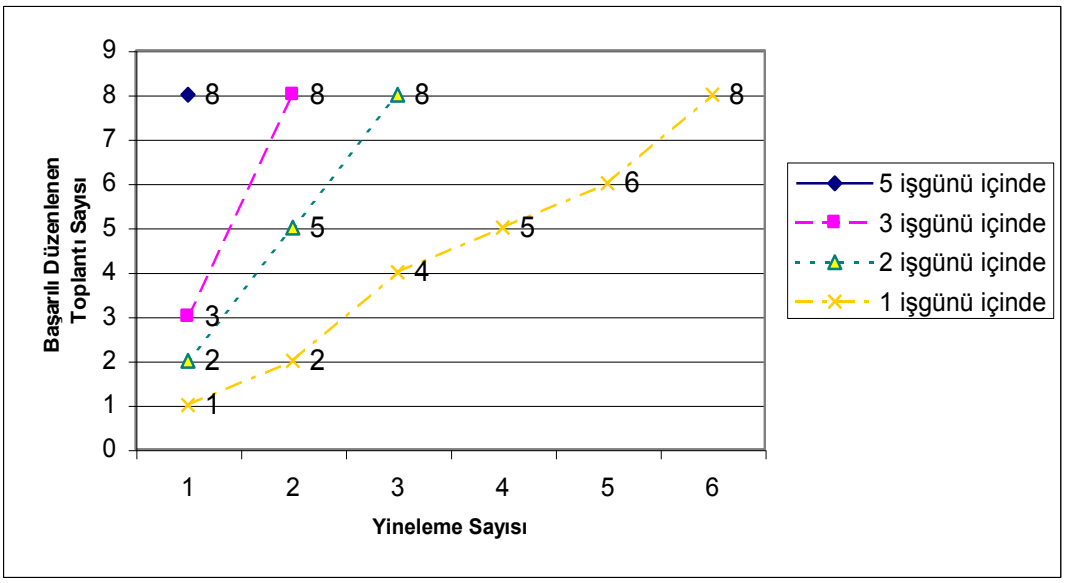
İşgünü sayısı daha da azaltılıp 2'ye indirildiğinde ise şekil 4.19'da görüldüğü gibi 1 yinelemede ancak 2 toplantı başarılı bir şekilde düzenlenebilmektedir. Yineleme sayısı arttıkça düzenlenen toplantı sayısı da artmaktadır. En sonunda 3 yinelemede 8 toplantının 8'i de başarılı bir şekilde düzenlenebilmektedir.



