itüdergisi/***d***
mühendislik
Cilt:7, Sayı:2, 83-94
Nisan 2008

Güvenli bir hareketli etmen sistemi

Suat UĞURLU*, Nadia ERDOĞAN

İTÜ Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Programı, 34469, Ayazağa, İstanbul

Özet

Hareketli etmen mimarisi istemci-sunucu çalışma modeline karşın dağıtık işlemeye farklı bir yaklaşım sunmaktadır. Kodun hareketliliğine dayanan hareketli etmen sistemlerinde, güvenlik düşünülmesi gereken önemli bir unsurdur; çünkü artık durağan bir yazılım parçası değil, kodunu ve verisini uzak düğümlere taşıyabilen yazılımlar, yani etmenler söz konusudur. Bu türden hareketli yazılımların hem kodunun hem de verisinin izlenme veya değiştirilmesi gibi yeni güvenlik risklerinin ortaya çıkması kaçınılmazdır. Birbirleri ile haberleşebilen etmenlerin mesajlaşmaları sırasında da aynı tehlikeler söz konusudur. Daha da önemlisi, güvenlik riskleri ile karşı karşıya olan sadece etmenler değildir, etmenleri üzerlerinde çalıştıran düğümler de aynı ölçüde risk altındadırlar. Bu yazıda, hareketli etmen sistemlerindeki mevcut güvenlik tehlikelerini ortadan kaldıracak yeni bir mimarinin tasarım ve gerçeklenme ayrıntıları incelenmiştir. Geliştirilen güvenli etmen sistemi, hem etmenlerin güvenlik gereksinimlerine yanıt vermek, hem de kolay kullanımlı ve esnek bir çalışma ortamı sunmak üzere tasarlanmış ve gerçeklenmiştir. Gelişmiş güvenlik özellikleri yanında sistem, değişen güvenlik ihtiyaçlarına kolay ve anında uyum sağlayabilmek için güvenlik politikalarını kullanmaktadır. Güvenlik politikaları, değişen güvenlik ihtiyaçlarına, hızlı ve etmenin yeniden programlanmasını gerektirmeden cevap verebilmeyi sağlar. Sistem ayrıca sadece etmenlerin değil, etmenlere çalışma ortamı sunan düğümlerin güvenliği için de gerekli mekanizmaları sunmaktadır. Geliştirilmiş olan hareketli etmen sistemi, güçlü bir mesajlaşma altyapısı sunmasının yanında, izlenilebilirlik, yönetilebilirlik ve süreklilik için de esnek arayüzler barındırmaktadır. Sistem katmanlı bir mimariye sahiptir ve geliştirilmeye açıktır.

Anahtar Kelimeler: *Etmen, hareketli etmen sistemleri, hareketli etmen sistemlerinde güvenlik.*

*Yazışmaların yapılacağı yazar: Suat UĞURLU. suat@suatugurlu.com; Tel: (212) 427 35 62.
Bu makale, birinci yazar tarafından İTÜ Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Programında tamamlanmış olan "Güvenli bir hareketli etmen sisteminin tasarımı ve gerçeklenmesi " adlı doktora tezinden hazırlanmıştır. Makale metni 12.02.2007 tarihinde dergiye ulaşmış, 08.03.2007 tarihinde basım kararı alınmıştır. Makale ile ilgili tartışmalar 01.02.2009 tarihine kadar dergiye gönderilmelidir.

A secure mobile agent system

Extended abstract

According to the accepted definition, an agent is a small application with some special features. Being autonomous, capable of adapting itself to its environment, communicating with other agents for coordination or cooperation, intelligence, ability to clone itself and ability to make decisions are the features that can distinguish an agent from ordinary software. Even though mobility, ability to migrate from one host to another host, is not a required feature, agents with this ability have advantages especially in terms of distributed data processing. A mobile agent is not restricted to the node where it is running and can migrate to anywhere on the network of its own accord. While moving from one host to another, not only the agent's executable code is transferred, but also data that the agent has collected or constructed are transferred as well. Thus, the agent can preserve its state even when it is mobile. The execution framework necessary for a mobile agent is provided by a mobile agent system. This framework simply provides the basic agent related tasks and functions such as agent creation, activation, migration, communication, cloning and destruction. The competence and power of a mobile agent system depends on the flexibility of these functions.

Even though using mobile agent technologies provides potential benefits to applications, an agent's ability to move introduces significant security risks. Mobile agents are under security threats during their life times. Since the code is mobile, it can be stolen or altered by a third party. The same danger is present for the messages agents send to each other and for the data that determines the agent's state. Furthermore, not only the agents but also hosts are also under many security risks in mobile agent systems.

Several mobile agent systems have been proposed and developed up to now. They all have their software agent specific features. Although most of them have enough features for mobile agents to communicate with each other and migrate to remote hosts, agent security related tasks are not available in most of them. Some provide limited security for agents, but do not provide any features to protect hosts. Most of these mobile agent systems leave the security to agent programmer or to the traditional net work security solutions which may be very difficult and inefficient to implement or integrate. The mobile agent system is expected to include all necessary security mechanisms for both agents and computers hosting mobile agents.

The scope of this paper is the design and implementation of a new, secure, flexible, highly available and fast mobile agent system (SECMAP). The architecture of the system is especially designed for security purposes, and requirements not only for agent security but also for host security are also provided. Besides ensuring security of both agents and hosts, SECMAP also presents a very flexible agent programming interface. Naturally, these features play an important role on the usability and popularity of the system. SECMAP also presents a policy based management framework to protect system-level resources and agents against unauthorized access, as well. The policy architecture allows for dynamic manipulation of policy content, which results in an adaptive and flexible framework that eliminates the reprogramming of the agents on changing conditions. Logging and monitoring of the basic agent activities are also possible.

Availability is very important for the collaborating agents. For this reason, a mobile agent system should be up and running even only one host in the system is active. When necessary the system should be able to transfer the duties of a dead host to another one in the system. SECMAP includes very powerful algorithms to ensure the availability of the overall system. It accomplishes this by assigning special working modes to different agent servers in the system. Another important feature is that the system and agents can be managed and monitored from a browser in the network. All agents present in the system can be monitored from a single window. Any module of the system can also be managed by a browser from remote hosts.

SECMAP is worth being used not only for the security features it presents for agents and hosts, but also for its flexibility and powerful agent programming interface. The system has a layered architecture and is open to be improved with more powerful features.

