Anlamsal Web Servislerinin Dinamik Çağrımı

Özgür GÜMÜŞ[1], İsmail YÜREK[2]

[1]Bilgisayar Mühendisliği Bölümü, Ege Üniversitesi, İzmir, Türkiye
[2]Bilgisayar Mühendisliği Anabilim Dalı, Fen Bilimleri Enstitüsü, Ege Üniversitesi, İzmir, Türkiye
ozgur.gumus@ege.edu.tr, ismail.yurek@gmail.com
(Geliş/Received: 12.03.2014; Kabul/Accepted: 03.04.2015)
DOI: 10.17671/btd.52483

Özet— Web servisleri, tanımlanabilen, yayımlanabilen ve standart Web protokolleri ile ağ üzerinden erişilebilen platform bağımsız, özerk hesaplama birimleridir. Günümüzde Web servisleri, iş dünyasında dağıtık uygulama geliştirme standardı olarak kabul görmektedirler. Ancak Web servisleri daha çok insanlar tarafından bulunmakta ve geliştirilen uygulamaya statik bir şekilde gömülerek kullanılmaktadır. Halbuki ihtiyaç duyulan işlevi yerine getiren fakat farklı arayüzlere sahip birçok Web servisi olabilir ve zaman içerisinde yenileri de eklenebilir. Dolayısıyla bu Web servislerinin uygulamalar tarafından otomatik olarak bulunması ve dinamik bir şekilde çalıştırılması gerekmektedir. Ancak Web servislerinin otomatik olarak bulunması, birleştirilmesi, çalıştırılması ve izlenmesi için, anlamsal Web adı verilen yeni nesil Web vizyonunun sunmuş olduğu ontolojiler kullanılarak modellenmeleri gerekmektedir. Bu servislere ise anlamsal Web servisleri denilmektedir. Dolayısıyla var olan Web servislerinin anlamsal Web servislerine dönüştürülerek yayınlanması ve ihtiyaç duyulduğunda dinamik olarak keşfedilmesi, seçilmesi, çağrılması, izlenmesi gibi işlevleri yerine getirecek araçlara ihtiyaç vardır. Bu çalışmada, servis istemcilerinin, anlamsal mesajlar ile Web servis işletimi yapabilmeleri için bir dinamik Web servis çağrım yöntemi önerilmiştir. Önerilen yöntem için, SAWSDL kullanılarak tanımlanmış olan anlamsal Web servislerinin dinamik olarak çağrımlarının yapılabildiği bir Web servis işletim çerçevesi geliştirilmiştir. Ayrıca önerilen yöntemin daha iyi açıklanması için ve geliştirilen çerçevenin işleyişini göstermesi için müşteri ilişkileri yönetimi alanında örnek bir durum çalışması yapılmıştır.

Anahtar Kelimeler— anlamsal Web servisleri, dinamik Web servis çağrımı, SAWSDL, XSLT dönüşüm

Dynamic Invocation of Semantic Web Services

Abstract— Web services are autonomous computing units which are identifiable, platform independent and accessible through the network with the standard web protocols. Nowadays web services are accepted as the standard way of distributed application development in the business world. But web services are usually discovered by humans and used statically by embedding into the developed application. However there could be many web services performing the desired functionality having different interfaces, and also new ones could be added in time. So it is required that those web services could be discovered automatically and executed dynamically by applications. On the other hand, in order to discover, compose, execute and monitor those web services automatically, they should be modelled using ontologies introduced with new generation web vision which is called the semantic web. Those web services are called as semantic web services. Hence some tools are needed to publish transforming existing web services to semantic web services and to discover, select, invocate, monitor them dynamically. In this work, a dynamic Web service invocation method has been proposed in order to carry out the Web service execution which is done by service clients using semantic messages. A Web service execution framework, which semantic Web services defined using SAWSDL can be dynamically invoked, has been developed. Also a case study in the field of customer relationships management has been performed in order to explain better the proposed method and to illustrate operation of the developed framework.

Keywords— semantic Web services, dynamic Web service invocation, SAWSDL, XSLT transformation

1. GİRİŞ (INTRODUCTION)

Web servisleri tanımlanabilen, yayımlanabilen ve standart Web protokolleri ile ağ üzerinden erişilebilen platform bağımsız, zayıf bağlı ve yeniden kullanılabilen özerk hesaplama birimleridir. Günümüzde çok yaygın olarak kullanılmakta olan Web servisleri, WSDL (Web Services Description Language) ile hazırlanmış arayüzler ile geliştirildikleri yazılım dili ve ortamına bağlı kalmaksızın çok çeşitli ortamlarda çalışan istemci yazılımlar tarafından kullanılabilmektedir.

W3C (World Wide Web Consortium) Web servisleri mimarisi çalışma grubu, Web servisleri için şu tanımı vermektedir: Bir Web servisi, bir ağ üzerinde makineden-makineye birlikte işlerliği desteklemek için tasarlanmış yazılım sistemidir ve makine tarafından işlenebilir şekilde WSDL ile hazırlanmış bir arayüze sahiptir. Diğer sistemler, bu ara yüzdeki tanımlara göre, HTTP (Hyper-Text Transfer Protocol) ya da başka protokoller üzerinden taşınan XML (eXtensible Markup Language) ile serileştirilmiş SOAP (Simple Object Access Protocol) mesajları kullanarak bu servis ile etkileşirler.

Günümüzde Web servisleri, iş dünyasında dağıtık uygulama geliştirme standardı olarak kabul görmektedirler [1]. Ancak Web servislerinin alt yapısının günümüzdeki en büyük yetersizliği, anlamsal bilgi eksikliğidir. Yani servisler tarafından kullanılan ve servisler arası iletişimde aktarılan bilgi sözdizimsel bir yapıdadır. İki farklı servisin kullandığı bilgi, aynı kavramı ifade ederken sözdizimsel farklılıklar bu bilginin iki farklı kavrammış gibi yorumlanmasına neden olabilir. Günümüzde kullanılan mimari XML verilerinin iletişimine dayanır. Bu nedenle iletinin anlamsallığı desteklenemez. Ayrıca Web servisleri daha çok insanlar tarafından bulunmakta ve geliştirilen uygulamaya statik bir şekilde gömülerek kullanılmaktadır. Halbuki ihtiyaç duyulan işlevi yerine getiren fakat farklı arayüzlere sahip birçok Web servisi olabilir ve zaman içerisinde yenileri de eklenebilir. Dolayısıyla bu Web servislerinin uygulamalar tarafından otomatik olarak bulunması ve dinamik bir şekilde çalıştırılması gerekmektedir.

Yeni nesil Web olarak adlandırılan anlamsal Web vizyonu [2], WWW Web sayfası içeriklerinin ontolojiler kullanılarak yorumlanabileceği bir seviyeye taşımayı hedeflemektedir. Anlamsal Web, kişi ya da organizasyonların ihtiyaç duyduğu servisleri arama ve düzenleme işini bilgisayarların yapmasını sağlar [3]. Sadece içeriğe değil, Web üzerindeki servislere de erişim sağlar.

Anlamsal betimlemeler ve bilgi yönetim teknikleri kullanılarak, servis yönelimli yeteneklerin keşif, birleştirme, yayımlama işlemleri daha detaylı bir seviyeye çıkarılabilir [4]. WSDL kullanılarak hazırlanan Web servisi arayüzleri, ihtiyaç duyulan anlamsal çıkarsama mekanizmaları göz önünde bulundurulduğunda anlamsal Web ortamı için yetersiz kalmaktadır. Dolayısıyla Web servislerinin otomatik olarak bulunması, birleştirilmesi, çalıştırılması ve izlenmesi için, anlamsal Web adı verilen yeni nesil Web vizyonunun sunmuş olduğu ontolojiler kullanılarak modellenmeleri gerekmektedir. Bu servislere ise anlamsal Web servisleri denilmektedir. Bu amaçla, üzerinde çalışılan ve geliştirilmekte olan OWL-S (Web Ontology Language for Services), WSMO (Web Service Modeling Ontology) ve SAWSDL (Semantic Annotations for WSDL) gibi bazı ontolojiler mevcuttur. Bu ontolojiler Web servislerinin özelliklerini ve yeteneklerini açık bir şekilde ve bilgisayar tarafından yorumlanabilir bir biçimde tanımlar.

Anlamsal Web servis teknolojileri, servisleri bulma, seçme, birleştirme, yürütme ve bu servisi diğerleri arasında izleme gibi servis odaklı uygulamaların yaşam döngüsü ile ilgili görevleri otomatikleştirmeyi amaçlamaktadır [5]. Anlamsal Web servisleri üzerine yapılan araştırmaların bir bölümü, anlamsal Web servis sistemlerinin gereksinimlerini belirlemeye ve hatta kavramsal çerçeveler ile tüm yaşam döngüsünü kapsayan mimarileri tanımlamaya ayrılmıştır [6].

Sonuç olarak, var olan Web servislerinin anlamsal Web servislerine dönüştürülerek yayınlanması ve ihtiyaç duyulduğunda dinamik olarak keşfedilmesi, seçilmesi, çağrılması, izlenmesi gibi işlevleri yerine getirecek araçlara ihtiyaç vardır. Bu çalışmada, anlamsal Web servislerinin dinamik çağrımına odaklanılmıştır. Bu amaçla servis istemcilerinin, anlamsal mesajlar ile Web servis işletimi yapabilmeleri için bir dinamik Web servis çağrım yöntemi önerilmiştir. Önerilen yöntem için, SAWSDL kullanılarak tanımlanmış olan anlamsal Web servislerinin dinamik olarak çağrımlarının yapılabildiği bir Web servis işletim çerçevesi geliştirilmiştir. Ayrıca önerilen yöntemin daha iyi açıklanması için ve geliştirilen çerçevenin işleyişini göstermesi için müşteri ilişkileri yönetimi alanında örnek bir durum çalışması yapılmıştır.

Makalenin bundan sonraki bölümleri şu şekildedir: Bölüm 2'de, bu çalışmanın alt yapısını oluşturan konular açıklanmış ve literatürde yapılan ilgili çalışmalar tanıtılmıştır. Bölüm 3'te, anlamsal Web servislerinin dinamik çağrımı için geliştirilecek olan çerçevenin tasarımı verilmiştir. Bölüm 4'te, çerçevenin gerçekleştirimi için alan ontolojileri kullanılarak anlamsal Web servis tanımlarının yapılması, servis çağrımı sırasındaki mesaj dönüşümleri ve istemci ile sunucu arasındaki etkileşim açıklanmıştır. Bölüm 5'te, müşteri ilişkileri yönetimi alanında gerçekleştirilen durum çalışması anlatılmıştır. Bölüm 6'da ise sonuçlar ile yapılan katkılar özetlenmiş ve ileriye yönelik planlanan çalışmalar verilmiştir.

2. ALT YAPI VE İLGİLİ ÇALIŞMALAR (FOUNDATIONS AND RELATED WORKS)

Bu bölümde bu çalışmanın alt yapısını oluşturan Web servisleri, anlamsal Web servisleri ve anlamsal Web servis ontolojileri hakkında temel bilgiler ve daha önce anlamsal Web servis işletim çerçeveleri alanında yapılmış ilgili çalışmalar hakkında bilgi verilmektedir.

2.1. Web Servisleri (Web Services)

Web servisleri birlikte çalışılabilen dağıtık uygulamalar geliştirmek için kullanılan bir platformdur. Bu platform, uygulamaların Web aracılığı ile birlikte-çalışabilirliği elde etmek için izledikleri standartlar kümesidir. Web servisleri herhangi bir programlama dilinde ve herhangi bir uygulama geliştirme platformunda geliştirilebilir.