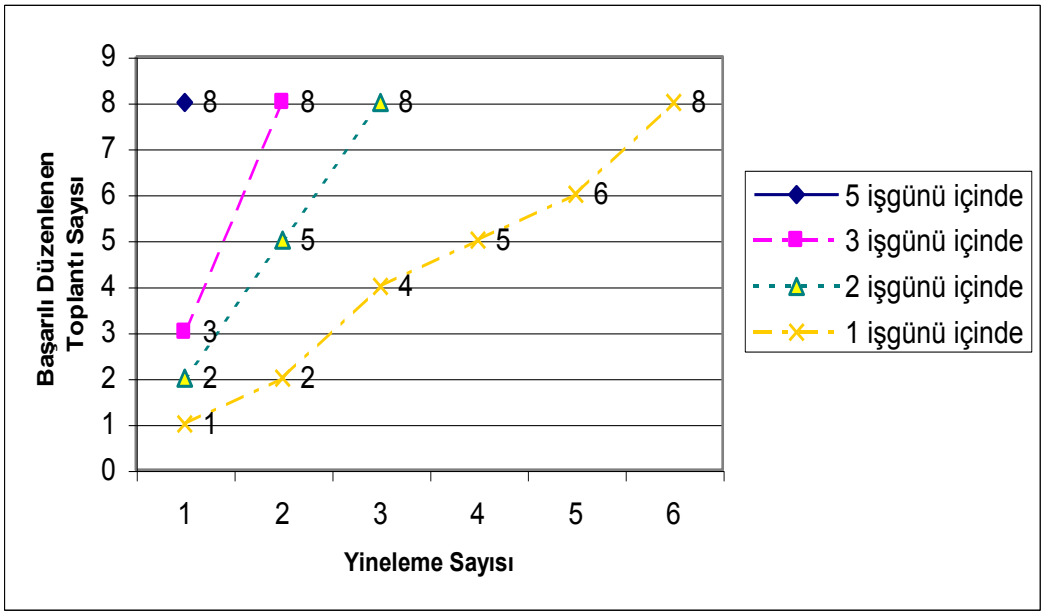
4.20 8 kişi 1 işgünü sonuçları

1 işgünü için bu denemeleri yaptığımızda ise yukarıdaki şekil 4.20 elde edilir. 1 yineleme ile ancak bir toplantı düzenlenebilirken 6 yineleme ile ancak 8 toplantı düzenlenebilmektedir.

Yukarıdaki grafikleri birleştirdiğimizde ise aşağıdaki şekil 4.21'deki gibi bir grafik elde etmekteyiz. Bu grafiğe bakar isek toplantının düzenlenmesi gereken işgünü aralığı küçüldükçe bütün toplantıları başarılı şekilde düzenlemek için gerekli olan yineleme sayısı artmaktadır.