Keywords: *Agents, Mobile agent systems, Security in mobile agent systems.*

Giriş

Bilgisayar sistemlerinin gelişim sürecine baktığımızda, yazılım metodolojilerinin gün geçtikçe merkezi çalışma modellerinden, dağıtık çalışma modellerine doğru evrim geçirmekte olduğunu görmekteyiz. Merkezi bir bilgisayardan istekte bulunan terminaller bu evrim sürecinin başlangıcında yer alırken, farklı bir bilgisayardan kodun yüklenmesiyle yerel olarak çalışmaya başlayan java appletlerini sürecin son aşamalarında görmekteyiz. Yakın zamanda yeni bir yazılım felsefesi olarak bir adım daha ileriye gidilmiş, daha geniş ölçekli platformlar oluşturabilmek ve bütünüyle dağıtık işlemeye elverişli bir çalışma modeli meydana getirmek için hareketli etmen sistemleri oluşturulmuş ve yazılım nesnelerinin hareketliliği sağlanmıştır.

Basitçe tanımlamak gerekirse, bir etmen, otonom, kendi başına karar verebilen, akıllı, çevresine uyum sağlayabilen, haberleşebilen ve işbirliği içinde çalışabilen yazılım parçaları olarak adlandırılmaktadır. Hareketli bir etmen ise bu özelliklerin yanısıra, bulunduğu ortamla sınırlı olmayıp, istediği an ağ üzerindeki başka bir düğüm üzerine kendini taşıyabilme yeteneğine sahiptir. Etmen bu taşınma sırasında sadece çalıştırılabilir kodunu değil, o ana kadar edindiği verileri de taşıyarak durumunu korur.

Hareketli bir etmen sistemi, hareketli etmenler için gerekli olan çalışma ortamını sunar. Öyle ki, etmenin, yaratılma, aktif olma, göç etme, çoğalma, mesajlaşma gibi ihtiyaçları bu alt yapı aracılığı ile gerçeklenir.

Kodun hareketliliğine dayanan hareketli etmen sistemlerinde, güvenlik düşünülmesi gereken önemli bir unsurdur. Hareketli olan kod, yaşam süresince birçok risk altındadır (Borselius, 2002). Hareketli olan etmenin kodu ve verisi taşınma sırasında çalınabilir veya değiştirilebilir. Etmenin çalıştırılması geciktirilebilir, yanlış bilgilendirme ile amaçlanan sonuca erişmesi engellenebilir. Taşındığı ortamda yer alan diğer etmen ya da üçüncü parti yazılımların müdahalesi ile karşı karşıya kalabilir.

Benzer şekilde, düğümler de kötü niyetli etmenler tarafından kötüye kullanılabilirler. Düğüm kaynakları aşırı kullanım sonucu tüketilebilir, gizli verisi çalınabilir ya da düğüm kullanıcısı yanlış yönlendirilerek kandırılabilir.

Çok kısaca açıklanan ve hem etmenlerin hem de düğümlerin karşı karşıya oldukları güvenlik tehlikeleri bunlarla da sınırlı değildir. Bu nedenle, hareketli etmenlerin kullanımı ancak güvenlik ihtiyaçlarına çözümler üretebilen hareketli bir etmen sisteminin oluşturulması ile mümkün olabilir. Ayrıca değişen çevresel koşullara kısa zamanda uyum sağlayabilmek için de gerekli mekanizmalar hareketli etmen sistemleri içinde barındırılmalıdır. Diğer önemli bir ihtiyaç ise, etmen programcısına mümkün olduğunca esnek bir programlama ve yönetim arayüzünün sunuluyor olmasıdır. Hareketli etmen sistemlerinin güvenlik gereksinimleri, sadece, güvenlik problemlerinin tasarım aşamasında dikkate alınarak çözümlerin oluşturulduğu bir sistem tarafından karşılanabilir. Mevcut hareketli etmen sistemleri incelendiğinde, genellikle güvenlik gereksinimlerini göz önüne alan tasarımlar yapılmadığı, bu gereksinimlerin, sonradan, yama yöntemleri ile karşılanmaya çalışıldığı görülür. Ayrıca sınırlı olan etmen güvenlik çözümleri düğümler için ise hiç düşünülmemiştir. Çözüm; güvenlik problemine çözümlerin tasarım anında düşünülerek gerçeklendiği, hem etmenleri hem de düğümleri tehditlere karşı koruyacak gerekli fonksiyonları içeren, gerektiğinde dinamik yapısı nedeniyle çevresel değişimlere kolay uyum sağlayabilecek, kolay yönetilebilen, izlenebilen, hızlı yeni bir hareketli etmen sisteminin gerçeklenmesidir. Bu çalışmanın amacı, hareketli etmen sistemlerindeki güvenlik gereksinimlerine cevap verebilen, esnek ve güçlü bir programlama ve yönetim arayüzü sunan yeni bir hareketli etmen sisteminin tasarlanması ve gerçeklenmesidir. Ana hedef, mevcut sistemlerin cevap veremediği güvenlik tehditleri için yeni çözümler üretmek, yan hedefler ise oluşturulan sistemin mümkün olduğu kadar kullanılabilirliği arttıracak özelliklere sahip olmasıdır.

Güvenli etmen sistemi

JAVA dilinde geliştirilmiş yeni bir hareketli etmen sistemli olan Güvenli Etmen Sistemi (GES), mevcut etmen sistemlerinden, özellikle güvenlik

mekanizmaları göz önüne alınarak tasarlanmış güvenli mimarisiyle ayrılır. GES, tüm temel etmen fonksiyonlarını desteklemesinin yanında, etmen programcısına esnek geliştirme fonksiyonları ve çalışma ortamı da sunar.

GES, etmen programcısına olay tabanlı kod geliştirmesi için gerekli ortamı sunar. Programcıya etmenin yaşam döngüsü içinde yer alabileceği her durum için farklı bir metod yazma olanağı sunar. Bu model programcıya esnek bir programlama arayüzü sağlar.

Sistem, etmenlerin, mesajlar aracılığıyla, birbirleriyle senkron ve asenkron olarak haberleşmelerine olanak sağlar. Mesajlaşma alt yapısı ölçeklenebilir olma özelliğine sahiptir. Ayrıca sistem, etmenlere birbirlerine o an üzerlerinde çalıştıkları düğüm adresinden bağımsız olarak mesaj gönderme olanağı sunmaktadır. Konuma göre saydam olan mesajlaşma altyapısı sayesinde, bir etmen diğer bir etmene mesaj gönderebilmek için onun ağ üzerinde nerede olduğu bilgisine ihtiyaç duymaz.