Bir Web servisi, XML ile serileştirilmiş SOAP mesajları aracılığı ile ağ üzerinden erişilebilir işlemler topluluğunu tanımlayan bir arayüzdür. Bir Web servisi, standart ve biçimsel XML tasarımı kullanılarak tanımlanır ve buna Web servisinin servis tanımı denir. Servis tanımı, Web servisi ile etkileşimde bulunmak için gerek duyulan mesaj formatları, aktarım protokolleri ve konum gibi tüm detayları içerir. Arayüz, servisin gerçekleştirim detaylarını gizler. Böylece, servisin gerçekleştirildiği yazılım ya da donanım platformundan ve yazıldığı programlama dilinden bağımsız olarak kullanabilmesine olanak verir. Bu da, Web servisleri tabanlı uygulamaların zayıf bağlı, bileşen tabanlı ve çapraz teknoloji gerçekleştirimli olmalarını sağlar.

Web servisleri mimarisi servis sağlayıcı, servis kütüğü ve servis istemci rolleri arasındaki etkileşimler üzerine kuruludur. Etkileşimler yayınla, bul ve bağlan işlemlerini içerir. Bu roller ve işlemler birlikte, Web servis bileşenleri üzerinde çalışırlar. Tipik bir senaryoda, bir Web servis sağlayıcısı ağ üzerinden servisin gerçekleştirimi olan erişilebilir yazılım modülüne ev sahipliği yapar. WSDL dokümanı ile Web servisinin tanımını yapar ve bu tanımı servis kütüğünde yayınlar. Servis istemcisi, servis tanımını yerel olarak ya da bir servis kütüğünden bulmak için bul işlemini kullanır. Servis sağlayıcıya bağlanmak ve Web servis gerçekleştirimini kullanmak için servis tanımını kullanır. Bu işlemler sırasında veri ve mesajlar HTTP üzerinden XML olarak aktarılır.

Web servislerinin bir araya getirilmesiyle Web süreçleri oluşur. Web süreçleri, kuruluşların rakipleriyle, tedarikçileriyle ve müşterileriyle olan iletişimini ve etkileşimini kolaylaştırmayı amaçlayan yeni nesil bir iş akış teknolojisidir. Web süreçleri kurumsal seviyede ve çekirdek iş etkinlikleri sağlar. Kurumlar arası ve kurum içi iş akışını destekler. Web süreçleri, güvenilir ve etkin iş süreçleri yaratabilmek için Web servislerinin nasıl bir araya getirilmesi gerektiğini tanımlar. Servislerin girdi ve çıktıları birbirlerine bağlanarak bir iş akışı oluşturulur. Bir kurum için Web süreçlerinin rolü, iş ve uygulama süreçlerinin tümleşimini sağlamaktır.

2.1.1. Temel Web Servis Standartları (Basic Web Service Standards)

Web servisinin arayüzü ve etkileşimi, WSDL [7] kullanılarak tanımlanır. WSDL, XML dokümanlarını servisin teknik detayına uygun olarak düzenlemektedir. Servis tarafından kullanılan tip tanımlamaları söz dizimsel olarak ifade edilir. Bu da servisler arasında aktarılan kavramların, aynı şeyi ifade etseler de, bilgisayarlar tarafından farklı anlaşılmasına yol açmaktadır.

SOAP [8], XML ile dizgelenmiş verinin iletilebilmesi için standart sağlayan bir ileti gösterimidir. SOAP iletilerin aktarımı için HTTP protokolünü temel iletişim protokolü olarak kullanır. DCOM (Distributed Component Object Model), RMI (Remote Method Invocation) ve CORBA (Common Object Request Broker Architecture) gibi teknikler yerel ağlarda başarılı olsalar da, Web ortamına taşındıklarında büyük ölçüde başarısız olmaktadırlar. Çünkü bu teknolojiler bileşenleri arasında çok sıkı bağımlılıklar getirmektedir. Web gibi geniş ve açık bir ortamda bu kadar sıkı bir bağımlılık sağlamak mümkün değildir. Bunun yerine daha basit, RPC (Remote Procedure Call) benzeri bir mekanizma sağlamak SOAP'ın hedefidir. SOAP, HTTP üzerinden XML ileti gönderimini sağladığı için güvenlik duvarı sorunlarıyla karşılaşmaz. SOAP iletisinin taşıyacağı bilgi, Web servis tanımının yazıldığı WSDL dokümanında tanımlanır. Diğer bir deyişle SOAP içinde gönderilen veri WSDL içinde tanımlanmış olan karmaşık tiplerden biri olabilir.

UDDI (Universal Description, Discovery and Integration), istemcilerin ilgilendikleri servisleri bulabilmeleri için bir düzenek sağlar. UDDI arayüzü kullanılarak servislerin yayınlanması ve istemciler tarafından keşfedilebilmesi sağlanır. İş uygulamaları için DNS (Domain Name Server) hizmeti olarak düşünülebilir. UDDI kayıtçılarının iki kullanıcısı vardır. Birincisi servislerini yayınlamak isteyen servis sağlayıcıları, ikincisi ise uygun servis arayan istemcilerdir. UDDI, SOAP üzerine inşa edilmiştir. Yani UDDI istek ve yanıt iletileri SOAP protokolüne uygundur.

Servis istemcilerinin, karmaşık iş süreçlerini gerçekleştirebilmeleri için servislerin bir araya getirilip çalıştırılmaları gerekebilir. Bu tip süreçleri otomatik servislerle gerçekleştirmek mümkün olmayabilir. Bu durumda kullanılacak olan servislerin önceden bir araya getirilmiş olmaları gerekir. Bu tümleşim için XML tabanlı bir dil olan BPEL4WS (Business Process Execution Language for Web Services) kullanılır. BPEL4WS, süreç içindeki servislerin girdi ve çıktılarını birbirlerine uygun bir şekilde bağlayarak iş akışını oluşturur.

2.2. Anlamsal Web Servisleri (Semantic Web Services)

Anlamsal Web, veriyi hem insanların okuyabileceği hem de makinaların anlayabileceği şekilde tanımlar ve bağlar [9]. İnsanların okuyabileceğinden kasıt, geleneksel metin/resim Web belgelerinin makine tarafından gösterimi ve insan tarafından kullanılmasıdır. Makinelerin anlayabileceğinden kasıt ise, verinin çıkarsama için hazır olması ve çeşitli uygulamalarda yeniden kullanılabilir olmasıdır. Bilgi gösterimi, bilgi getirme, veri tabanları, doğal dil işleme ve makine öğrenimi gibi birçok alanı kapsayan anlamsal Web, Web bilgisinin anlamını ontolojiler aracılığıyla modeller.

Anlamsal Web, Web mimarisine dayanır: kaynakları tanımlamak için URI'leri (Uniform Resource Identifier), ontolojileri birleştirmek için bağlantıları, söz dizimi için XML'i, iletişim için de HTTP'yi kullanmaktadır. Anlamsal Web ortamındaki bilgi, makine tarafından işlenebilir ve anlaşılabilir bir anlama sahiptir. Anlamsal Web'in temel yapı taşı ontolojilerdir. Ontolojiler kavramları ve ilişkileri tanımlar. Ontolojinin eksiksiz bir tanımı yapılmaya çalışılacak olursa; ontoloji, paylaşılan bir kavramsallaştırmanın biçimsel, belirgin bir tanımıdır. Bu tanım irdelendiğinde paylaşılan kelimesi ile anlatılmaya çalışılan, ontolojilerin genel kabul görmüş bir anlayışı ifade etmesidir. Kavramsallaştırma kelimesi, bir alanın kavramsal modelinin ortaya çıkartılmasıdır. Biçimsel bir tanım olması sayesinde makine okunabilirliği sağlanır ve belirgin bir tanım olması sayesinde ise kesin terminoloji tanımlarını içerir.

Günümüzde yaygın olarak kullanılmakta olan Web servisleri, kendilerini temsil eden ve WSDL kullanılarak hazırlanan arayüzleri sayesinde geliştirildikleri yazılım dili ve/veya ortamına bağlı kalmaksızın yine çok çeşitli ortamlarda çalışan istemci yazılımlar tarafından kullanılabilmektedir. Ancak, WSDL kullanılarak tanımlanan bir servis arayüzü, özellikle ihtiyaç duyulan anlamsal çıkarsama mekanizmaları göz önünde bulundurulduğunda anlamsal Web ortamı için yetersiz kalmaktadır.

Anlamsal Web servisleri [10] ilk olarak Sheila McIlraith tarafından, hem servislerin arayüzlerinin resmi bildirim tanımlarını hem de Web servislerinin ne yaptığını açıklamak amacıyla anlamsal tanımlamalarla birlikte, Web servislerinin uzantısı olarak ileri sürülmüştür. Anlamsal Web servislerinin yetenekleri ve erişim yöntemleri ontolojiler kullanılarak tanımlanır. SAWSDL [11, 12], anlamsal Web servisi yeteneklerinin anlamsal Web ortamında temsil edilmesi ve dinamik olarak bulunup kullanılması için geliştirilen standart bir Web servisi ontolojisidir. Ayrıca, üzerinde çalışılan ve geliştirilmekte olan OWL-S [13] ve WSMO [14] gibi aday ontolojiler de mevcuttur. Bunlar Web servislerinin özelliklerini ve yeteneklerini, açık bir şekilde ve bilgisayar tarafından yorumlanabilir bir biçimde tanımlar. Endüstrideki Web servisleri ile ilgili çalışmaları tamamlayıcı niteliktedirler. Endüstri; kayıtlanma ve arama, platform bağımsız birlikte çalışabilirlik ve iyi tanımlı doküman değişimi amacıyla Web servislerinin alt-seviye tanımları için standartlar sunmaktadır. Anlamsal Web servisleri ise tam otomatik servis bulma, çağırma, birleştirme ve izlemeyi desteklemek için WSDL servislerinin özelliklerini daha anlamlı ve yorumlanabilir bir şekilde tanımlar.

2.3. Anlamsal Web Servis Ontolojileri (Semantic Web Service Ontologies)

Web servislerinin anlamsal Web ortamında çalışması için anlamsal Web servisleri alanında bazı standartlaşma çabaları vardır. SAWSDL, W3C tarafından kabul edilmiş bir standarttır. Bunun dışında en ilgi çekiciler OWL-S ve WSMO'dur. OWL-S, WSMO'ya göre daha yaygın kabul görmekte ve anlamsal Web ortamında çalışacak servislerin geliştirilmesinde daha fazla kullanılmaktadır [15]. Bu bölümde bu ontolojiler daha detaylı incelenecektir.

2.3.1. SAWSDL (Semantic Annotations for WSDL)

WSDL, bir Web servisin soyut işlevselliğini tanımlamayı ve bu servisin somut olarak nerede, nasıl çağırılacağını belirler ancak Web servisleri tanımlamasında anlamsallık içermez. Bu nedenle, iki servisin anlamları tamamen farklı olmasına rağmen benzer tanımlamaları olabilir, ya da bu servislerin benzer anlamlarının çok farklı tanımlamaları olabilir. Web servis tanımlamalarındaki bu tür belirsizliklerin çözülmesi, Web servislerini tanımlamayı ve anlamayı otomatikleştirme yönünde önemli bir adımdır ve iş uygulama entegrasyonu da dahil olmak üzere birçok alanda önemli bir üretkenlik sağlayıcısıdır.