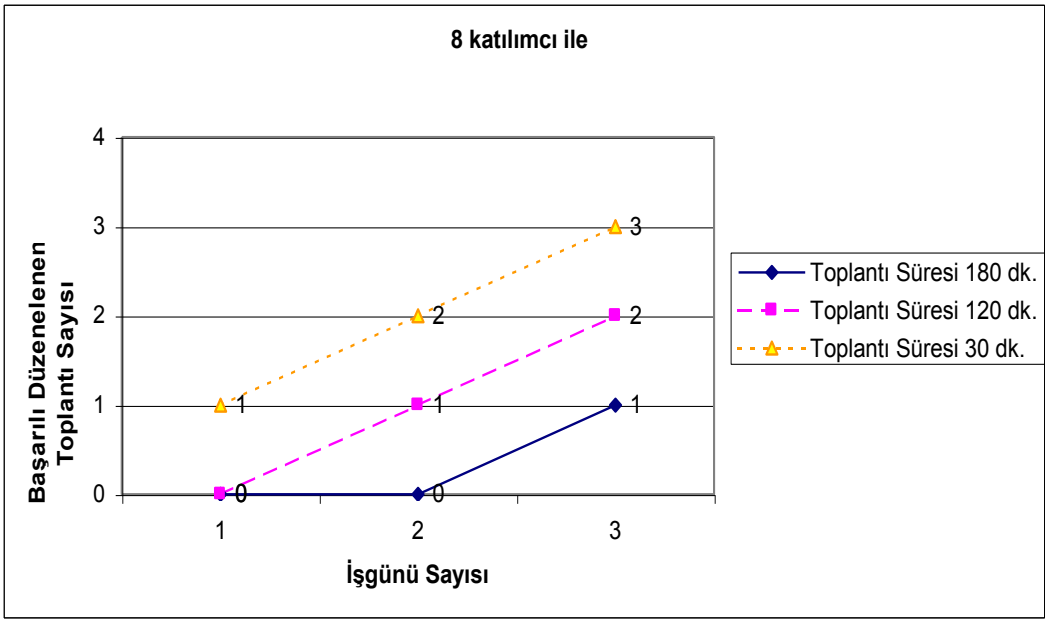


4.21. 8 kişi 5-3-2-1 işgünü sonuçları

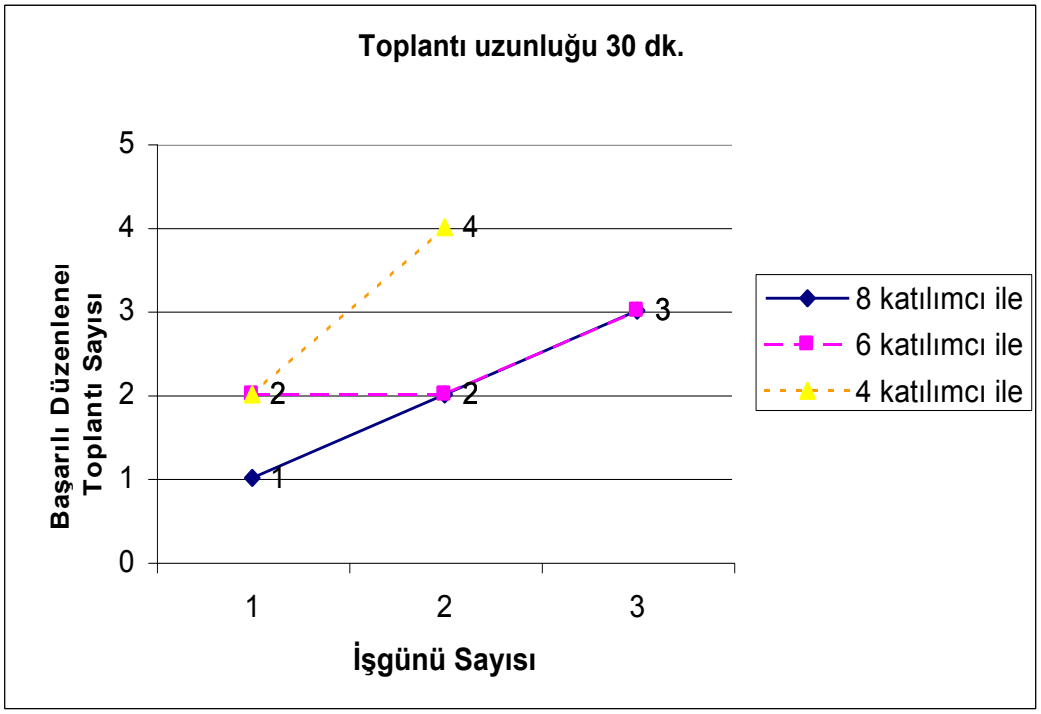


4.22 8 kişi 5-3-2-1 işgünü 2 toplantı yeri sonuçları

Toplantı yeri sayısı değişirse : Yukarıdaki grafikler bir toplantı yeri ile elde edilmişti. Toplantı yeri sayısını 1 den 2'ye çıkardığımızda ise yukarıdaki şekil 4.22'deki gibi bir grafik elde ediyoruz. Şekil 4.21 ile 4.22'yi karşılaştırdığımızda ise pek bir fark göremiyoruz. Bu ise yer sayısının artırılmasının karşılıklı aynı anda düzenlenen toplantıların başarı şansını pek artırmadığını göstermektedir.



4.23 Toplantı uzunluğu farklılaştırması sonuçları



4.24 Toplantı katılımcısı farklılaştırması

Toplantı süresi değiştirilirse : Daha önceki denemelerde toplantı süresi 30 dakika tutulmuştu. Toplantı süresini değiştirdiğimiz zaman elde ettiğimiz değerler şekil 4.23'de gösterilmiştir. Toplantı uzunlukları 120 ve 180 dakikaya çıkarılmış ve denemeler 1,2 ve 3 işgünü için, 8 katılımcı ve 8 toplantı ile yapılmış olup yineleme sayısı 1 olarak tutulmuştur. Belli bir güne belli bir toplantı süresinden kaç adet toplantı sığacağı bellidir. Örneğin bir gün içersinde 180 dakikalık toplantılardan iki tane yapılabilir. Bu sebepten dolayı aynı işgünü sayısında istenen toplantıların süresi uzadıkça başarılı düzenlenen toplantı sayısı azalmaktadır.

Toplantı istek sayısı değiştirilirse : Bir sonraki deneme olarak ise sistemdeki kişi sayısı değişken haline getirilmiş ve her kişinin bir toplantı isteğinde bulunduğu düşünülürse, aynı anda toplantı düzenleme istek sayısı değiştirilmiştir. Daha önceki denemelerde kişi sayısı 8 idi ve aynı anda istenen toplantı sayısı da 8 idi. Toplantı istek sayısını değiştirdiğimiz zaman elde ettiğimiz değerler şekil 4.24'deki grafikte gösterilmiştir. Kişi sayısı 6 ve 4 olarak belirlenmiş, toplantı uzunluğu 30 dakika ve denemeler 1,2 ve 3 gün için yapılmış olup yineleme sayısı 1 olarak tutulmuştur. Toplantı süresine benzer olarak aynı sayıdaki işgününde sabit uzunlukta düzenlenebilecek toplantı adedi bellidir. Bu sebepten dolayı toplantı isteğini artırdığımız zaman başarılı düzenlenen toplantı sayısı azalmaktadır.