Mimarinin en önemli özelliklerinden biri de güvenlik politikalarını etkin bir şekilde kullanarak, hem etmeni hem de düğümleri koruyan fonksiyonlar içermesidir. Güvenlik politikaları dinamik olarak tanımlanabilir, değiştirilebilir ve uygulanabilir. Bu yaklaşım, etmenin değişen çevresel koşullara uyum sağlamasını kolaylaştırdığı gibi, etmen programcısına da, etmen kodunda değişiklik yapmadan etmen davranışlarını farklılaştırma olanağı sağlar.

GES, etmen durum ve kod bilgisinin korunması içinde gerekli araçlara sahiptir. Etmen kod ve durumu bir etmenin yaşam döngüsü boyunca, üzerinde çalıştığı düğümün ana belleği hariç, hiç bir şekilde açık biçimde tutulmaz. Şifreleme metodları ile etmen kodu, durumu ve politikası, yetkisiz değişimleri de sezme özelliğine sahip fonksiyonlar ile sürekli koruma altındadır.

Sistemdeki tüm uzak haberleşmeler de ulaşım katmanı seviyesinde SSL protokolü ile gerçeklenir. Bu özellik, ağı dinleyen yetkisiz etmen veya kişilerin etmenlere ait özel bilgileri çalmasını ya da sistemdeki etmenleri izlemesini engeller. Tüm mimari bileşenleri, ağ üzerindeki herhangi bir düğüm üzerinde çalışan bir tarayıcı (web browser) ile izlenebilir ve yönetilebilir. Bu özellik yönetilebilirlik açısından sisteme büyük esneklik sağlamaktadır. Etmen yaratılması, etmenin aktif-pasif duruma geçirilmesi yada göç etmesi veya mesajlaşması gibi aktivitelerin saklanması ve izlenmesi, etmenin düğüm üzerinde çalışırken diğer etmenlere karşı korunması gibi daha bir çok özellik mimaride mevcuttur.

GES sunucusu

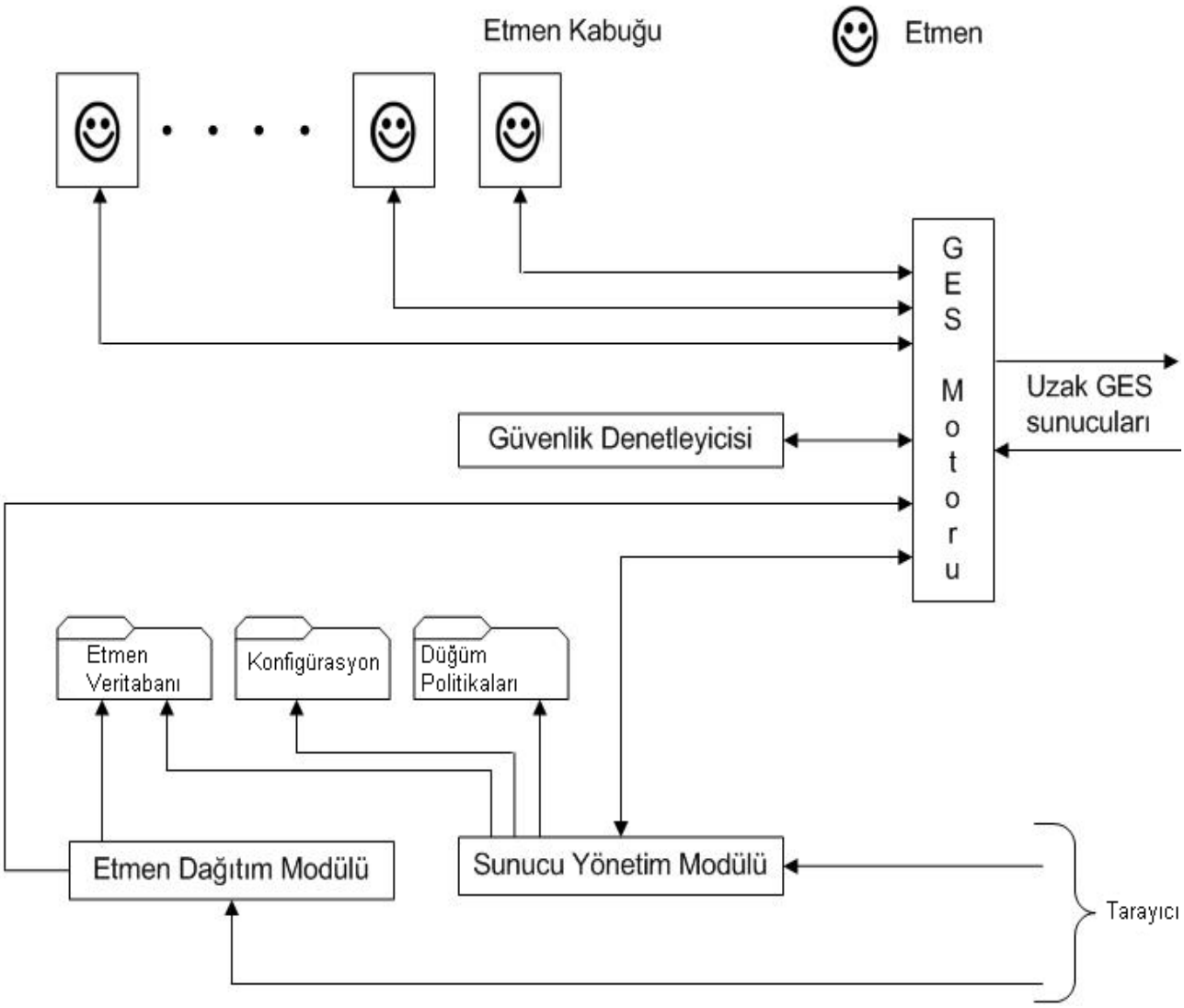
GES mimarisinin ana bileşeni, etmen kabul edecek her düğüm üzerinde kurulu ve çalışır durumda olması gereken GES sunucusudur. JAVA dilinde geliştirilmiş olan GES sunucusu, etmen yaratma, çalıştırma, etmen haberleşmesi ve etmen göçü gibi temel fonksiyonları sağlar. Oldukça karmaşık sayılabilecek fonksiyonları olmasına rağmen modüler bir yapıya sahip olması nedeniyle yeni fonksiyonlar eklenmeye olanak veren bir özelliğe sahiptir. Şekil 1 bir GES sunucunun ana bileşenlerini göstermektedir.

Güvenlik Denetleyicisi; yürütülen etmen aktivitelerinin güvenlik politikalarına uygun olup olmadığını kontrol eder. GES motoru fonksiyonları bu türden aktiviteleri yerine getirmeden önce isteği Güvenlik Denetleyicisine iletir ve aldığı yanıta göre sadece izin verilen aktivitelerin yürütülmesini sağlar.