SAWSDL [11, 12], hangi anlamsal açıklamaların kullanılarak WSDL bileşenlerine eklenebileceğini belirtir. SAWSDL, anlamsal modelleri anlatacak dil belirtmez. Bunun yerine, WSDL belgesi içinde veya dışında tanımlanan anlamsal model kavramlarını, WSDL bileşenleri içinde ek açıklamalar olarak başvurulabileceği mekanizmalar sağlar. Bu anlamlar, resmi dillerde ifade edildiğinde, Web servislerinin otomatik olarak tanımlanması ve anlaşılması sırasında, tanımlamalardaki belirsizliği gidermeye yardımcı olur.

SAWSDL, WSDL bileşenlerine anlamsal açıklamalar sağlayabilmek için WSDL 2.0 elemanlarına aşağıdaki üç yeni genişletilebilirlik özelliğini tanımlar: *modelReference* adında bir uzantı niteliği, bir WSDL bileşeni ile bazı anlamsal model kavramları arasındaki ilişkiyi tanımlar. Bu modelReference özelliği özellikle XML şema tipi tanımlarını, eleman bildirimlerini, nitelik bildirimlerini ve aynı zamanda WSDL arayüzlerini, işlemlerini ve hataları açıklamak için kullanılabilir. *liftingSchemaMapping* ve

loweringSchemaMapping adında iki uzantı niteliği, anlamsal ve sözdizimsel mesajlar arasındaki dönüşümleri yapmak için XML şema eleman bildirimlerine ve tip tanımlarına eklenir. Bu dönüşümler servis çağrımı sırasında çalışma anında yapılır.

2.3.2. OWL-S (Web Ontology Language for Services)

OWL-S servis ontolojisi [13] servis profili, servis modeli ve servis zemini olmak üzere üç bileşenden oluşur. Servis profili, servisin ne yaptığını belirtir. Servis arayan bir yazılım uygulaması o servisin ihtiyaçlarını karşılayıp karşılamadığına karar vermek için bu bilgiye ihtiyaç duyar. Bu bilgiler şunlardır:

- Servisin insanlar tarafından okunabilir tanımı
- Servis tarafından sunulan fonksiyonelliklerin tanımı (girdi ve çıktı parametreleri ile önkoşul ve etkileri)
- Benzer yeteneklere sahip birçok servis arasında muhakeme yapmaya yarayan, servis hakkında ek bilgi ve gereksinimler sağlayan fonksiyonel olmayan nitelikler. (coğrafi konum, kalite düzeyi, servis tipi, kalite garantisi, vb.)

Servis modeli, servisin nasıl çalıştığını belirtir. Yani servis çalıştırıldığında ne olacağını tarif eder. Servis modeli ayrıca birçok servisi bir görev için birleştirmeye yardımcı olur. Birleştirme eylemi temel olarak servislerin girdilerini ve çıktılarını eşleştirir ve birleştirir.

Servis zemini bir servise nasıl erişileceğinin detaylarını tanımlamaktadır. Tipik olarak servis zemini bir iletişim protokolü, mesaj şekilleri ve servise bağlanmak için gereken soket port numarası gibi diğer servise özel detayları içerir.

Özetle, servis profili uygulamanın servisi bulması için gereken bilgiyi, model ve zemin ikisi birlikte uygulamanın servisi kullanması için gereken bilgiyi sağlar. OWL-S, anlamsal Web servisleri kavramı için ortaya atılan ilk çalışmalardandır; fakat tam bir sistem değildir. Bazı elemanlarının anlamını açıkça tanımlamamıştır. Birçok araştırma grubu tarafından OWL-S ontolojilerinin kullanıldığı eşleştiriciler, planlayıcılar ve aracılar gerçekleştirilmeye çalışılmıştır.

2.3.3. WSMO (Web Service Modeling Ontology)

WSMO [14], anlamsal Web servislerini ilgilendiren birçok kavramı tanımlamak için WSMO çalışma grubu tarafından geliştirilmiş kavramsal bir modeldir. WSMO'da tanımlanan temel kavramlar şunlardır: Ontolojiler, Web servisleri, hedefler ve arabulucular. Ancak servislerin otomatik keşfi, birleştirimi ve işletimi için sadece kavramsal model yeterli değildir. Buna ek olarak formal bir dile ihtiyaç vardır, o da WSML'dir (Web Service Modeling Language). Bu dil kullanılarak Web servislerinin anlamsal tanımlamaları yapılabilir. Ayrıca bu dil çıkarsama yapabilmek için mantık tabanlı muhakeme yapabilme yeteneğine sahiptir ve kural tabanlıdır. WSMO tabanlı anlamsal Web servislerinin dinamik keşfi, seçimi, çağrımı ve arabuluculuk işlemleri için bir işletim ortamı olan WSMX (Web Service Execution Environment) de yine aynı grup tarafından geliştirilmiştir.

WSMO'da dört temel modelleme elemanı vardır:

- Ontolojiler: Diğer elemanları tanımlamak için alana özel terminolojileri sağlarlar ve iki amaca hizmet ederler: Bilginin formal anlamını tanımlamaya ve makine ve insan terminolojilerini bağlamaya.
- Web Servisleri: Üç farklı açıdan tanımlanırlar: işlevsel olmayan özellikleri, işlevselliği ve davranışı.
- Hedefler: İhtiyaç duyulan işlevselliğe uygun kullanıcı isteğini tanımlar. Hedefler de Web servislerine benzer şekilde tanımlanırlar.
- Arabulucular: Farklı bileşenler arasındaki yapısal, anlamsal veya kavramsal uyuşmazlıkları çözmek için gerekli elemanları tanımlarlar.

WSMO daha bütünleşik bir çerçevedir ancak OWL (Web Ontology Language) ve SWRL (Semantic Web Rule Language) gibi W3C standartları tabanlı değildir. Daha çok dağıtık ve heterojen servis ortamında bir iş akış sistemine benzemektedir.

2.4. İlgili Çalışmalar (Related Works)

Anlamsal Web servislerinin yayınlanması, keşfi ve yürütülmesi için geliştirilmiş birçok çerçeve bulunmaktadır. Bu çalışmaların en önemlileri bu bölümde yakından incelenmektedir.

2.4.1. WSMX (Web Service Modelling eXecution Environment)

WSMX [16], çeşitli iş yazılımları için geliştirilmiş WSMO tabanlı Web servislerinin bütünleştirilmesiyle oluşturulan gelişmiş Web servislerinin çalıştırıldığı bileşen tabanlı bir yürütme ortamıdır. İş süreçlerinin otomasyonunu, ölçeklenebilir çözümlerle esnek bir biçimde arttırmayı hedeflemektedir. WSMX Web servis tanımlamak için WSML dilini kullanır. WSMX işletim çerçevesi, WSMO için bir referans uygulamadır. WSMX anlamsal Web servislerinin kayıt, keşif ve yürütme süreçlerini yerine getirmektedir.

WSMX servis işletim çerçevesi, WSML ile tanımlanmış olan WSMO tabanlı anlamsal Web servislerin çağrımını yapmaktadır. Bu çalışmada geliştirilen işletim çerçevesi ise, W3C tarafından standart olarak kabul edilen SAWSDL ile tanımlanmış anlamsal Web servislerinin çağrımını yapmaktadır. Standart bir tanımlama dili, oluşturulan anlamsal Web servislerinin farklı

platformlardaki gerçekleştirimi ve farklı uygulamalar tarafından kullanımı sırasında ortaya çıkabilecek uyumsuzlukları önemli ölçüde azaltacaktır.

2.4.2. IRS-III (Internet Reasoning Service-III)

IRS-III (Internet Reasoning Service-III) istemciler ve servis sağlayıcıları arasında etkili iletişimi sağlayan, aracı (broker) tabanlı bir platformdur [17]. IRS-III, SESA (Semantically-enabled Service Oriented Architecture) mimarisine [5] dayanır. SESA mimarisinde, anlamsal işletim ortamı (Semantic Execution Environment) olarak adlandırılan ara yazılım (middleware) katmanı, mimarinin çekirdeğini oluşturmaktadır. Bu katman, mimaride belirtilen kavramsal işlevsellikleri tanımlamaktadır. Buradaki her bir işlevsellik, ara yazılım servisleri denilen bir takım servisler aracılığıyla gerçekleştirilmektedir. IRS-III, SESA mimarisinin örnek bir gerçekleştirimidir. Burada, bir servis istemci ile sağlayıcılar arasında arabuluculuk yapılarak, anlamsal Web servislerinin kullanıldığı uygulamalar oluşturulması için anlamsal aracı temelli bir yaklaşım izlenmektedir. IRS-III servis ontolojisi, üst düzey anlamsal Web servis kavramları için üst sınıfları kullanır. IRS-III kullanıcılara, gereken hedefleri, arabulucuları ve Web servislerini WSMO kavramlarına karşılık gelen alt sınıflar olarak tanımlayabilmelerine izin verir.

IRS-III, mimari olarak, bu çalışmada önerilen çözüm ile benzerlik göstermektedir. Her iki anlamsal servis çağrım ortamında, geleneksel Web servislerinin anlamsal ifadelerinin saklandığı servis sunucusu bulunmaktadır. Servis istemcileri kullandıkları anlamsal mesajlar ile servis bulma ve ilgili servisin çağrım işlemini sunucu üzerinden gerçekleştirmektedir. Ancak IRS-III, aracı (broker) tabanlıdır, yani keşfedilen servisler arasından hangisinin işletileceğine kendisi karar verip istemci adına servisi işletmektedir. Otomatik servis keşfi ve seçimi, bu çalışmanın kapsamı dışındadır. Ayrıca, IRS-III de WSMX gibi, standart olarak belirlenmiş servis tanımlama ontolojilerini kullanmamaktadır; servis tanımları, kendine özgü bir anlamsal servis tanımlama dili ile yapılmaktadır. Bu çalışmada ise, servis tanımları bir W3C standardı olan SAWSDL ile yapılmaktadır.

2.4.3. METEOR-S (Managing End To End OpeRations – Service Oriented Architecture)

METEOR-S (Managing End To End OpeRations – Service Oriented Architecture) projesi Wright State Üniversitesi'nde Knoesis Merkezi'ndeki çalışmaları takiben Georgia Üniversitesi'nde LSDIS Laboratuvarı'nda yürütülmüştür [18]. METEOR-S anlamsal Web servislerinin, yayınlama, bulma, dinamik bağlama ya da birleştirme, açıklama ve yürütme gibi tüm yaşam döngüsü boyunca anlamsallığını destekler ve güçlendirir. METEOR-S projesi araştırmalarında üstlenilen ayırt edici özellik, mevcut Web servis standartlarıyla güçlü bir şekilde eşlenebilmesidir. Aslında, METEOR-S felsefesi, Web servislerinin keşfi, birleştirilmesi ve yürütülmesi için daha iyi destek sağlayabilmek adına anlamsal kavramlar ile önceden var olan standartları adım adım genişletmektir.

METEOR-S projesi Web servislerinin anlamsal açıklamalarını, Web servislerinin anlamsal tabanlı keşfini ve onların birleştirilmesini ele almıştır. Anlamsal servis tanımı için SAWSDL kullanılmaktadır. Bu çalışmada da, anlamsal servisler tanımlanırken METEOR-S ile benzer bir şekilde SAWSDL kullanılmıştır. Ancak METEOR-S, anlamsal Web servis süreçlerini standartlaştırmaya odaklanmışken; bu çalışmada ise tanımlanan anlamsal Web servislerinin dinamik çağrımına odaklanılmıştır.