Deneysel Sonuçların Tartışılması: Program ile yapılan denemeler sonucunda algoritmanın işleyişi ile ilgili değişik sonuçlar elde edilmiştir. Aynı zaman dilimi için sistemdeki etmenlerin karşılıklı olarak katıldıkları ve bir birinden aynı anda toplantı istedikleri zaman algoritmanın işleyişinde bazı farklılıklar ortaya çıkmaktadır. Katılımcı sayısına(aynı anda istenen toplantı adedi), toplantı uzunluğuna ve toplantı istenen tarih aralığının uzunluğuna bağlı olarak elde edilen sonuç değişmektedir. Toplantı istenen tarih aralığı ve katılımcı sayısı sabit tutularak toplantı uzunluğu artıkça başarılı bir şekilde düzenlenen toplantı sayısı azalmaktadır. Toplantı istenen tarih aralığı ve toplantı sayısı sabit tutularak katılımcı sayısı artıkça başarılı bir şekilde düzenlenen toplantı sayısı azalmaktadır. Katılımcı sayısı ve toplantı sayısı sabit tutularak toplantı istenen tarih aralığı artıkça başarılı bir şekilde düzenlenen toplantı sayısı artmaktadır. Bazı durumlarda program kullanılmadan kağıt üzerinde el ile toplantılar yerleştirilebilmekte fakat program ile çok daha az sayıda toplantı başarılı bir şekilde yerleştirilebilmektedir. Bu ise programda toplantılar için zaman aralıklarının geçici olarak bloklanmasından kaynaklanmaktadır. Blok işleminin kaldırılması ise tutarsızlığa yol açmaktadır.

Denemeler sırasında kullandığımız yineleme yöntemi ile daha fazla toplantı başarılı bir şekilde düzenlenmiştir. Ancak yineleme işlemi gerçek anlamda bir çözüm

değildir. Çünkü yinelemenin gerçek anlamda çözüm olması için aynı anda, karşılıklı olarak ve aynı tarih aralığında etmenlerin karşılık toplantı istediğinin bilinmesi gerekir. Bu ise ancak merkezi bir sistem ile yani sistemin bütün bilgisinin tek bir etmen tarafından bilinmesi ile mümkün olmaktadır. Farklı bir yöntem olarak, sistemde aynı anda kaç toplantı düzenlendiği bilinmediği için toplantı süresi ve toplantı istenen işgünü aralığına bağlı olarak yineleme adedi belirlenebilir. Örneğin toplantı istenen işgünü aralığı 1 ve toplantı süresi 30 dk. ise en az 3 yineleme yap ya da toplantı istenen işgünü aralığı 3 ve toplantı üresi 10 dk. ise en az 3 yineleme yap gibi. Ancak belli bir fonksiyon ile başarısız düzenleme işlemi sonrasında yenileme yapmak sisteme yeni bir yük getirecektir. Ayrıca yukarıda belirttiğimiz durum aynı anda, karşılıklı olarak ve aynı tarih aralığında etmenlerin karşılıklı toplantı istemesi gerçek hayatta çok çok az bir olasılıkla ve belki de hiç bir zaman olmayacak bir durumdur.

4.8. Diğer Dağıtık Toplantı Düzenleme Sistemleri

Çoklu etmen sistemleri kullanarak toplantı düzenleme üzerine değişik çalışmalar yapılmıştır. Burada bunlardan iki tanesine değinilecektir.