Etmen Kabuğu; etmeni bir kabuk gibi sarmalayıp, diğer etmenlerden ya da yazılımlardan kaynaklanabilecek doğrudan erişimleri engelleyerek etmeni korur. Etmen Kabuğu etmenin dış dünya ile iletişimi için bir arayüz sunar. Ayrıca etmen giriş çıkış mesaj kuyruklarının yönetiminden de sorumludur.

Etmen Veritabanı; etmenlerin saklanan son durumları, kodu ve politikasının tutulduğu şifreli disk alanıdır.

Konfigürasyon; etmen sunucusuna ait konfigürasyon bilgisinin tutulduğu alandır. Örneğin sunucunun hangi düzende çalıştığı, ortaklık ilişkisi içinde olduğu diğer sunucuların adresleri,



Şekil 1. GES sunucu ana bileşenleri

dinleyen tcp port numaraları gibi ayarlar burada bulunur.

Düğüm Politikaları; düğüm yöneticisinin düğümler için oluşturduğu politikaların tutulduğu disk alanıdır.

Sunucu Yönetim Modülü; sunucunun uzaktan web tarayıcı ile yönetilmesi için gerekli olan fonksiyonları barındırır. Etmen programcısı veya düğüm yöneticisi, etmenleri ve sunucuyu uzaktan bu bileşen aracılığı ile izleyebilmekte ve yönetebilmektedir.

Etmen Dağıtım Modülü; uzaktan etmen programcısının yazdığı etmeni tarayıcı aracılığı ile sunucu üzerine dağıtma görevini yerine getirir.

GES sunucu mimarisi katmanlı bir yapıya sahiptir. Bu özellik, katmanların ayrı ayrı programlanarak yeteneklerinin geliştirilebile-ceği anlamına gelir. Katmanlar birbirlerinin gerçekleme ayrıntıları ile ilgilenmezler ancak birbirlerine sundukları ara yüzleri ve formatlarını tanırlar.

GES motoru

GES sunucusunun ana bileşeni GES motorudur. Uzak sunucular ile olan tüm iletişimler GES motoru aracılığı ile sağlanmaktadır. GES motoru, uzak GES sunucular ile olan iletişimleri gerçekleme, etmen için etmen kabuğu yaratarak kontrolü kabuğa aktarma, etmenler arası iletişimi gerçekleme ve etmen göçlerini sağlamakla sorumludur.

GES motoru GES sunucusunun içinde çalıştığı düzene bağlı olarak oldukça karmaşık sayılabilecek görevleri yerine getirir. Standart düzende çalışan bir GES sunucusunda GES motoru yukarıda sayılan görevleri yerine getiriyor olmasının yanında, çalıştığı düzene bağlı olarak, kimlik denetim bilgilerinin tutulması, standart GES sunucuları kimlik denetiminden geçirme, etmen şifreleme anahtarlarını tutma ve dağıtma, güncel etmen listesini tutma ve dağıtma gibi daha bir çok görevi yerine getirmektedir. Bütün GES sunucular arasındaki tüm iletişimler "SSL over RMI" adı verilen protokol ile gerçekleştirilir. SSL protokolünün çalışması için de sertifika gerektiğinden, her GES sunucusunun standart X.509 sertifika yaratmaları için gerekli fonksiyonlar mevcuttur. GES sunucuyu yöneten programcı yada yönetici yönetim modülü ile sunucu sertifikasını yaratabilir.

Etmen kabuğu

GES mimarisi etmeni, çevresindeki diğer etmenlerden koruyacak gerekli mekanizmalara sahiptir. Bu amaçla GES sunucusu, bir etmeni aktif duruma geçirmeden önce "Etmen Kabuğu" (AgentShield) isimli bir nesne yaratır. "Sarmalanmış Etmen Modeli" adı verilen bu modelde her etmen bir etmen kabuğu tarafından korunur. Etmen, etmen kabuğu nesnesinin özel bir değişkeni olarak tanımlanıp, dışarıdan gelecek erişimlere karşı korunur. Etmen kabuğunun başlıca görevleri şunlardır.

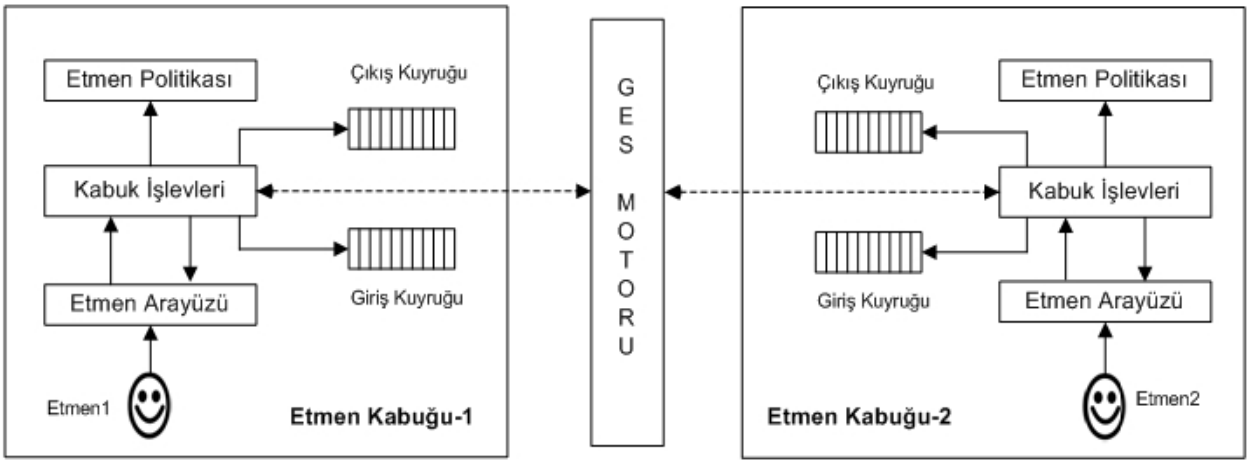
- Etmeni sarmalayarak etmene doğrudan erişimi engellemek
- Etmen arayüzü ile etmenin ilettiği istekleri gerçekleştirmek
- Etmen için mesaj giriş ve çıkış kuyrukları oluşturmak
- Etmen politikası ile etmeni ilişkilendirmek.
- Etmen kodu, durumu ve politikasını şifrelemek ve çözmek

Sarmalanmış Etmen Modeli Şekil 2 de gösterilmiştir. Etmen kabuğu, etmen kodu ve durumunun şifrelenerek diske yazılmasından da sorumludur. Etmen kod ve durum bilgisi DES algoritması kullanılarak şifrelenir. Gerekli olan DES şifreleme anahtarı ise "Güvenlik Denetleyicisi" düzende çalışan GES sunucusundan alınır. Şifreleme fonksiyonları için javanın javax.crypto ve javax.crypto.spec paketlerinden yararlanılmıştır.