3. DİNAMİK ÇAĞRIM ÇERÇEVESİNİN TASARLANMASI (DESIGN OF DYNAMIC INVOCATION FRAMEWORK)

WSDL dokümanı ile tanımlanan ve XML tabanlı SOAP mesajları ile çağrımı yapılan Web servisleri, bu aşamadan sonra geleneksel Web servisleri olarak adlandırılacaktır. Geleneksel Web servisleri, belirli alan ontolojileri kullanılarak anlamsal servis kütüphanesine kaydedildikten sonra servis istemcileri tarafından keşfedilebilir ve çağrılabilir durumdadırlar. Servis istemcisi ile servis sağlayıcısı arasında anlamsal mesaj kullanılarak iletişim sağlanır. İstemciden gelen anlamsal mesaj kullanılarak servis çağrımı yapılırken, ilgili geleneksel Web servisi için kullanılacak SOAP mesajı çalışma anında servis sağlayıcısı tarafından oluşturulur ve dönen sonuç istemcinin anlayabileceği anlamsal mesaja dönüştürülür.

Servis sağlayıcısı, geleneksel Web servislerini belirli alan ontolojileri ile anlamsal model üzerinden istemcilerin kullanımına sunar. Servisin keşfi ve çağrımı sırasında bütün iletişimin belirli ontolojiler ile yapılabilmesini sağlar. Aynı amaca hizmet eden fakat girdi ve çıktıları farklı sözdizimine sahip olan geleneksel Web servislerini tek bir anlamsal model ile istemcilere sunar. Bu sayede, istemciler her bir geleneksel Web servisi için uyarlama yapmak zorunda kalmadan hepsini kullanabilir hale gelir. Anlamsal servis sağlayıcısı, geleneksel Web servisleri ile istemciler arasında bir köprü görevi görmektedir.

Bu bölümde, geleneksel Web servislerinin anlamsal Web ortamına tümleştirilmesi için keşif ve dinamik çağrım işlevlerini sunacak bir anlamsal Web servis sağlayıcısı için gereksinimler tanımlanacak, bu sağlayıcının yer aldığı anlamsal servis çağrım çerçevesinin mimarisi verilecek ve bu çerçeve üzerinde gerçekleşen anlamsal servis çağrım süreci anlatılacaktır.

3.1. Gereksinimler (Requirements)

İstemciler ile geleneksel Web servisleri arasında köprü görevini üstlenen anlamsal Web servis sağlayıcısının,

servis işletimi sırasında hem keşif hem çağrım işlemlerini yerine getirmesi beklenmektedir. Bunun için, servis sağlayıcısı aşağıda belirtilen görevleri yerine getirmelidir:

- Geleneksel Web servislerinin yeteneklerini yayınlama
- Servis istemcisinden gelen isteğe göre aday servisleri bulma
- Servis istemcilerinden gelen anlamsal mesajları yorumlama ve bu mesajı çağrım sırasında girdi parametresi olarak kullanma
- Geleneksel Web servis çağrımından elde edilen sonuçları servis istemcisine anlamsal mesaj olarak geri döndürme

Anlamsal servis sağlayıcısının bu görevleri yerine getirebilmesi için, geleneksel servis tanımlarının SAWSDL kullanarak alan ontolojileri ile genişletilmiş şekilde servis kütüphanesine kaydedilmesi gerekmektedir. Bu işlem servis kaydı sırasında bir defa yapılır. Ayrıca, çağrım sırasında çalışma anında mesaj dönüşümlerinin yapılabilmesi için anlamsal ve sözdizimsel mesajların eşleştirilmesi ve bu eşleşmenin kaydedilmesi gerekmektedir. Bu işlemler geleneksel servis sağlayıcıları tarafından, anlamsal servis kütüphanesine servis kaydı sırasında yapılır.

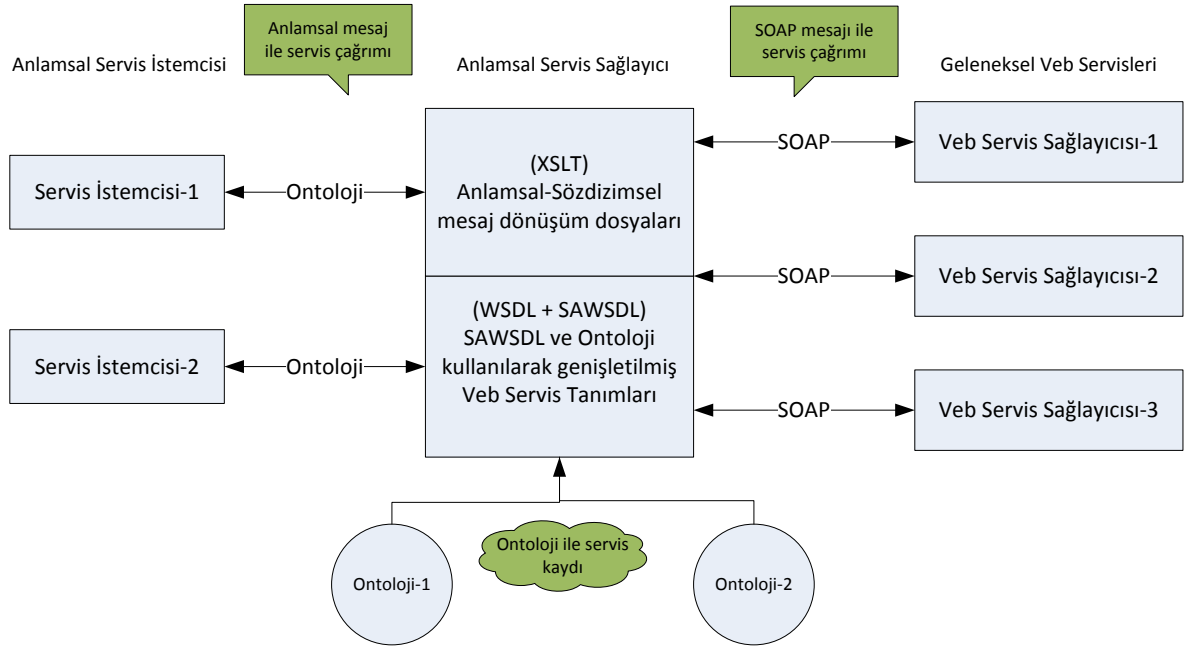
3.2. Çağrım Çerçevesinin İçsel Mimarisi (Internal Architecture of the Invocation Framework)

Anlamsal Web servis sağlayıcısı ve etkileşim içinde olduğu servis istemcisi ile geleneksel Web servis sağlayıcıları Şekil 1'de gösterilmiştir.

Servis sağlayıcı, hem servis istemci hem de geleneksel servis sağlayıcısı ile iletişim halindedir. Geleneksel Web servislerinin SAWSDL yardımıyla genişletilen ve ontolojilerle ilişkilendirilen servis tanımları, servis kütüphanesinde saklanır. Bu bilgiler servislerin yayınlanmasında ve keşfinde kullanılır. Geleneksel Web servis çağrımı sırasında kullanılacak olan SOAP mesajının oluşturulmasını ve servisten elde edilen sonuçların anlamsal mesaja dönüştürülmesini sağlayan eşleştirme bilgileri, XSLT (Extensible Stylesheet Language Transformations) dokümanı ile saklanır. Bu doküman sayesinde, çalışma anında mesaj dönüşümü sağlanır.

Anlamsal servis istemcileri, ilgilendikleri alan ontolojilerindeki kavramları kullanarak oluşturdukları anlamsal mesajlar ile servis keşfi ve çağrımı yaparlar. İstemciler geleneksel servis tanımlarıyla ilgilenmezler. İstemciler için sadece alan ontolojilerindeki kavramlar önemlidir. Bu sayede sisteme yeni bir geleneksel Web servisi dahil olduğunda, herhangi bir uyarlama yapmadan bu servisi mevcut ontolojideki kavramlar yardımıyla kullanabilir hale gelirler.

Servis sağlayıcısı, istemciden gelen isteğe göre çalıştırılacak geleneksel Web servisini belirler ve ilgili mesaj çevrim dokümanını kullanarak SOAP mesajını oluşturur. Oluşturulan SOAP mesajı ile Web servis çağrımı başlatılır ve servisten elde edilen sonuç anlamsal mesaja çevrilerek istemciye döndürülür. Servis sağlayıcısı bu işlemleri, Web servislerinin kaydı sırasında oluşturulan WSDL ve XSLT dokümanlarını kullanarak çalışma anında yapar ve geleneksel Web servisinin çağrımı için ayrı bir uyarlama yapılmaz.



Şekil 1. Anlamsal Web servis çağrım çerçevesi
(Semantic Web service invocation framework)

3.3. Çağrım Süreci (Invocation Process)

Servis istemcisi, anlamsal Web servislerinin keşfi için alan ontolojisindeki bir kavramı seçerek servis sağlayıcıya keşif isteğinde bulunur. Servis sağlayıcısı, istemcinin göndermiş olduğu alan ontolojisinde yer alan kavram için uygun servis listesini istemciye döndürür. Servis istemcisi, dönen servis sonuçlarından birini seçer ve oluşturduğu anlamsal mesaj ile beraber servis çağrım isteğini servis sağlayıcısına iletir. Servis sağlayıcısı, belirtilen anlamsal Web servisi için mesaj dönüşümlerinde kullanılacak olan XLST dokümanlarını kullanarak SOAP mesajını oluşturur ve geleneksel Web servis çağrımını yapar. Servis işletimi sonrasında dönen sonuçları da anlamsal mesaja çevirerek istemciye geri döndürür.

4. DİNAMİK ÇAĞRIM ÇERÇEVESİNİN GERÇEKLEŞTİRİMİ (IMPLEMENTATION OF DYNAMIC INVOCATION FRAMEWORK)

Geleneksel Web servislerinin birer anlamsal Web servise dönüştürülmesi için kullanılabilecek ontolojiler, Bölüm 2.3'te açıklanmıştır. Bu çalışmada, W3C tarafından standart olarak kabul edilmiş olması ve var olan geleneksel Web servisleri ile daha kolay tümleştirilebilecek olması nedeniyle SAWSDL tercih edilmiştir. Anlamsal servis çağrımı sırasında istemci ile sunucu arasındaki iletişim protokolleri, servis keşfi ve uzlaşma gibi süreçler, bu çalışmanın kapsamı dışındadır. Bu çalışmada ise, SAWSDL ontolojisi kullanılarak oluşturulmuş bir anlamsal Web servisi için, istemciden gelen istek mesajı kullanılarak ilgili operasyonun yürütülmesi, çıktının oluşturulması ve istemciye sonucun geri döndürülmesi konu alınmıştır.

Belirli bir iş alanı (domain) için oluşturulmuş olan ontolojiler kullanılarak SAWSDL yardımıyla genişletilmiş olan WSDL dokümanı, anlamsal Web servis tanımını oluşturmaktadır. Bu dokümanda, anlamsal servisin iş alanına uygun içerdiği operasyonlar, girdi ve çıktıları ile beraber tanımlanmıştır. Bu sayede istemcilerin servis çağrımı yapabilmesi için anlamsal Web servis tanımını bilmelerine gerek kalmadan, sadece yapmak istedikleri iş için ontolojide tanımlı olan kavramlar ile oluşturduğu anlamsal mesajı sunucuya iletmeleri yeterlidir. Geleneksel Web servislerinde servis çağrımı için kullanılan SOAP mesajının yerini, iş alanına özgü ontolojilerde yer alan, iş emri olarak da tanımlayabileceğimiz anlamsal mesajlar almaktadır. Anlamsal Web servis çağrımı için girdi olarak kullanılan anlamsal mesajın XML tabanlı söz diziminin, iş alanı ontolojisine ait olması gerekmektedir.