4.8.1. Otomatik Dağıtık Toplantı Düzenleyicisi

Sandip Sen tarafından "An automated distributed meeting scheduler"[8] adlı makalesinde bireyler adına toplantı düzenleme işlemini yürüten otomatik toplantı düzenleyicilerden oluşan bir sistem önerilmiştir. Toplantı isteğinde bulunan konak etmen, toplantıya katılanlar ise davetli etmen olarak adlandırılmıştır. Her etmenin öncelikleri ve takvim yöneticisi bulunmaktadır. Etmenler arası iletişim elektronik mail aracılığı ile yürütülmektedir. Sistemde toplantı aşağıda belirtilen parametreler ile tanımlanmıştır;

- Katılımcılar listesi,
- Toplantı uzunluğu,
- Toplantı önceliği,
- Toplantı başlama zamanı,
- Düzenleme bitiş tarihi,
- Diğer sınırlamalar

Bu bilgileri kullanarak toplantı düzenleme işleminde kullanılan toplantı düzenleme protokolü ise aşağıdaki şekildedir;

- Toplantı düzenleme işlemini başlatan konak kendisine uygun zaman dilimleri bulur ve bulduğu zaman dilimlerini katılımcılara bildirir. Eğer uygun bir zaman dilimi bulamamış ise toplantı düzenleme işlemi sonlanır.

- Davetliler kendilerine gönderilen zaman dilimlerini değerlendirirler ve zaman dilimleri kendilerine uyuyorsa kabul, uymuyorsa karşı önerilerini gönderirler. Karşı öneri bulamaz iseler red gönderiler.

- Yanıtları alan konak kendi değerlendirmesini yapar. Ortak bir uzlaşma oluşmuş ise bütün katılımcılara mesaj gönderir. Eğer uzlaşma olmamış ise yeni öneriler bulup onları katılımcılara gönderir. Aldığı ve kabul etmediği bütün önerilere ayrıca red gönderir.

- Yeni önerileri alan davetliler yukarıdaki gibi yanıtlarlar. Düzenlendi mesajı alan katılımcılar bu zaman diliminin hala boş mu diye bakarlar boş takvimlerini işaretler dolu ise red mesajı gönderirler.

Ayrıca sistemde daha önceden planlanan toplantının iptal edilmesi için de bir mekanizma bulunmaktadır.

Bu örnekte, bu tezde sunulan sistemden farklı olarak toplantı yeri bilgisi dikkate alınmamıştır. Oysa biliyoruz ki günlük hayattaki toplantı düzenleme işlemi sırasında en fazla sorun çıkaran konulardan biri de toplantı yeri ile ilgili sorunlar ve sıkıntılardır. Bu ise sistemin günlük hayatın bir modeli olmaktan uzaklaştırmaktadır. Ayrıca bu örnekte toplantı katılımcıların hepsi aynı tiptedir, önemli-sıradan katılımcı ayrımı yoktur. Örnek çalışmada karşı öneri ve iptal mekanizması vardır.

4.8.2. Akıllı Toplantı Düzenleyicisi

Jin Yan, Lin Shu, Situ Qihua, Wu Xudong tarafından önerilen "An Intelligent Meeting Scheduler"[9] bireyler adına toplantı düzenleme işlemini yürüten otomatik toplantı düzenleyicilerden oluşan bir sistemdir. Toplantı isteğinde bulunan konak etmen, toplantıya katılanlar ise davetli etmen olarak adlandırılmıştır. Her etmenin öncelikleri, takvim yöneticisi ve toplantı yerleri bilgisine erişmek için aracı olan yer yöneticisi bulunmaktadır. Sistemde toplantı katılımcıları, toplantıya katılması zorunlu olan önemli katılımcı ve belli bir oranda toplantıya katılması gereken sıradan katılımcı olarak ikiye ayrılmıştır. Etmenler arası iletişim elektronik mail aracılığı ile yürütülmektedir. Sistemde toplantı aşağıda belirtilen parametreler ile tanımlanmıştır;

- Önemli katılımcı listesi,
- Sıradan katılımcı listesi,
- Toplantı konusu,
- Toplantı uzunluğu,
- Tarih aralığı,
- Toplantı sırasında kullanılan araçlar.