Kabuk, şifreli ve sıkıştırılmış etmen kodunun belleğe yüklenmesi için özel olarak geliştirilmiş yeni bir *sınıf yükleyicisi* (Class loader) kullanır.

Güvenlik yöneticisi GES

GES sunucu üç farklı düzende çalışmayı destekler. Her GES sunucusu standart düzende çalışan bir GES sunucunun (SGES) desteklediği fonksiyonları sağlar. Bununla birlikte istenirse bir GES sunucu, "Gözleyici" dezen veya "Güvenlik Yöneticisi" düzen de ya da her iki düzende birlikte çalışacak şekilde de ayarlanabilir. "Güvenlik Yöneticisi" düzen (GYGES) ve "Gözleyici" düzen (GGES) sistemde özel görevleri olan GES sunucular için kullanılır.



Şekil 2. Sarmalanmış etmen modeli

Herhangi bir GES sunucu, başarılı bir şekilde başlayıp etmenlere ev sahipliği yapmadan önce kimlik denetiminden geçmek zorundadır. Bu kimlik denetimini kendisi için düğüm yönetici tarafından tanımlanmış olan GYGES ten karşılar. Her GES sunucu için birden fazla GYGES tanımı yapılabilmesine rağmen aynı anda aktif olan sadece bir GYGES olabilir. Aktif olan GYGES'e ulaşılamaz ise GES sunucu güvenlik ihtiyaçları için tanımlanmış olan diğer GYGES sunuculara bağlanmaya çalışır. Kimlik denetiminden geçebilen GES sunucuları artık uzak sunucular ile iletişime geçebilir ve etmenlere ev sahipliği yapabilir.

GYGES kimlik denetiminden geçirdiği her GES'e belirli bir yaşam süresine sahip olan bir bilet (ticket) verir. Bileti alan GES uzak GES sunucular ile olan haberleşmesinde bu bileti de sunmak zorundadır. Uzak GES sunucusu kendisine gelen bir isteğin sistemde kimlik denetiminden geçmiş bir GES sunucuya ait olup olmadığını anlamak için bileti GYGES'e yollar ve gerçekten GYGES'in bu bileti verip vermediğini kontrol eder. GYGES, bileti verdiğini onaylar ise isteği karşılar aksi durumda karşılamaz. Her GES sunucu bileti aldıktan sonra, yaşam süresi dolmadan yeni bir bilet almak zorundadır, aksi durumda uzak GES sunucular ile haberleşemez. GES sunucu, bilet yaşam süresinin 4 te 3 ü geçtiği andan itibaren GYGES ten yeni bir bilet isteğinde bulunur. Eğer başarısız olunursa (Örneğin GYGES'e ulaşamıyordur) varsayılan olarak 10 sn süreyle yeniden dener.

GES mimarisi sistemin hataya dayanıklı olması ve sürekliliği sağlamak üzere, sistemde birden çok GYGES'in etkin olmasına olanak sağlamaktadır. İki standart GES sunucu, GYGES olarak farklı sunucuları kullanıyor olabilir. Bu durumda kimlik denetiminden geçtikten sonra iki standart GES'in aldıkları biletler farklı sunucular tarafından verilmiş olacaktır. Bu iki sunucu üzerinde çalışan iki etmen birbirlerine mesaj yolladıkları zaman, mesajların gerçekten kimlik denetiminden geçmiş bir GES sunucudan gelip gelmediği kontrolü yapılacaktır ki bu durumda bu kontrolün hangi GYGES üzerinden yapılacağı problemi ortaya çıkmaktadır. Mimari bu problemin çözümü için GYGES ler arasında "ortak" çalışma anlayışını kullanır. Bu çalışma modeline göre ortak bir anahtarı paylaşan iki veya daha çok GYGES birbirlerinin ortağı olabilir. Hangi GYGES sunucuların birbirleri ile ortaklık ilişkisi kuracağını düğüm yöneticisi belirler. Bir GYGES bir onaylama isteği yerine getiremiyor ise, bu isteği ortaklarına yönlendirebilir.

Gözleyici GES

Hareketli bir etmen sisteminde etmenler arasında, konumdan (yer) bağımsız bir mesajlaşma alt yapısının etkin olabilmesi için, her etmenin güncel yer bilgisinin tutuluyor olması gerekmektedir. Gözleyici düzende çalışan GES sunucunun görevi güncel "etmen-yer" bilgilerini tutmak ve istendiğin de bunu GES sunuculara iletmektir.

GES etmenleri aktif duruma geçtikleri zaman kendilerini bir "isim" ile ortama duyururlar. Etmenler birbirlerine mesaj göndermeden önce, etmen isimlerini sorgular ve geriye sorguladıkları etmene bir referans (AgentPointer) elde ederler. Bu referansı kullanarak birbirlerine mesaj gönderebilirler. Herhangi bir GES sunucu, gözleyici olarak en az bir GES sunucuyu tanımlıyor olmalıdır. Birden fazla gözleyici GES tanımı olması durumunda, sunucu başarılı bir şekilde iletişim kuracağı gözleyici GES'i bulana kadar sırayla bağlantı kurmayı dener. Ancak, belirli bir anda, sadece bir gözleyici GES'i kullanıyor olabilir (Aynen bir GES sunucu için aynı andan aktif sadece bir GYGES'in olması gibi).

Güvenlik Yöneticisi düzende çalışan GES sunucular gibi Gözleyici düzende çalışan GES sunucular arasında da "ortaklık" ilişkisi vardır. Farklı olarak GGES'ler GYGES'ler gibi ortaklık ilişkisi için paylaşılan bir anahtar kullanmazlar, çünkü GGES'ler zaten GYGES'lerden kimlik denetimi almış olan sunuculardır; birbirlerini ortak olarak tanımlamaları yeterlidir.

GES etmenleri

GES mimarisi içinde etmenler JAVA dili ile geliştirilmelidirler. Her etmen yaşam döngüsü boyunca belirli bir anda yaratılma, aktif, pasif, ha-

reket halinde veya sonlanma durumlarından birinde olabilir. Bu durumlar etmenin davranışını belirleyen durumlardır ve etmen programcısı, bu durumların her biri için, etmenin yürüteceği adımları etmen kodu içinde belirtmelidir. Etmen programcısı kendisine verilen etmen şablonu içinde bu durumlara karşılık gelen metodları etmenin işlevine uygun olarak gerçekler.