İstemci ile anlamsal Web servis sağlayıcısı arasındaki bilgi alışverişinde, alan ontolojisinde tanımlanan mesajlar girdi ve çıktı olarak kullanılır. Anlamsal Web servisinin üzerinde tanımlandığı geleneksel Web servisinin girdi ve çıktılarının, alan ontolojisindeki kavramlar ile eşleştirilmesi gerekmektedir. Bu eşleştirme için kullanıcı dostu, yarı ya da tam otomatik eşleştirme araçları tasarlanabilir. Bu sayede, geleneksel Web servislerinin anlamsal servis kütüphanesine ilgili alan ontolojisi ile genişletilerek kaydedilmesi, hem hızlandırılmış hem de hatalardan arındırılmış bir şekilde gerçekleştirilebilir.

Servis çağrımı sırasında mesaj dönüşümü için W3C tarafından tavsiye edilen XSLT [19] kullanılmıştır. Geleneksel Web servislerinin, anlamsal Web servis kütüphanesine kaydı sırasında, içerdiği operasyonlarda kullanılan parametreler için alan ontolojisinde karşılık gelen kavramlar belirlenir ve eşleştirilir. Bu eşleştirme bilgileri, servis kaydı sırasında XSLT dokümanı kullanılarak saklanır. Eşleştirmelerin tam ve eksiksiz yapılabilmesi için kullanıcı dostu araçlar tasarlanabilir. Servis çağrımı sırasında geleneksel Web servisinin kullandığı SOAP mesajı ile anlamsal Web servis istemcisinden gelen iş alanı ontolojisinde yer alan mesajlar arasındaki dönüşümler, oluşturulan XSLT dokümanı kullanılarak sağlıklı bir şekilde yapılır.

Bu bölümde, dinamik servis çağrım çerçevesinin gerçekleştirilmesi için ihtiyaç duyulan alan ontolojilerinin ve anlamsal Web servislerinin tanımlamaları için, mesaj dönüşümleri için ve servis çağrımı için kullanılan teknolojiler anlatılacaktır.

4.1. Alan Ontolojileri (Domain Ontologies)

Ontolojiler, belirli bir alandaki bilgilerin ortak ve paylaşılabilen bir anlamının oluşmasına imkan veren, yeni nesil Web olarak tanımlanan anlamsal Web'in anahtar teknolojilerinden biridir. Kavramları, bu kavramların özelliklerini, birbirleri ile olan ilişkilerini ve kısıtlamalarını, makinelerin de anlayabileceği şekilde ifade etmek için kullanılan ontolojilerini günümüzde bazı endüstri uygulamalarında da görmekteyiz.

Ontoloji, bir uygulama alanına ilişkin bilginin paylaşımı için ortak bir sözlük tanımlar. Bu sözlük kullanım alanına ilişkin kavramlar ve bu kavramlar arasındaki ilişkilerin makineler tarafından da anlaşılabilmesini sağlayacak tanımlamalar içerir.

4.2. Anlamsal Web Servis Tanımlamaları (Semantic Web Service Descriptions)

SAWSDL, WSDL belgesi içinde veya dışında tanımlanan anlamsal model kavramlarının, WSDL bileşenleri içinde ek açıklamalar olarak başvurulabileceği mekanizmalar sağlar. WSDL dokümanı, genişleyebilir bir çerçeveye sahiptir. SAWSDL bu özelliği kullanılarak, Web servisleri tanımı ve Web servislerinin çağrımı için anlamsal açıklamalar sağlar.

Web servis tanımı için kullanılan WSDL dokümanındaki bileşenler ile anlamsal modeldeki kavramlar arasındaki ilişkiyi tanımlamak için *modelReferans* adındaki uzantı niteliği kullanılır. *modelReferans* niteliği, eleman bildirimleri, nitelik bildirimleri, arayüz, işlemler ve hataları anlamsal modeldeki kavramlar ile tanımlamak için kullanılabilir.

Anlamsal Web servis çağrımı sırasında, istemciden gelen anlamsal mesaj ile geleneksel Web servisi çağrımı için kullanılan SOAP mesajı arasındaki dönüşümler için kullanılan XSLT dokümanları, *liftingSchemaMapping* ve *loweringSchemaMapping* adındaki iki uzantı niteliği aracılığıyla servis tanımında kullanılabilir.

WSDL arayüz elemanı üzerindeki *modelReferans* niteliği, anlamsal modeldeki kavramlar ile ilişki kurmaya yardımcı olur. Ek-1'deki örnekte, arayüz elemanı anlamsal modeldeki bir kavram ile ilişkilendirilmiştir.

WSDL dokümanındaki operasyonları, anlamsal modeldeki kavramlar ile tanımlamak için de *modelReferans* niteliği kullanılabilir (Ek-2).

WSDL dokümanında yer alan hata tanımlarının, anlamsal model ile ilişkilendirilmesi de Ek-3'teki gibi yapılabilir.

Geleneksel Web servislerinde kullanılan SOAP mesajlarının, anlamsal modeldeki kavramlara dönüşümü için kullanılacak olan XSLT dokümanı, *liftingSchemaMapping* niteliği ile belirtilir (Ek-4). Anlamsal modeldeki kavramların SOAP mesajına dönüşümü için kullanılacak olan XSLT dokümanını belirtmek için ise *loweringSchemaMapping* niteliği kullanılır.

4.3. Mesaj Dönüşümleri (Message Transformations)

Anlamsal Web servis çağrımı sırasında istemciden gelen anlamsal mesajın, geleneksel Web servis çağrımı sırasında fonksiyon girdisi olarak kullanılan ilgili SOAP mesajına ve çağrım sonrasında fonksiyon çıktısı olan SOAP mesajının istemcinin anlayabileceği anlamsal mesaja çevrilmesi gerekmektedir.

Bu çalışma kapsamında, servis çağrımı sırasındaki mesaj dönüşümleri için XSLT kullanılmaktadır. Kullanılan XSLT dokümanı, geleneksel Web servislerin anlamsal Web servis olarak sisteme kaydedilmesi sırasında oluşturulmaktadır. XSLT dokümanın kolay ve hatasız oluşturulması için yardımcı araçlar tasarlanabilir.

Anlamsal Web servis tanımı yapılırken, WSDL dokümanında *loweringSchemaMapping* ve *liftingSchemaMapping* nitelikleri ile belirtilen XSLT dokümanları, mesaj dönüşümleri için kullanılır. Anlamsal verinin SOAP mesajına dönüşümü sırasında, *loweringSchemaMapping* ile belirtilen dönüşüm dosyası kullanılır. SOAP mesajının anlamsal veriye dönüştürülmesi için ise *liftingSchemaMapping* ile belirtilen dönüşüm dosyası kullanılır.

Örneğin, geleneksel Web servisi olarak tasarlanan bir sipariş servisinden çağrım sonrası dönecek olası cevabın, Ek-5'teki SOAP mesajı olduğunu varsayalım. Sipariş servisinin dönüşü olan SOAP mesajının, alan ontolojisine uygun olan anlamsal mesaja dönüşümünü sağlayan XSLT tanımı, Ek-6'daki gibidir.

4.4. Servis Çağrımı (Service Invocation)

Servis çağrımı için istemcinin, kullanmak istediği anlamsal Web servisini seçerek gerekli anlamsal mesajı servis sağlayıcısına iletmesi gerekmektedir. İstemci, hangi alandaki servisleri kullanacağını belirlemeli ve bu alan için tanımlanmış olan ontolojiyi kullanarak servis sağlayıcıya mesajları iletmelidir.

Servis sağlayıcısı tarafından, servis kütüphanesine kayıtlı ve kullanılabilir olan anlamsal Web servisleri yayınlanmaktadır. Bu sayede istemciler oluşturdukları anlamsal mesaj için kullanabilecekleri servisleri, servis sağlayıcısından sorgulama yaparak öğrenebilirler. İlgili anlamsal mesaj için birden fazla servis kullanılabilir durumda ise, istemciler bu servislerden bir tanesini seçerek servis çağrımını yaparlar.

Servis seçimi için servis kalitesi, servis ücreti gibi parametreler göz önünde bulundurularak servis sağlayıcı ile istemci arasındaki anlaşma sonrası bir servis üzerinde uzlaşma sağlanabilir [20, 21]. Servis seçimi sırasında istemci ile servis sağlayıcı arasındaki uzlaşma, bu çalışmanın kapsamı dışındadır. Anlamsal Web servislerinin keşfi ve uzlaşma için SAWSDL-MX isimli mevcut bir çerçeve kullanılabilir. SAWSDL-MX [22], SAWSDL ile oluşturulmuş anlamsal Web servislerin eşleştirmesini yapan bir çerçevedir.

Anlamsal Web servisi kaydı sırasında WSDL dokümanı, SAWSDL kullanılarak ilgili alan ontolojisindeki kavramlar ile genişletilir. Servis keşfi sırasında, WSDL dokümanı üzerinde modelReferans niteliği ile ilişkilendirilmiş olan anlamsal kavramlar, eşleştirme sırasında SAWSDL-MX çerçevesi tarafından kullanılır. İstemcinin göndermiş olduğu anlamsal mesaj için kullanılabilir olan servisler bu sayede keşfedilir.

İstemciden gelen anlamsal mesaj, anlamsal-sözdizimsel mesaj dönüşümü yapan XSLT kullanılarak geleneksel Web servis çağrımı için kullanılacak olan SOAP mesajına dönüştürülür. SOAP mesajı, geleneksel Web servis çağrımı sırasında girdi olarak kullanılır. Çağrım sonrası geleneksel Web servisi tarafından dönüş değeri olan SOAP mesajı, sözdizimsel-anlamsal mesaj dönüşümü için kullanılan XSLT dosyası ile anlamsal mesaja

dönüştürülür ve oluşturulan anlamsal mesaj istemciye çağrım sonucu olarak döndürülür.

5. DURUM ÇALIŞMASI (CASE STUDY)

Bu çalışma kapsamında anlamsal Web servislerinin dinamik çağrımı için geliştirilen işletim çerçevesinin kullanımını ve işleyişini daha somut olarak gösterebilmek için bir durum çalışması yapılmıştır. Durum çalışması için, müşteri ilişkileri yönetimi (CRM) alanı seçilmiştir. Bu bölümde, müşteri ilişkileri yönetimi alanına yönelik özel kavramların bulunduğu basit bir ontoloji oluşturulacak, bu ontoloji kullanılarak müşteri ilişkileri yönetimi alanı için örnek anlamsal Web servis tanımlamaları yapılacak ve alan ontolojisindeki kavramlar ile Web servis mesajları arasındaki dönüşümler anlatılacaktır.

Kurumsal kaynak planlama uygulamaları, içerisinde farklı süreçleri barındıran geniş kapsamlı uygulamalardır. Bu uygulamaların çoğu, firmalar için özelleşmiş durumdadır. Firmalardaki satış, satış sonrası hizmet, finans yönetimi, muhasebe gibi birçok süreç dış ortamla paylaşılmaz. Bu süreçler firma içinde birbirleriyle tam entegre olarak işler. Fakat satış yapmak için yeni müşterilerle iletişime geçilmesi ve mevcut müşterilerin takip edilmesi gerekmektedir. Çeşitli pazarlama çalışmalarıyla yeni müşterilere ulaşılıp satış süreci başlatılabilir. Fakat bu işlem hem maliyetli hem de zaman alıcıdır. Müşteri takibi ve müşteri memnuniyetine yönelik çalışmalar için kullanılan uygulamalar müşteri ilişkileri yönetimi olarak adlandırılmaktadır. Bu tür uygulamalar, firmanın diğer uygulamaları ve firmanın ilişkide olduğu başka firmaların uygulamalarıyla birlikte çalışabilir hale getirilirse hem çok daha fazla verim alınabilir hem de harcanan emek en aza indirilebilir. Çalışma kapsamında, müşteri ilişkileri yönetim süreci ele alınmıştır. Bu alana özgü kavramlar, firmaların anlamsal Web servis tanımları yapılırken kullanılmak üzere ontolojiler ile modellenmiştir.