Toplantı düzenleme işlemi sırasında kullanılan pazarlık protokolü aşağıdaki şekildedir;

- Toplantı isteğinde bulunulan konak kendi önceliklerini ve takvimini kontrol ederek yer yöneticisi ile irtibat kurar. Toplantının özelliklerine göre yer yöneticisinden uygun zaman dilimlerini alır. Uygun zaman dilimlerinden sadece 3 tanesini alır.
- Bu üç zaman dilimini VIP olarak adlandırılan önemli katılımcılara önerir. Bu önerileri alan önemli katılımcılar öneriler ile ilgili kendi yanıtlarını geriye bildirirler.
- Yanıtları alan konak yanıtları değerlendirir. Ortak bir zaman dilimi için kabul varsa bu zaman dilimini sıradan katılımcılara önerir. Ortak bir zaman dilimi için kabul yoksa konak işleme baştan başlar.
- Öneriyi alan sıradan katılımcılar yanıtlarını konağa bildirirler.
- Sıradan katılımcılardan yanıtları alan konak, yanıtları değerlendirir. Daha önceden belirtilen kabul sayısından az kabul aldıysa, konak baştan başlar. Yeterince kabul aldıysa en erken olan zaman aralığını alır ve bunu sıradan ve önemli katılımcılara onay için gönderir.
- Onayı alan katılımcılar takvimlerinde o zaman dilimi hala boşsa o zaman dilimine takvimlerinde ayırırlar, boş değilse konağa red gönderirler.

Ayrıca sistemde karmaşıklığı çözmek için katılımcılara önceliklerini yok saymaları yönünde teklif götürelebilinir, toplantı gerçekleşme tarih aralığı genişletilebilir ve son olarak sıradan katılımcıların kendi yerlerine toplantıya başka birisini göndermeleri istenebilir.

Bu örnek çalışmada öneri ve pazarlıklar sadece üç adet öneri üzerinden işlemektedir. Bu ilk bakışta sistemin yükünü azaltır gibi görünse de sistemdeki pazarlığa davetli etmenlerin katılımını sınırlamaktadır. Öneri karşı öneri mekanizmasının gücü yeterince bu örnekte kullanılmamıştır. Ayrıca bu örnekte toplantı yerlerini temsil eden bir etmen olmadığı için toplantı yerleri pazarlıkta aktif olarak yer almaz oysa burda sunulan tez çalışmasında toplantı yerlerini temsil eden bir etmen sistemde bulunduğu için bu etmen pazarlığa aktif olarak katılır. Karmaşıklık çözümü için örnek çalışmada bir yapı tanımlanmış olması sistemin avantajlı yönlerinden birisidir.

4.9. Sistem Özellikleri

Dağıtık Çoklu Etmen Toplantı Planlama Sisteminin temel karakteristiklikleri aşağıdaki gibidir.

İşlemin Gerçek Karakterine Uygun : Dağıtık Çoklu Etmen Toplantı Sistemi toplantı planlama işleminin günlük hayattaki işleyişine ve genel yapısına uygundur. Gerçek yaşamda toplantı planlama işlemi dağıtık bir işlemdir ki sunulan sistemde bu özeliğe sahiptir. Gerçek toplantı planlama işleminin gerçek bir simülasyonudur.

Özerk Olma : Özerk etmen ortamından gelen istekler karşısında özerk bir şekilde karşılık veren etmendir. Sistemdeki etmenler sistemdeki diğer etmenlerden gelen isteklere kendisinde var olan bilgilerden yola çıkarak, kendi başına karar vererek, kullanıcının müdahalesi olmadan yani özerk bir şekilde kullanıcı adına yanıt vermektedir.

Akıllı Olma : Sistemdeki etmenler kendilerinde bulunan bilgileri, çalışma sırasında edindikleri bilgileri daha önce ya da çalışma sırasında tanımlanmış kuralları kullanarak belli kararları kendi başlarına verebilme yeteneği sahiptirler. Bu ise doğal olarak etmenlerin akıllı olması anlamına gelmektedir.

Öğrenme : Sistemdeki planlama etmenleri toplantı planlama işlemi sırasında katılımcı etmenlerden aldıkları ret sebepleri içeren mesajları kullanarak katılımcı etmenlerin takvimleri hakkında bilgi edinir. Bu bilgiler ile katılımcı etmenlerin takvimlerini oluşturmaya çalışır. Yani öğrenme işlemi gerçekleştirir.