Yaratılma etmenin yaşam döngüsü boyunca sadece bir kez içinde bulunduğu durumdur. Etmenlere yaratıldıkları anda bir kimlik bilgisi atanır. Bu kimlik bilgisi tekil olmayı sağlar ve GES sunucular arasındaki etmenlere ait bütün mesajlaşmalar bu kimlik bilgilerini içerir.

Bir etmen aktif duruma getirilirken daha önce yaratılmış olan etmenin çalıştırılabilir kod parçası belleğe alınır ve artık etmen bu aşamadan sonra çevresi ile etkileşime geçebilir. GES sunucusu etmenleri, bir ya da daha çok iplik (Thread) olarak çalıştırır.

Pasif hale geçirilen bir etmen ise artık etkin değildir ve son durum bilgisi GES sunucu diskinde saklanmıştır. Etmene ait çalıştırılabilir kod parçası bellekte yer almaz ve diğer etmenler pasif bir etmenin varlığından haberdar olmazlar.

Aktif olan bir etmen istediği anda başka bir düğüm üzerine taşınmak için göç isteğinde bulunabilir. Bu durumda etmen önce pasif duruma geçirilir, daha sonra kodu, durumu ve politikası uzak düğüme iletilir ve kaynak düğümdeki etmen silinerek karşı düğümde tekrar aktif hale getirilir.

Yok edilme durumu ise etmenin herhangi bir anda kendini yok etme isteğinde bulunarak sistemden silinmesi durumudur. Etmen önce pasif duruma geçirilir ve kodu, durumu ve politikası disk üzerinden silinir.

Etmen arayüzü

GES sunucusu bir etmeni yarattığı zaman "Etmen Kabuğu" (AgentSheild) isimli yeni bir nesne yaratır ve etmen nesnesi de "Etmen Kabuğu" nesnesinin özel (private) bir değişkeni olur. "Etmen Kabuğu" etmenin çevresi ile etkileşimi için etmene bir arayüz (AgentInterface) sunar. Etmen bu arayüz aracılığı ile GES sunucuya, mesajlaşma, göç etme, kopya yaratma (clone) ya da pasif hale gelme gibi isteklerini iletir.

GES mimarisi, programcıya karmaşık etmen davranışını programlayabileceği basit bir şablon ve oldukça esnek fonksiyonlar sunmaktadır. Şablon, etmenin yaşam döngüsü boyunca içinde bulunabileceği her farklı durum için, programcının bu durumlarda etmenin davranışlarını yönlendirecek kodlarını yazabileceği hazır metodlar içermektedir.

Etmen şablonu Şekil 3 te verildiği gibidir. Programcı, örneğin etmene mesaj geldiğinde yapılmasını istediklerini "OnMessageArrive" metodu içinde, etmen aktif olduğunda yapılmasını istediklerini "OnActivate" metodu içinde, etmen pasif olduğunda yapılmasını istediklerini ise "OnInactivate" metodu içinde tanımlar.

```
import agent.*;

public class Main extends Agent {
  public void OnMessageArrive() {
  }
  public void OnCreate() {
  }
  public void OnActivate() {
  }
  public void OnInactivate() {
  }
  public void OnEnd() {
  }
  public void OnTransfer() {
  }
}
```

Şekil 3. Etmen şablonu

Programcı etmen şablonundaki bu metodlarla sınırlı değildir, isterse farklı metodlar gerçekleyebilir.

GES haberleşme altyapısı

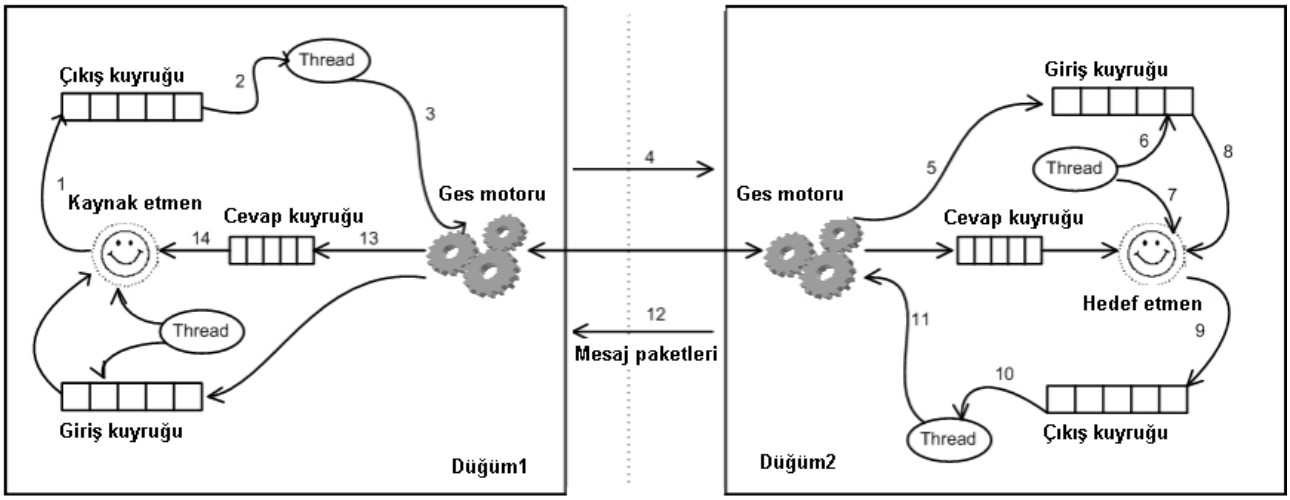
GES mimarisi etmen haberleşmesi için oldukça esnek bir yapıya sahiptir. Etmeni bloke etmeyen ve güvenli iletişimi sağlayan mekanizmalar mevcuttur.