5.1. Müşteri İlişkileri Yönetimi (Customer Relationships Management)

İşletmelerin mevcut müşterilerini korumaları ve gelecekte müşteri sadakati oluşturmaları amacıyla müşteri beğenilerine uygun şekilde hareket edebilecekleri yazılımlar ile müşteri bilgilerini kayıt altında tutarak işletme davranışlarını geliştirmeye müşteri ilişkileri yönetimi denilir. Son yıllarda işletmelerin satış süreçlerini de kayıt altına almaya yönelik çalışmaları, müşteri ilişkileri yönetimi uygulamalarının sayısını ve kullanım oranlarını arttırmıştır. Günümüzde birçok firma satış süreçlerini, müşteri ilişkileri yönetimi uygulamaları ile yönetir hale gelmiştir.

Müşteri ilişkileri yönetimi, bir ürün veya hizmetin satışından önce gerçekleşen süreçlerin takip edilebilmesi için kullanılır. Bu süreçler şöyle sıralanabilir: pazarlama aktiviteleri, aday oluşumu, aday yönetimi, fırsatlaştırma, fırsat yönetimi, satış takibi, sipariş yönetimi, faturalaştırma. Herhangi bir ürünün satışından önce birtakım pazarlama aktiviteleri gerçekleştirilmektedir. Satış yaratmak amacıyla yapılan tüm etkinlikleri, pazarlama aktiviteleri olarak düşünebiliriz. Bu aktivitelerin kayıt altında tutulması, ileri bir tarihte bu faaliyetlerin raporlanıp, getirilerinin izlenmesine olanak sağlayacaktır. Gerçekleştirilen bu aktiviteler ile potansiyel müşterilerin (aday ve fırsat) kimler olduğu takip edilebilmektedir.

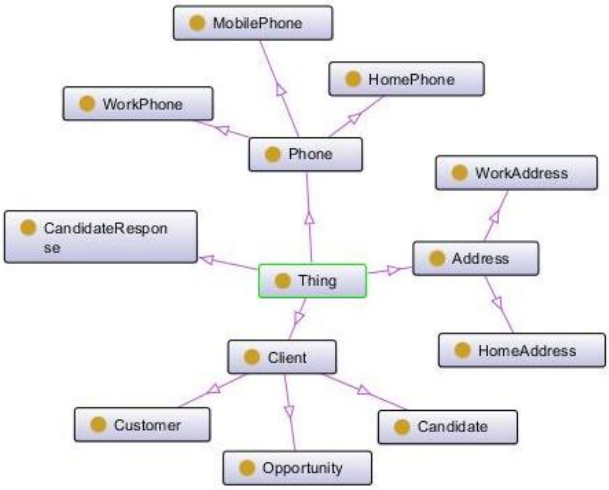
Satış yapılma olasılığı bulunan her müşteri bir aday olarak düşünülebilir. Bu müşteri daha önce sistemde kayıtlı olsa da olmasa da, kendisine yeni bir ürün satılma olasılığı varsa aday olarak düşünülmelidir. Müşteri adayı, sadece isim olarak tespit edilmiş ve sisteme kaydedilerek henüz hiçbir işlem yapılmamış olabileceği gibi kendisiyle daha önceden iletişime geçilmiş, randevu alınmış gibi aşamalarda da olabilir. Fırsat ise satış organizasyonun yapısına göre belirlenmiş aşamalardan geçmiş ve belli kriterleri sağlamış adayların, satış yapılacak potansiyel müşteri haline dönüşmesidir. Sistemde daha önceden kayıtlı olan, mevcut müşteriye yeni bir ürün satılma olasılığı da yeni bir fırsat olarak görülebilir. Fırsatlar satış aşamasına daha yakın olup, satış olasılığı yüksek, daha yakından ilgilenilmesi gereken potansiyel müşteriler olarak düşünülmelidir.

5.2. Müşteri İlişkileri Yönetimi Alan Ontolojisi (Domain Ontology of Customer Relationships Management)

Müşteri ilişkileri yönetimi alanında kullanılan aday, fırsat, müşteri, telefon ve adres bilgileri gibi kavramlar ontoloji ile modellenmiştir. Şekil 2'de kavramların tanımlandığı ontoloji ağacı yer almaktadır.

Müşteri ilişkileri yönetiminde ürün satışı yapılan müşteri satış işlemine kadar üç farklı şekilde tanımlanır.

- Aday: Satış yapmak için iletişim bilgilerine ulaşılan kişi ya da kurum aday olarak değerlendirilir. Adayın ürünü alma olasılığı düşüktür ve ikna edilmesi gerekir. Ontolojide aday için *Candidate* sınıfı tanımlanmıştır.
- Fırsat: Aday ile konuşulduğunda ve ürün tanıtımı yapıldığında ürünü alma olasılığı yüksek ise bu bir fırsat olarak değerlendirilebilir. Fırsat ürünü almaya çok yakındır. Ontolojide *Opportunity* olarak tanımlanmıştır.
- Müşteri: Ürün satışı yapılmış olan kişi ya da kurum müşteri olarak değerlendirilir. Müşteri en az bir kere ürünü almıştır. Ontolojide *Customer* olarak tanımlanmıştır.



Şekil 2. Müşteri ilişkileri yönetimi alan ontolojisi
(Domain ontology of customer relationships management)

Süreç içinde takip edilen müşterilerin iletişim bilgileri de önemli bir yere sahiptir. Müşteriler ile iletişime geçerken ya da ziyaretine giderken kullanılan önemli bilgilerdir.

- Telefon: Satış yapmak için müşteri ziyaretine gitmeden önce telefon ile ön görüşme yapmak ve ön bilgi almak, satışın gerçekleşmesinde önemli bir yere sahiptir. Ontolojide telefon bilgileri *HomePhone*, *MobilePhone*, *WorkPhone* olarak *Phone* sınıfı altında tanımlanmıştır.
- Adres: Telefon görüşmesinde ürünü alma olasılığı yüksek görülen müşteriler ile yerinde görüşmek, satışın başarılı olmasında önemli bir etkiye sahiptir. Müşterinin adres bilgileri *HomeAddress*, *WorkAddress* olarak *Address* sınıfı altında tanımlanmıştır.

5.3. Müşteri İlişkileri Yönetimi Anlamsal Web Servis Tanımları (Semantic Web Service Descriptions of Customer Relationships Management)

Müşteri ilişkileri yönetiminde satış için iletişim bilgilerine ulaşılan kişi ya da kurumlar aday olarak değerlendirilir. Müşterinin sisteme dahil olduğu ilk aşamadır. Bu çalışma kapsamında geliştirilen örnek uygulamada, aday kaydının yapıldığı iki tane geleneksel Web servisi tanımlanmıştır. Oluşturulan geleneksel Web servisleri, SAWSDL yardımıyla Bölüm 5.2'de oluşturulan alan ontolojisi ile genişletilerek birer anlamsal Web servis haline getirilmiştir.

WebServiceX olarak adlandırılan geleneksel Web servisinin WSDL dokümanındaki aday kaydetme metodunun girdi mesajı, SAWSDL ile Ek-7'deki gibi genişletilmiştir. Web servisinin WSDL tanımında *operation* ile belirtilen *CreateCandidate* metodunun girdi parametresi olarak kullanılan *CreateCandidateRequest* mesajı, *modelReferans* ile müşteri ilişkileri alan ontolojisindeki *Candidate* kavramı ile ilişkilendirilmiştir. Anlamsal Web servis çağrımı sırasında *CreateCandidate* metodu için *Candidate* anlamsal mesajı kullanılacaktır.

WebServiceY adıyla oluşturulan ikinci geleneksel Web servisinin WSDL dokümanındaki aday kaydetme metodu da, aynı alan ontolojisi kullanılarak Ek-8'deki gibi genişletilmiştir. Web servisinin WSDL tanımında *operation* ile belirtilen *SaveCandidate* metodunun girdi parametresi olarak kullanılan *SaveCandidateRequest* mesajı, *modelReferans* ile müşteri ilişkileri alan ontolojisindeki *Candidate* kavramı ile ilişkilendirilmiştir. Anlamsal Web servis çağrımı sırasında *SaveCandidate* metodu için *Candidate* anlamsal mesajı kullanılacaktır.

Müşteri ilişkileri yönetim sisteminde aday kaydı için kullanılan iki farklı geleneksel Web servisi, farklı metot isimleri ve farklı girdiler ile aynı hizmeti vermektedir. Bu iki geleneksel Web servisi, müşteri ilişkileri yönetimi alanı için oluşturulan ontolojideki ortak kavramlar kullanılarak SAWSDL yardımıyla birer anlamsal Web servis haline getirilmiştir. Bu sayede, anlamsal Web servisleri ile aday kaydı yapılırken ortak bir anlamsal mesaj kullanılabilir hale gelmiştir. İstemci, aday kaydı yaparken kullanmak istediği her servis için ayrı bir SOAP mesajı oluşturmak yerine tek bir anlamsal mesaj kullanabilir hale gelmiştir. İstemcinin aradığı servisi bulması kolaylaşmıştır. Servis arama işlemi daha doğru sonuçlar verir hale gelmiştir.

5.4. Müşteri İlişkileri Yönetimi Mesaj Dönüşümleri (Message Transformations of Customer Relationships Management)

Servis çağrımı sırasında, aday kaydı için istemci tarafından gönderilen anlamsal mesajın, ilgili geleneksel Web servisi çağrımı için girdi olarak kullanılan SOAP mesajına ve servis çıktısının, anlamsal mesaja dönüştürülmesi gerekmektedir. Mesaj dönüşümleri XSLT dokümanı ile yapılmıştır. Anlamsal mesajın SOAP mesajına dönüştürülmesi için kullanılan XSLT dokümanı Ek-9'daki gibidir.

Anlamsal-sözdizimsel mesaj dönüşümü ile oluşturulan SOAP mesajı, Web servis çağrımı için kullanılır ve servis çıktısı olan SOAP mesajı, sözdizimsel-anlamsal mesaj dönüşümü ile alan ontolojisindeki kavramları içeren bir anlamsal mesaja dönüştürülür. Bu dönüşümü yapan XSLT dokümanı tanımı Ek-10'daki gibidir.

Anlamsal Web servis çağrımı sırasında mesaj dönüşümleri için XSLT dokümanından yararlanılmıştır. Bu çalışma kapsamında müşteri ilişkileri alanında geliştirilen uygulamadaki mesaj dönüşümlerinde, XSLT dokümanı yeterli olmuştur. Daha karmaşık anlamsal mesajların olduğu anlamsal servis çağrımlarında, XSLT dokümanı dönüşüm için yeterli olmayabilir. Bu aşamada

XSLT yerine başka dönüşüm yöntemleri kullanılabilir ya da programlama seviyesinde ek geliştirmeler yapılabilir.

5.5. Müşteri İlişkileri Yönetimi Dinamik Web Servis Çağrımı (Dynamic Web Service Invocation of Customer Relationships Management)

Geliştirilen örnek uygulamada müşteri ilişkileri yönetimi sisteminde aday kaydı hizmeti veren iki tane anlamsal Web servisi tanımlanmıştır. Servis çağrımı sırasında anlamsal Web servis sağlayıcı ile istemci arasındaki etkileşim için örnek senaryo hazırlanmıştır.