İletişim : Dağıtık toplantı planlama işleminin doğası gereği sistem içindeki etmenler bir biriyle sürekli iletişim halindedir. Bu iletişimin merkezinde ise JATLite'ın sunduğu etmen mesaj yönlendiricisi bulunmaktadır.

Birlikte Çalışma : Dağıtık toplantı düzenleme işlemi ancak etmenlerin birlikte çalışarak sonuca ulaştırabilecekleri bir problemdir. Bu yüzden sistem içindeki

etmenler birlikte çalışarak toplantı planlama işlemini başarılı ya da başarısız bir şekilde çözüme ulaştırırlar.

Esneklik : Sistemde iki tür esneklik vardır. Bunlardan birincisi bize JATLite etmen mesaj yönlendiricisinin sağladığı etmenlerin sisteme herhangi bir yerden(IP'den) bağlanabilmeleri ve mesajlarının kaybolmamış olmasıdır. Yani etmenler için sabitlenmiş bir yer yoktur ve etmen sisteme girdiğinde ona gelen bütün mesajlar yönlendirici tarafından kendisine gönderilir. Diğer esnekli ise sistemin esnekliğidir. Yani sistem yeni etmenlerin girişine müsaittir ve yeni etmenler sisteme eklenebilir.

SONUÇLAR VE TARTIŞMA

Bu çalışma dağıtık çoklu etmen sistemi kullanılarak geliştirilmiş toplantı planlama sisteminin yapısını ve gerçekleştirilmesini ayrıntılarını içermektedir. Sistem gerçek yaşam problemlerinin yazılım uygulamaları ile çözülmesine güzel bir örnektir. Sistem sahip olduğu pek çok özelliğin yanı sıra kullanıcıya da pek çok kolaylıklar sağlamaktadır.

Sistem daha önce de belirttiğimiz gibi çoklu etmen sistemlerinin özelliklerini taşımaktadır. Sistem içersindeki etmenler akıllı bir etmendirler. Bu ise onların kendi başlarına karar verebilme ve öğrenme özelliklerine sahip oldukları anlamına gelmektedir.

Sistemde kullanılan bazı teknikler mevcut toplantı planlama algoritmasının işlem zamanı kısaltmaktadır. Bunlardan en önemlilerinden birisi toplantı planlama işlemi sırasında uygulanan katılımcıların karşı öneri getirmesi tekniğidir ki bu sayede toplantı planlama işlemi daha kolay sonuca gidebilmektedir. Toplantı planlama işleminin zamanını kısaltmaktadır. Bir diğer teknik ise davetlilerin öneri ret sebeplerini düzenleyiciye bildirmeleridir. Bu sayede düzenleyici davetli etmenin takvimlerinin bir kısmını öğrenmiş olur ve kendi aramalarında bu bilgileri dikkate alır ki bu bilgi sayesinde sonuca ulaşmak yine hızlı olmaktadır. Bu aynı zamanda etmen öğrenme yeteneğini göstermektedir.

Sistemde toplantı yerleri bilgilerinin de dikkate alınması, onu gerçek hayata yaklaştırmaktadır. Çünkü gerçek yaşamdaki toplantı planlama işlemlerindeki en büyük sorunlardan birisi planlanması düşünülen toplantı için boş toplantı yerleri bulunamamasıdır. Sistemdeki toplantı yerleri tek bir etmen tarafından temsil edilmektedir. Sistemde toplantı yerleri bilgisi merkezi tutulduğu ve tek bir etmen tarafından yönetildiği için yeni toplantı yerleri bilgileri eklemek çok kolaydır.