GES mesajlaşma mimarisi şu özelliklere sahiptir:

- Etmenin çalıştığı yerden bağımsız bir mesajlaşma modeli sunar.
- İletişim için mesaj paketleri kullanır.
- Merkezi değil, dağıtık mesajlaşma yapısına sahiptir.
- İletilen mesajlar şifreli olarak gönderilir. Açık herhangi bir veri iletilmez.
- Mesajlaşma alt yapısı etmeni bloke etmeyen bir yapıya sahiptir; asenkron iletişim sözkonusudur.
- Her etmen için mesaj gönderim kuyruğu, mesaj alım kuyruğu ve yanıt alım kuyruğu oluşturulur. Bu kuyruklar Etmen Kabuğu (AgentShield) tarafından oluşturulurlar.
- Mesaj giriş ve mesaj çıkış kuyruklarını dinleyen ve mesaj varlığından etmeni haberdar eden ayrı iplikler oluşturulur. Bu nedenle, etmenin yeni bir mesajın varlığını belirlemek için kontrol kodu yürütmesine gerek yoktur; asli görevlerine yoğunlaşabilir. Ancak etmen isterse, mesaj alım kuyruğunu kendisi de dinleyebilir.
- Etmene mesajların akibeti konusunda bilgi edinebileceği ve yanıtları edinebileceği esnek fonksiyonlar sunulmuştur.

Şekil 4 iki etmen arasındaki mesaj aktarımının gerçekleşme aşamalarını göstermektedir.

Sistemdeki bir alıcı etmene mesaj iletmek isteyen gönderici etmen, önce mesaj göndereceği etmenin kimlik bilgisine bir işaretçi elde eder. Daha sonra bu işaratçi ve mesajı parametre olarak kullanarak mesaj gönderme isteğinde bulunur. Çağrı sonunda "Etmen Kabuğu" etmenin mesajını mesaj kuyruğuna yerleştirir. İletişim asenkron olarak gerçekleştiği için, bu noktadan sonra, etmen iletimin gerçeklenme ayrıntılarıyla ilgilenmez. Eğer bir yanıt bekliyorsa, mesaj yanıtı gelene kadar diğer işleriyle ilgilenebilir. Dilerse, uygun gördüğü bir zamanda, haberleşme isteğinin sonucunu da sorgulayabilir. Bu arada, çıkış kuyruğunu gözleyen iplik, mesaj iletim isteğini GES motoruna aktarır. GES motoru alıcı etmenin üzerinde yer aldığı düğümde etkin olan GES motoruyla yaptığı işbirliği sonucu mesajın alıcı etmenin giriş kuyruğuna yerleştirilmesini sağlar. Giriş kuyruğunu gözleyen iplik, derhal alıcı etmeni uyarır. Etmen bu uyarıdan sonra, uygun gördüğü anda mesajı alma isteğinde bulunabilir. Yanıtlar da mesaj şeklinde iletilir, ancak gönderici kabuk, yanıt mesajlarını paketin sonuna özel bir işaret (ACK) koyarak, GES motorunun yanıt mesajını karşı tarafın yanıt kuyruğuna yerleştirmesini sağlar. Gönderici etmen yanıt kuyruğundan istediği zaman yanıtları çekebilir. Etmen herhangi bir mesaj aldığında mesajın adını, parametre sayısını, tiplerini ve değerlerini sorgulayıp ona göre farklı davranış şekilleri gösterebilir.



Şekil 4. GES haberleşme altyapısı

Güvenlik politikaları

Hareketli etmenlerin uygulama alanlarının genişlemesi, beklenilen güvenlik ihtiyaçlarına cevap verecek yapıların geliştirilmesi ile mümkündür. Diğer önemli bir konu ise bu güvenlik ihtiyaçlarının değişken olmasıdır. Bugün için kullanılmasında sakınca olmayan bir kaynağın kullanımı, değişen çevresel faktörler nedeniyle ileride sakıncalı olabilir. O an geldiğinde bu değişikliğe uyum sağlamak mümkün olduğu kadar esnek ve kolay olmalıdır.

GES, güvenlik politikalarını etmen ve düğüm politikaları olarak ikiye ayırır. Etmen politikaları etmen ile birlikte ağ üzerinde taşınırken, düğüm politikaları sabittir. Her iki politika da çalışma anında değiştirilip etkin duruma getirilebilir.

Bir GES etmeni genel olarak iki farklı türden çağrı yapabilir. GES çağrıları ve Java API çağrıları. GES çağrıları ile etmen, haberleşme, göç etme, kopya yaratma ve diğer etmenlerin kimliklerini öğrenme gibi isteklerini etmen arayüzü aracılığı ile GES sunucusuna iletir. Her iki türden çağrı da gerçeklenmeden önce güvenlik denetleyicisi tarafından kontrol edilir ve etmen ve düğüm politika kurallarına göre işleme izin verilir ya da verilmez. Bütün JAVA API çağrıları kontrol edilemez. Buna rağmen aşağıda sıralanan ve güvenlik riski oluşturacak bütün çağrılar bu denetimden geçerler.

- Dosya sistemi fonksiyonları (yazma, okuma, silme)
- Ağ fonksiyonları (dinleme ve bağlantı amaçlı soket yaratma)
- Class yükleme fonksiyonları
- Sistem kaynaklarına erişim fonksiyonları (yazma kuyruğu, clipboard, olay kuyruğu, sistem özellikleri, vb.)
- Uygulamayı sonlandırma

GES sunucuları, aşağıdaki türden GES çağrılarını güvenlik politikaları ile denetleyebilirler.

- Mesaj gönderme ve alma
- Göç etme
- Kopya yaratma

Politikaların kontrol edilerek yukarıda söz edilen çağrılara izin verilip verilmeyeceğini GES sunucuların "Güvenlik Denetleyici" isimli modülleri gerçekler. Bu modül, JAVA'nın varsayılan "Security Manager" sınıfından türeyen oldukça yetenekli yeni bir güvenlik denetleyicidir.

Hareketli etmenlerin politika tabanlı yönetimi için çalışmalar bulunsa da (Montanari vd., 2004), GES'in esnek politika yapısı ile karşılaştırıldıklarında yetersiz kaldıkları görülmektedir.

Yönetim ve izleme

GES mimarisinin sağladığı bir diğer önemli özellik ise sistemin uzaktan bir tarayıcı yardımıyla yönetilebiliyor ve izlenebiliyor olmasıdır. Böylece düğümler üzerinde çalışan hareketli etmen sisteminin herhangi bir modülü, uzak düğümlerden yönetilebilmekte, sistemdeki tüm etmenler tek bir pencereden ve herhangi bir düğümden izlenebilmektedir.

GES, kendine özel bir sunucu yönetim modülü içerir. Bu modül, HTTP protokolü ile bir tarayıcıdan gelen bütün istekleri, kimlik denetiminden geçtikten sonra karşılayabilir.