Bu senaryoda, istemci oluşturmuş olduğu anlamsal mesaj için servis keşfine başlar. Servis sağlayıcıdan ilgili anlamsal mesaj için kullanabileceği servis listesini alır. İstemci uygulamasında, bulunan Web servisleri listelenir ve kullanıcının servis seçimi yapması beklenir (Ek-11). Kullanıcı servis seçimini yaptıktan sonra ilgili servis için çağrım işlemi başlar. İstemci tarafından servis sağlayıcıya, anlamsal mesaj ve seçilen anlamsal Web servis bilgisi gönderilir. Servis sağlayıcı anlamsal mesaj için hazırlanmış olan XSLT dokümanını bulur ve anlamsal mesaj SOAP mesajına dönüştürülür (Ek-11). Seçilen anlamsal Web servisi için kullanılan geleneksel Web servisinin adresi bulunur ve bu adrese oluşturulan SOAP mesajı gönderilerek çağrım işlemi yapılır. SOAP mesajı gönderilerek çağrılan Web servisinden dönen sonuç için sözdizimsel-anlamsal mesaj dönüşümü yapan XSLT dokümanı bulunur ve çağrım sırasındaki dönüşüme benzer fakat ters yönde bir şekilde SOAP mesajı anlamsal mesaja dönüştürülür. Oluşturulan anlamsal mesaj, istemciye çağrım sonucu olarak döndürülür.

Hazırlanan uygulama ile anlamsal Web servis çağrımı sırasında gerçekleşen adımlar örnek senaryo üzerinden anlatılmıştır. Servis sağlayıcı ile istemci arasındaki mesaj alışverişi programlama dili seviyesindeki metot çağrımı ile gerçekleştirilmiştir. İletişim için intranet ya da internet gibi ortamlar kullanılmak istenilirse ilgili iletişim protokollerini gerçekleştiren bir iletişim alt yapısı hazırlanmalıdır.

6. SONUÇLAR VE İLERİYE DÖNÜK ÇALIŞMALAR (CONCLUSIONS AND FUTURE WORKS)

Son olarak bu bölümde, anlamsal Web servislerinin kullanılma sebebi, seçilen müşteri ilişkileri yönetimi alanında geliştirilen anlamsal Web servis çağrımının doğurduğu sonuçlar ve bu sonuçların dinamik Web servis çağrımına sağlayabileceği katkılar ile ilgili bilgi ve önerilerden bahsedilecektir.

Geleneksel Web servislerinde SOAP ile taşınan mesajlar sözdizimsel olarak ifade edilmektedir. Bunun sonucu olarak aynı işi yapan fakat farklı sözdizimine sahip SOAP mesajları ile çalışan Web servisleri, makinalar tarafından farklı Web servisleri olarak algılanmaktadır. Bu servislerin uygulamalar tarafından kullanılabilmesi için geliştiricilerin ilgili servisleri sisteme dahil ederken uygulamalarında gerekli uyarlamaları yapmaları gerektirmektedir.

Aynı hizmeti veren ve farklı sözdizimsel mesajlar ile çalışan Web servislerinin, makinalar tarafından aynı olduğunun algılanabilmesi ve herhangi bir uyarlama yapmadan kullanılabilir hale gelmesi için ortak bir ontoloji kullanılarak anlamsal Web servislerine dönüştürülmeleri gerekmektedir. Anlamsal Web servis tanımı için kullanılabilen OWL-S, SAWSDL ve WSMO gibi farklı servis ontolojileri mevcuttur. Bu ontolojilerden özellikle OWL-S ve WSMO tabanlı anlamsal Web servislerini bulma, seçme, birleştirme, yürütme ve izleme gibi etkinlikleri otomatikleştirmek için çeşitli platform ve çerçeveler bulunmaktadır. Bölüm 2.4'te bu çalışmaların en önemlilerinden bahsedilmiştir. Bu çalışmada ise bir W3C standardı olarak kabul edilen SAWSDL tercih edilmiştir. Böylece hem var olan hem yeni geliştirilecek olan servislerden aynı işlevi sunan fakat farklı arayüzlere sahip Web servisleri, standartlara dayanarak çalışma zamanında dinamik olarak bulunabilir ve erişilebilir hale gelecektir. Uygulama geliştiricilerin, veri dönüşümü ve statik servis çağrımı için, tasarım zamanında uygulamaya özel kod yazmalarına gerek kalmayacaktır. Bu amaçla SAWSDL tabanlı anlamsal Web servislerinin dinamik olarak çağrımlarının yapılabildiği bir servis çağrım çerçevesi tasarlanmış ve prototipi gerçekleştirilmiştir. Ayrıca bu prototip üzerinde, müşteri ilişkileri yönetimi alanında örnek bir durum çalışması yapılarak, prorotipin beklenen işlevselliği sağladığı görülmüştür.

Gerçekleştirilen prototip temel görevlerini yerine getirmekle birlikte bazı eksiklikleri de bulunmaktadır. Bundan sonraki dönemlerde ileriye dönük olarak yapılabilecek çalışmalar aşağıda listelenmiştir:

- Otomatik servis çağrımı yapabilmek için servis sağlayıcı ile uzlaşma sırasında müzakere ve pazarlık yapılabilmesini sağlayacak modülün gerçekleştirilmesi, işletim çerçevesinin ek katma değer üretebilmesini sağlayacaktır.
- Yeni Web servislerinin sisteme dahil edilmesini kolaylaştırmak ve olası hataları en aza indirmek için kullanıcı dostu yarı ya da tam otomatik araçların tasarlanması, çerçevenin kullanılırlığını arttıracaktır.
- İşletim çerçevesinin çalışma anında izlenebilmesini sağlayan modülün geliştirilmesi, geleneksel Web servislerinin fonksiyonel olmayan, müzakere sırasında kullanılabilecek kalite parametrelerinin hesaplanmasını sağlayacaktır.
- Aynı alana (domain) ait farklı ontolojiler kullanılarak oluşturulan anlamsal Web servislerinin bir üst ontoloji yardımıyla eşleştirilebilmesi için ontoloji yönetim modülünün geliştirilmesi de ek katma değer sağlayacaktır.

KAYNAKLAR (REFERENCES)

[1] M. P. Singh, M. N. Huhns, "Service-Oriented Computing - Semantic, Processes, Agents", *John Wiley & Sons*, 2005.

[2] T. Berners-Lee., J. Hendler, O. Lassila, "The Semantic Web", *Scientific American*, 284(5), 34-43, 2001.

[3] S. Decker, S. Melnik, F. van Harmelen, D. Fensel, M. Klein, J. Broekstra, M. Erdmann, I. Horrocks, "The Semantic Web: the roles of XML and RDF", *IEEE Internet Computing*, 4(5), 63–74, 2000.

[4] D. Martin, M. Paolucci, S. McIlraith, M. Burstein, D. McDermott, D. McGuinness, K. Sycara, "Bringing Semantics to Web Services: The OWL-S Approach", **In Proceedings of the 1st International Workshop on Semantic Web Services and Web Process Composition**, 26-42, 2004.

[5] T. Vitvar, A. Mocan, M. Kerrigan, M. Zaremba, M. Zaremba, M. Moran, E. Cimpian, T. Haselwanter, D. Fensel, "Semantically-enabled Service Oriented Architecture: Concepts, Technology and Application", *Service Oriented Computing and Applications*, 1(2):129-154, 2007.

[6] M. Burstein, C. Bussler, M. Zaremba, T. Finin, M. Huhns, M. Paolucci, A. Sheth, S. Williams, "A Semantic Web Services Architecture", *IEEE Internet Computing*, 9(5), 72–81, 2005.

[7] Internet: Web Services Description Language (WSDL), http://www.w3.org/TR/wsdl, 27.11.2014.

[8] Internet: Simple Object Access Protocol (SOAP), http://www.w3.org/TR/soap/, 27.11.2014.

[9] M. Paolucci, T. Kawamura, T. R. Payne, K. Sycara, "Semantic Matching of Web Service Capabilities", **In Proceedings of the 1st International Semantic Web Services Conference**, 2002.

[10] S. A. Mcilraith, T. C. Son, H. Zeng, "Semantic Web Services", *IEEE Intelligent Systems*, 16(2), 46-53, 2001.

[11] J. Kopecky, T. Vitvar, C. Bournez, J. Farrell, "SAWSDL: Semantic Annotations for WSDL and XML Schema", *IEEE Internet Computing*, 11(6), 60-67, 2007.

[12] Internet: Semantic Annotations for WSDL and XML Schema (SAWSDL), http://www.w3.org/TR/sawsdl/, 27.11.2014.

[13] D. Martin, M. Burstein, D. McDermott, S. McIlraith, M. Paolucci, K. Sycara, D. L. McGuiness, E. Sirin , N. Srinvasan, "Bringing Semantics to Web Services with OWL-S", *World Wide Web*, 10(5), 243-277, 2007.

[14] C. Feier, A. Polleres, R. Dumitru, J. Domingue, M. Stollberg, D. Fensel, "Towards Intelligent Web Services: The Web Service Modeling Ontology (WSMO)", **In Proceedings of the 1th International Conference on Intelligent Computing**, Hefei, China, 2005.

[15] R. Lara, D. Roman, A. Polleres, D. Fensel, "A Conceptual Comparison of WSMO and OWL-S", **In Proceedings of the 2nd European Conference on Web Services**, Erfurt, Germany, 2004.

[16] T. Haselwanter, P. Kotinurmi, M. Moran, T. Vitvar, T., M. Zaremba, "WSMX: A Semantic Service Oriented Middleware for B2B Integration", **In Proceeding of the 4th International Conference on Service-Oriented Computing**, 477-483, 2006.

[17] L. Cabral, J. Domingue, S. Galizia, A. Gugliotta, V. Tanasescu, C. Pedrinaci, B. Norton, "IRS-III: A Broker for Semantic Web Services Based Applications", **In Proceeding of the 5th International Semantic Web Conference**, 201-214, 2006.

[18] A. A. Patil, S. A. Oundhakar, A. P. Sheth, K. Verma, "METEOR-S Web Service Annotation Framework", **In Proceedings of the 13th International Conference on World Wide Web**, 553-562, 2004.

[19] Internet: Extensible Stylesheet Language Transformations (XSLT), http://www.w3.org/TR/xslt20/, 27.11.2014.

[20] J. Cardoso, A. P. Sheth, J. A. Miller, J. Arnold, K. Kochut, , "Quality of Service for Workflows and Web Service Processes", *Journal of Web Semantics*, 1:281–308, 2004.

[21] Y. Lu, Z. Gao, K. Chen, "A Dynamic Composition Algorithm of Semantic Web Service Based on QoS", **In Proceeding of the 2nd International Conference on Future Networks**, Sanya, Hainan, 354-356, 2010.

[22] M. Klusch, P. Kapahnke, "Semantic Web Service Selection with SAWSDL-MX", **In Proceeding of the 7th International Semantic Web Conference**, Karlsruhe, Germany, 2008.