Sisteme bu çalışmada dikkate alınmayan, toplantı planlama işlemi karmaşıklığa düştüğü zaman yani toplantı başarılı olarak planlanamadığı zaman toplantı düzenleyen kişinin önceliğini veya toplantı önceliğini dikkate alarak bazı toplantıları iptal edip toplantı planlama işleminin karmaşıklık çözme adımına geçme şeklinde

ekleme yapılarak geliştirilebilir. Ancak bu çalışmaya böyle bir yapının eklenmesinin nedeni, böyle bir eklemenin işlemin dağıtık yapısını bozacağı ve mesajlaşmayı çok artıracağı tespitidir. Çünkü uygun toplantı zamanı arama işlemi tekrar eden ve dağıtık bir arama olduğu için iptal edilecek toplantı bilgileri düzenleyici etmene gönderilebilir ve bu bilgiler içerisinden ortak zaman aranabilir. Bu durumda toplantı planlama işlemi merkezi bir işleme dönüşür. Bir diğer kaygı ise bu şekilde bir iptal mekanizması ile pek çok toplantının iptali ve sistemde bol miktarda iptal edilmiş ve yeniden planlanmaya başlanan toplantılar olabilir. Bu şekildeki işlemler ise mesaj trafiğini ister istemez arttıracaktır.

Sistemde aynı anda, karşılıklı olarak ve aynı tarih aralığında etmenlerin karşılıklı toplantı istemesi durumda başarılı düzenlenen toplantı sayısı deneysel sonuçlar bölümünde anlatıldığı gibi azalabilmektedir.

Sistem kendisinden ve JATLite etmen mesaj yönlendiricisinden kaynaklı olmak üzere çok esnek bir yapıdadır. JATLite'ın etmen mesaj yönlendiricisinin sabit bir yere bağlı olmayan etmen bağlanma, mesaj gönderme yapısı sistemdeki etmenlerin farklı bilgisayarlarda, farklı zamanlarda çalışmalarına ve sisteme girer girmez işlemlerine kaldıkları yerden devam edebilmelerini olanak sağlamaktadır. Ayrıca sistemin yapısı gereği sisteme yeni etmenler kolayca eklenebilir, sisteme bağlı olmayan etmenler sisteme bağlandıkları andan itibaren işlemlerini kaldıkları yerden yürütebilirler.

KAYNAK

[1] **Michael Wooldridge,** 1999. Multiagent Systems A Modern Approach to Distributed Artificial Intelligence pp.41. Gerhard Weiss.

[2] **S. Franklin and A. Graesser,** 1996. "Is it an Agent, or just a program? A taxonomy for Autonous Agents" Proc. Third International Workshop on Agent Theories, Architecures, and Languages, Springer-Verlag.

[3] Knowledge Querying and Manipulation Language – KQML standard, Kasım 2001. http://www.cs.umbc.edu/kqml

[4] JATLite (Java Agent Template, Lite), Kasım 2001. http://cdr.stanford.edu/ProcessLink/Papers/Jatl.html

[5] Extensions to the KQML standard - http://cdr.stanford.edu/ProcessLink/kqml-proposed.html

[6] **Eugene C. Freuder, Marius Minca and Richard J. Wallace**, 2001. Privacy / Efficiency Tradeoffs in Distributed Meeting Scheduling by Constraint-Based Agents. International Joint Conference on Artificial Intelligence(IJCAI2001) Distributed Constraint Reasoning Workshop.

[7] **Sandip Sen and Edmund H. Durfee,** 1995. Unsupervised Surrogate Agents and Search Bias Change in Flexible Distributed Scheduling. Proceeding, First International Conference on Multi-Agent Systems p.336-343 San Franscisco, CA June 1995.

[8] **Sandip Sen,** 1996. An automated distributed meeting scheduler. IEEE Expert 12(4) p.41-45.

[9] **Jin Yan, Lin Shu, Situ Qihua, Wu Xudong,** 2001. An Intelligent Meeting Scheduler, http://web.cs.ualberta.ca/~qihua/scheduler/report.html

ÖZGEÇMİŞ

Ali Durmuş 1976 yılında Balıkesir'in Bigadiç ilçesinde doğmuştur. İlkokulu Yumrukluçetmi Köyü İlkokulunda, ortaokul ve liseyi ise Bigadiç Cumhuriyet Lisesinde okumuştur. Orta Doğu Teknik Ünversitesi Bilgisayar Mühendisliği bölümüne 1993 yılında girmiştir. Bu bölümden 1998 yılında mezun olmuştur. Halen İstanbul Teknik Üniversitesi Bilgisayar Mühendisliği Bölümünde master öğrenimine devam etmektedir. Ali Durmuş özel sektörde çalışmaktadır.