GES mimarisi, belirli etmen aktiviteleri ve sistem olaylarının izlerinin saklanması ve gözlemlenmesine olanak sağlamaktadır. Etmen aktifivitelerinin izlenebilmesi sadece gözlemleme açısından değil, güvenlik nedeniyle de önemlidir. Örneğin yazılan bir etmenin istenildiği gibi davranmadığı gözlenebilir ya da beklenildiğinden çok daha fazla mesajlaşma aktiviteleri gerçekleştiği farkedilebilir. Bunlar, yanlış bilgilendirme, diğer etmenler tarafından kötüye kullanılma yada atağa uğrama gibi güvenlik risklerinin habercisi olabilir. Oluşan bütün izler, yönetici yada etmen programcısı tarafından yönetim modülü aracılığı ile uzaktan görüntülenebilir.

GES ve mevcut hareketli etmen sistemleri

Bugüne kadar kod hareketliliğini sağlamak için çeşitli sistemler geliştirilmiştir (Johansen vd., 1998; Danny ve Oshima, 1998; Anand vd.,

1999; Varadharan ve Foster, 2003; Makino vd., 2000; Bryce ve Vitek, 2001; Jeon vd., 2000). Bir çoğu, hareketlilik, mesajlaşma gibi temel etmen fonksiyonlarını desteklemektedir. İlk hareketli etmen sistemlerine baktığımızda programlama dili olarak TCL gibi metin (script) dilleri kullanıldığını görmekteyiz, ancak JAVA dilinin yaygınlaşması ile birlikte yeni etmen sistemlerinin hemen hemen hepsi bu dil kullanılarak geliştirilmiştir. GES'i diğer hareketli etmen sistemlerinden ayıran temel özellikler şu şekilde sıralanabilir.

- Yeni ve etkin bir etmen koruma modeli
- Etmen gizliliği ve bütünlüğünün sağlanması
- Sertifika tabanlı hazır iletişim güvenliği
- Güçlü mesajlaşma altyapısı
- Yüksek süreklilik
- Politika desteği
- Düğümleri korumaya dayalı teknikler
- Yönetilebilirlik-İzlenilebilirlik

GES ayrıca çok esnek etmen geliştirme fonksiyonları içermektedir. Bu özellikler etmen programcısının karmaşık olabilecek hareketli etmen davranışlarını kolayca gerçeklemesini sağlar. Esnek programlama arayüzü bir hareketli etmen sisteminin kullanılabilirliğini arttıran önemli bir özelliktir.

Sonuçlar

GES'in sağladığı güvenlik çözümleri tasarım aşamasında planlanıp, mimari altyapıya entegre edildikleri için sistemle bir bütün oluştururlar. GES, bu yönüyle mevcut hareketli etmen sistemlerine göre bir adım öndedir. Ayrıca düğümleri ve etmenleri korumaya yönelik, politika kullanımına dayalı dinamik mekanizmalar, etmen programcısı ve düğüm yöneticisine ek yük getirmeden, esnek bir çalışma ortamı sunmakta ve kendini kolaylıkla değişikliklere uyarlayabilen etmen uygulamalarının geliştirilmesini mümkün kılmaktadır.

GES, etmen programcısına çok esnek bir etmen geliştirme arayüzü sunar. Sistemin sunduğu etmen modeli, durum değişiklerinde etmen davranışlarını belirleyecek kodun geliştirme sürecini kolaylaştırarak, karmaşık algoritmalara olan gereksinimi ortadan kaldırır. Ayrıca, etmen programcısı ve düğüm yöneticisi, etmenleri izleme, yönetme, düğümleri ve GES modüllerini yönetme gibi önemli yönetim işlevlerini bir tarayıcı aracılığı ile rahatça uygulayabilirler.

GES mimarisi geliştirmeye açık bir sistemdir. Katmanlı yapısı, her bir katmanda ekleme ve değişikliğin bağımsız bir şekilde yapılabilmesine olanak sağlamaktadır. Örneğin, etmenlerin kullanıcılar ile ilişkilendirilerek kullanıcı bazlı bir yetkilendirme mekanizmasının kurulması, hareket halindeki etmenlere gönderilen mesajların saklanarak etmen göçü sonunda etmenlere ulaştırılması, etmenlere rol desteğinin kazandırılması, etmen aktivite izlerinden anormal etmen davranışlarının sezilebilmesi, sahte JVM lerin sezilebilmesi, güvenlik politikalarının daha fazla kontrol içerecek şekilde genişletilmesi gibi önemli konuların yeni çalışmalar sonucu mimariye eklenmesi mümkündür.

GES, sahip olduğu önemli güvenlik özellikleri yanısıra, etmen programcısı ve sistem yöneticisine sağladığı esnek çalışma ortamı nedeniyle de kullanılmaya değer yeni ve açık bir hareketli etmen sistemidir.

Kaynaklar

Anand, R. T., Neeran, M.K., Manish, K.V, Tanvir, A. ve Ram D.S., (1999). Mobile Agent Programming in Ajanta, *19th IEEE International Conference on Distributed Computing Systems (ICDCS'99)*, 190-197, Austin, TX, USA.

Borselius, N., (2002). Mobile Agent Security, *Electronics & Communication Engineering Journal*, **14**, 5, 211, London, UK.

Bryce, C. ve Vitek, J., (2001). The JavaSeal Mobile Agent Kernel, *Autonomous Agents and Multi-Agent Systems*, **4**, 359-384.

Danny B. L. ve Oshima, M., (1998). *Programming and Deploying Java(TM) Mobile Agents with Aglets(TM)*, Addison-Wesley Professional, Boston, USA.

Jeon, H., Patrie C. ve Cutkosky Mark, R., (2000). JATLite: A Java Agent Infrastructure with Message Routing, *IEEE Internet Computing*, **4**, 2, 87-96.

Johansen, D., Fred, B.S. ve Van Renesse, R., (1998). What TACOMA Taught Us, *Mobility, Mobile*

Agents and Process Migration - An edited Collection, Addison Wesley Publishing Company, Boston, USA.

Makino, S., Okoshi, T., Nakazawa, J. ve Tokuda, H., 2000. s-agent:The Design of Secure Mobile Agent System, *Proceedings*, International Conference on Distributed Systems Platforms and Open Distributed Processing, IFIP Middleware 2000 Work in Progress Session, Palisades, NY.

Montanari, R., Lupu, E. ve Stefanelli, C., (2004). Policy-based dynamic reconfiguration of mobile-code applications, IEEE Computer Society Press, *Computer,* **37**, 7, 73-80.

Varadharan, V. ve Foster, D., (2003). A Security Architecture for Mobile Agent Based Applications, *World Wide Web:Internet and Web Information System*, **6**, 93-122.

Wheeler, T., (2002). *Reducing Development Effort using the Voyager ORB*, Recursion Software Inc, Frisco, TX, USA.