EKLER (APPENDICES)

1
```
<wsdl:interface name="Order"
     sawsdl:modelReference="http://example.org/electronics">
     …
</wsdl:interface>
```

2
```
<wsdl:operation name="order"
     sawsdl:modelReference="http://example.org/purchaseorder#RequestPurchaseOrder">
     <wsdl:input element="OrderRequest"/>
     <wsdl:output element="OrderResponse"/>
</wsdl:operation>
```

3
```
<wsdl:interface name="Order">
     <wsdl:fault name="ItemUnavailableFault"
          element="AvailabilityInformation"
          sawsdl:modelReference="http://example.org/purchaseorder#ItemUnavailable">
          …
</wsdl:interface>
```

4
```
<xs:element name="OrderResponse" type="confirmation" />
     <xs:simpleType name="confirmation"
          sawsdl:modelReference="http://example.org//purchaseorder#OrderConfirmation"
          sawsdl:liftingSchemaMapping="http://example.org/mapping/Response2Ont.xslt">
          <xs:restriction base="xs:string">
               <xs:enumeration value="Confirmed" />
               <xs:enumeration value="Pending" />
               <xs:enumeration value="Rejected" />
          </xs:restriction>
     </xs:simpleType>
</xs:element>
```

5
```
<OrderResponse
     xmlns="http://www.w3.org/2002/ws/sawsdl/spec/wsdl/order#">
     Confirmed
</OrderResponse>
```

6
```
<xsl:transform version="2.0"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:po="http://www.w3.org/2002/ws/sawsdl/spec/wsdl/order#"
     xmlns:POOntology="http://www.w3.org/2002/ws/sawsdl/spec/ontology/purchaseorder#">
     <xsl:output method="xml" version="1.0" encoding="iso-8859-1" indent="yes" />
     <xsl:template match="/">
          <rdf:RDF>
               <POOntology:OrderConfirmation>
                    <hasStatus
                         rdf:datatype="http://www.w3.org/2001/XMLSchema#string">
                         <xsl:value-of select="po:OrderResponse" />
                    </hasStatus>
               </POOntology:OrderConfirmation>
          </rdf:RDF>
     </xsl:template>
</xsl:transform>
```

7
```
<wsdl:message name="CreateCandidateRequest"
     sawsdl:modelReference="crm:Candidate">
     <wsdl:part name="parameters"
          element="ns:CreateCandidate" />
</wsdl:message>
<wsdl:message name="CreateCandidateResponse">
     <wsdl:part name="parameters"
          element="ns:CreateCandidateResponse" />
</wsdl:message>
<wsdl:portType name="WebServiceXPortType">
     <wsdl:operation name="CreateCandidate">
          <wsdl:input message="ns:CreateCandidateRequest"
               wsaw:Action="urn:CreateCandidate" />
          <wsdl:output message="ns:CreateCandidateResponse"
               wsaw:Action="urn:CreateCandidateResponse" />
     </wsdl:operation>
</wsdl:portType>
```

8
```
<wsdl:message name="SaveCandidateRequest"
     sawsdl:modelReference="crm:Candidate">
     <wsdl:part name="parameters"
          element="ns:SaveCandidate" />
</wsdl:message>
<wsdl:message name="SaveCandidateResponse">
     <wsdl:part name="parameters"
          element="ns:SaveCandidateResponse" />
</wsdl:message>
<wsdl:portType name="WebServiceYPortType">
     <wsdl:operation name="SaveCandidate">
          <wsdl:input message="ns:SaveCandidateRequest"
               wsaw:Action="urn:SaveCandidate" />
          <wsdl:output message="ns:SaveCandidateResponse"
               wsaw:Action="urn:SaveCandidateResponse" />
```

```
     </wsdl:operation>
</wsdl:portType>
```

9

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet version="1.0"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:CRM="http://www.ege.edu.tr/ontologies/2011/11/CRM.owl#" >
     <xsl:output method="xml" indent="yes" />
     <xsl:template match="/">
          <soap:Envelope
               xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema" >
               <soap:Body>
                    <CreateCandidate
                         xmlns="http://x.web.edu.ege" >
                         <candidate>
                              <xsl:for-each
                                   select="rdf:RDF/owl:NamedIndividual">
                                   <xsl:if
                                        test="contains(rdf:type/@rdf:resource,'Candidate')">
                                        <name
                                             xmlns="http://x.web.edu.ege/xsd">
                                             <xsl:value-of select="CRM:clientName" />
                                        </name>
                                        <phone
                                             xmlns="http://x.web.edu.ege/xsd">
                                             <xsl:value-of
                                                  select="/rdf:RDF/owl:NamedIndividual[contains(rdf:type/@rdf
                                                  :resource, 'MobilePhone')]/CRM:phoneNumber" />
                                        </phone>
                                        <address
                                             xmlns="http://x.web.edu.ege/xsd" >
                                             <xsl:value-of
                                                  select="/rdf:RDF/owl:NamedIndividual[contains(rdf:type/@rdf
                                                  :resource, 'WorkAddress')]/CRM:addressDescription" />
                                        </address>
                                   </xsl:if>
                              </xsl:for-each>
                         </candidate>
                    </CreateCandidate>
               </soap:Body>
          </soap:Envelope>
     </xsl:template>
</xsl:stylesheet>
```

10

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:CRM="http://www.ege.edu.tr/ontologies/2011/11/CRM.owl#"
     xmlns:savecandidateres="http://www.semanticweb.org/ontologies/2012/3/savecandidateres.owl#"
     xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
     xmlns:ns="http://x.web.edu.ege"
     xmlns:ax21="http://x.web.edu.ege/xsd">
     <xsl:output method="xml" indent="yes" />
     <xsl:template match="/">
          <rdf:RDF
               xmlns="http://www.semanticweb.org/ontologies/2012/3/savecandidateres.owl#"
               xml:base="http://www.semanticweb.org/ontologies/2012/3/savecandidateres.owl"
               xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
               xmlns:savecandidateres="http://www.semanticweb.org/ontologies/2012/3/savecandidateres.owl#"
               xmlns:owl="http://www.w3.org/2002/07/owl#"
               xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
               xmlns:CRM="http://www.ege.edu.tr/ontologies/2011/11/CRM.owl#">
               <owl:Ontology rdf:about="http://www.semanticweb.org/ontologies/2012/3/savecandidateres.owl">
                    <owl:imports rdf:resource="http://www.ege.edu.tr/ontologies/2011/11/CRM.owl" />
               </owl:Ontology>
               <owl:NamedIndividual rdf:about="savecandidateres;candidateres1">
                    <xsl:for-each
                         select="soapenv:Envelope/soapenv:Body/ns:CreateCandidateResponse/ns:return">
                         <rdf:type rdf:resource="crm;CandidateResponse"/>
                         <errorDescription rdf:datatype="xsd;string">
                              <xsl:value-of select="ax21:errorMessage" />
                         </errorDescription>
                         <sucess rdf:datatype="xsd;boolean">
                              <xsl:value-of select="ax21:success" />
                         </sucess>
                    </xsl:for-each>
               </owl:NamedIndividual>
          </rdf:RDF>
```

```
     </xsl:template>
</xsl:stylesheet>
```

11

```
Parsing semantic web service description file: WebServiceX.sawsdl.wsdl
Retrieving document at 'file:/C:/Users/iyure_000/Desktop/x/dwsi/files/swsrepository/WebServiceX.sawsdl.wsdl'.
Parsing semantic web service description file: WebServiceY.sawsdl.wsdl
Retrieving document at 'file:/C:/Users/iyure_000/Desktop/x/dwsi/files/swsrepository/WebServiceY.sawsdl.wsdl'.
Semantic web service execution environment is ready...
Do you want to make service request [y/n]? :
y
Client starting discover services...
Client select service...
2 semantic web service found for message: Candidate
        1. WebServiceXPortType::http://x.web.edu.ege::CreateCandidate
        2. WebServiceYPortType::http://y.web.edu.ege::SaveCandidate
Enter service number to call:
1
Invocation starting to call service: WebServiceXPortType::http://x.web.edu.ege::CreateCandidate

Request - Semantic message:
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
     <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
     <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
     <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
     <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
     <!ENTITY CRM "http://www.ege.edu.tr/ontologies/2011/11/CRM.owl#" >
     <!ENTITY savecandidatereq "http://www.semanticweb.org/ontologies/2012/3/savecandidatereq.owl#" >
]>
<rdf:RDF
     xmlns="http://www.semanticweb.org/ontologies/2012/3/savecandidatereq.owl#"
     xml:base="http://www.semanticweb.org/ontologies/2012/3/savecandidatereq.owl"
     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
     xmlns:savecandidatereq="http://www.semanticweb.org/ontologies/2012/3/savecandidatereq.owl#"
     xmlns:owl="http://www.w3.org/2002/07/owl#"
     xmlns:xsd="http://www.w3.org/2001/XMLSchema#" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:CRM="http://www.ege.edu.tr/ontologies/2011/11/CRM.owl#">
     <owl:Ontology rdf:about="http://www.semanticweb.org/ontologies/2012/3/savecandidatereq.owl">
          <owl:imports rdf:resource="http://www.ege.edu.tr/ontologies/2011/11/CRM.owl" />
     </owl:Ontology>
     <owl:NamedIndividual rdf:about="&savecandidatereq;candidate1">
          <rdf:type rdf:resource="&CRM;Candidate">d</rdf:type>
          <CRM:clientName rdf:datatype="&xsd;string">Ali Durmaz</CRM:clientName>
          <CRM:hasHomeAddress rdf:resource="&savecandidatereq;homeAddress" />
          <CRM:hasHomePhone rdf:resource="&savecandidatereq;homePhone" />
          <CRM:hasMobilePhone rdf:resource="&savecandidatereq;mobilePhone" />
          <CRM:hasWorkAddress rdf:resource="&savecandidatereq;workAddress" />
          <CRM:hasWorkPhone rdf:resource="&savecandidatereq;workPhone" />
     </owl:NamedIndividual>
     <owl:NamedIndividual rdf:about="&savecandidatereq;homeAddress">
          <rdf:type rdf:resource="&CRM;HomeAddress" />
          <CRM:addressDescription rdf:datatype="&xsd;string">İzmir, Bornova</CRM:addressDescription>
     </owl:NamedIndividual>
     <owl:NamedIndividual rdf:about="&savecandidatereq;homePhone">
          <rdf:type rdf:resource="&CRM;HomePhone" />
          <CRM:phoneNumber rdf:datatype="&xsd;string">02326545566</CRM:phoneNumber>
     </owl:NamedIndividual>
     <owl:NamedIndividual rdf:about="&savecandidatereq;mobilePhone">
          <rdf:type rdf:resource="&CRM;MobilePhone" />
          <CRM:phoneNumber rdf:datatype="&xsd;string">05553214455</CRM:phoneNumber>
     </owl:NamedIndividual>
     <owl:NamedIndividual rdf:about="&savecandidatereq;workAddress">
          <rdf:type rdf:resource="&CRM;WorkAddress" />
          <CRM:addressDescription rdf:datatype="&xsd;string">Urla, İzmir</CRM:addressDescription>
     </owl:NamedIndividual>
     <owl:NamedIndividual rdf:about="&savecandidatereq;workPhone">
          <rdf:type rdf:resource="&CRM;WorkPhone" />
          <CRM:phoneNumber rdf:datatype="&xsd;string">02328521122</CRM:phoneNumber>
     </owl:NamedIndividual>
</rdf:RDF>

Request - SOAP Message:
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope
     xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
     xmlns:owl="http://www.w3.org/2002/07/owl#" xmlns:CRM="http://www.ege.edu.tr/ontologies/2011/11/CRM.owl#"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
     <soap:Body>
          <CreateCandidate xmlns="http://x.web.edu.ege">
```

```
                <candidate>
                     <name xmlns="http://x.web.edu.ege/xsd">Ali Durmaz</name>
                     <phone xmlns="http://x.web.edu.ege/xsd">05553214455</phone>
                     <address xmlns="http://x.web.edu.ege/xsd">Urla, İzmir</address>
                </candidate>
           </CreateCandidate>
     </soap:Body>
</soap:Envelope>

Selected web service endpoint: http://x.web.edu.ege::CreateCandidate
Make a web service call via Http by using translated SOAP message...
```