T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR BİLİMLERİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

WEB SERVİSİ İLE ONTOLOJİ YÖNETİMİ

(Yüksek Lisans Tezi)

Tezi Hazırlayan: **Eyyüphan ÖZDEMİR**

İstanbul, 2010

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR BİLİMLERİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

WEB SERVİSİ İLE ONTOLOJİ YÖNETİMİ

(Yüksek Lisans Tezi)

Tezi Hazırlayan:
Eyyüphan ÖZDEMİR
Öğrenci No:
08082001

Danışman:
Yrd.Doç.Dr. Zeynep ALTAN

İstanbul, 2010

YEMİN METNİ

Yüksek lisans tezi olarak sunduğum “Web Servisi ile Ontoloji Yönetimi” başlıklı bu çalışmanın, bilimsel ahlak ve geleneklere uygun şekilde tarafımdan yazıldığını, yararlandığım eserlerin tamamının kaynaklarda gösterildiğini ve çalışmamın içinde kullanıldıkları her yerde bunlara atıf yapıldığını belirtir ve bunu onurumla doğrularım. 06/09/2010

Aday:Eyyüphan Özdemir

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZ SAVUNMA SINAVI SONUÇ TUTANAĞI

Beykent Üniversitesi
Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Aşağıda tez adı belirtilen yüksek lisans öğrencisi **080820001** no'lu **Eyyüphan ÖZDEMİR**'in 21/10/2010 tarihinde yapılan tez savunma sınavı[1] sonucunda 50 dakika süreyle sunduğu ve savunduğu tezi hakkında[2] **oybirliğiyle Kabul** kararı verilmiştir.

Bilgilerinize saygılarımızla arz ederiz.

Anabilim Dalı : Bilgisayar Mühendisliği
Programı : Bilgisayar Mühendisliği
Tez Başlığı[3] : Web Servisi ile Ontoloji Yönetimi

Tez Sınav Jürisi	**Öğretim Üyesi**	**İmza**
Danışman	: Yard.Doç.Dr.Zeynep ALTAN	.....
Üye	: Prof.Dr.Yahya KARSLIGİL	.....
Üye	: Yard. Doç.Dr.Gökhan SİLAHTAROĞLU	.....

---

[1] Jüri üyeleri söz konusu tezin kendilerine teslim edildiği tarihten itibaren en geç bir ay içinde toplanarak öğrenciyi tez savunma sınavına alır. Belirlenen günde yapılamayan jüri toplantısı, katılanların hazırladığı bir tutanakla enstitü yönetimine bildirilir. Bu durumda jüri en geç onbeş gün içinde toplanarak adayı tez savunma sınavına alır. Tez savunma sınav süresi en az 45 dakikadır. Yüksek lisans tez savunma sınavı, tez çalışmasının sunulması ve bunu izleyen soru-yanıt bölümlerinden oluşur ve dinleyiciye açıktır. (Beykent Lisansüstü eğitim ve Öğretim Yönetmeliği-Madde30-3)

[2] Tez sınavının tamamlanmasından sonra jüri, tez hakkında "kabul", "düzeltme" veya "red" kararı verir. Jüri başkanı, jüri üyelerince imzalanmış sınav tutanağını, tez sınavını izleyen üç gün içinde ilgili enstitü yönetimine teslim eder. Tezi başarısız bulunan öğrencinin Enstitü ile ilişiği kesilir. Tezi hakkında düzeltme kararı verilen öğrenci en geç üç ay içinde gerekli düzeltmeleri yaparak ve yönetmelikte belirtilen usullere uygun olarak tezini aynı jüri önünde yeniden savunur. Bu savunma sınavında da tezi kabul edilmeyen öğrencinin enstitü ile ilişiği kesilir. (Beykent Lisansüstü eğitim ve Öğretim Yönetmeliği-Madde30-4)

[3] İleride doğabilecek aksaklıkların engellenmesi için tezin başlığının yazılması gerekmektedir.

WEB SERVİSİ İLE ONTOLOJİ YÖNETİMİ

Tezi Hazırlayan: Eyyüphan Özdemir

ÖZET

İnternet'in üçüncü aşaması olarak Web 3.0, eş deyişle Anlamsal Ağ çağını yaşıyoruz. Anlamsal Ağ, sayısal veri miktarının katlanarak büyüdüğü ve çeşitlendiği günümüzde, İnternet erişimine açık her türlü bilgiyi anlamsal bağlarla bir araya getirmeyi amaçlamaktadır.

Anlamsal Ağ, İnternet'in önceki aşamalarından farklı olarak, sözdizimsel içerik yerine anlamsal içerik oluşturma ve işleme hedefindedir. Anlamsal Ağ, verilerin anlamlarının ve diğer verilerle olan ilişkilerinin bilgisayarlar tarafından çözülebilmesi için yeni kavramlar ve teknolojiler barındırmaktadır.

Anlamsal Ağ, günümüzde, başta içerik araştırması, içerik yönetimi, özelleştirme, veri bütünleştirme, çalışma alanı modellemesi, gelişmiş arama, doğal dil arayüzü, köken takibi, metin madenciliği, anlamsal belirtim ve sosyal ağ konularında sayıları hızla artan web siteleri ve uygulamaları tarafından kullanılmaktadır.

Anlamsal Ağ'ın olanaklarından yararlanan web siteleri ve uygulamaları için, ileride kendisini daha çok duyuracak gereksinimlerden biri de anlamsal içeriğin yönetimi olacaktır. Web sitelerinin içeriklerinin büyük bir bölümünün kullanıcılar tarafından sağlandığı "Katılımcı Web" döneminden beri önemli bir konu olan içerik yönetimi, anlamsal içeriğin kullanıcılar tarafından belirlenmesi ya da genişletilmesi bakımından Anlamsal Ağ çağı için de önemli bir konudur.

Günümüzde İnternet uygulamalarının gün geçtikçe daha belirgin biçimde yöneldiği bir başka teknoloji ise web servisleridir. Web servisleri, XML tabanlı olmaları, HTTP gibi oldukça yaygın bir protokol kullanmaları, esnek, modüler ve

platform bağımsız olmalarından dolayı, bilgisayarlar arasında bilgi alışverişi konusunda standart duruma gelmiştir.

Bu tez kapsamında, Anlamsal Ağ'ın temel bileşenleri olan ontolojilerin, web servisi yoluyla uzaktan yönetiminin nasıl sağlanacağı incelenmiş ve örnek bir uygulama ile gösterilmiştir.

Anahtar Sözcükler: Anlamsal Ağ, Ontoloji, Web Servisleri, Ağ Ontoloji Dili.

ONTOLOGY MANAGEMENT USING WEB SERVICE

Presented by: Eyyüphan Özdemir

ABSTRACT

We are living the Semantic Web age as the third stage of Internet, Web 3.0. Semantic Web is intended to combine any accessable information by semantical connections in the age of digital data increasing day by day.

Semantic Web, unlike the previous stages of the Internet, intends to generate and process the semantic content instead of the syntactic content. Semantic Web includes new concepts and technologies so that computers can resolve meanings of data and their connections with each other.

Today Semantic Web is being used by rapidly increasing numbers of web applications for data management, customization and integration, workspace modeling, natural language interface, text mining, semantic specification and social web.

One of the future necessity for the web applications using Semantic Web is going to be managing and sharing of the semantic content. Content management has been an important subject until the “Participating Web” age, which means web site’s contents mostly created by the users. Content management is also important to identify semantic content and expand it for the semantic age.

At the present time one of the technologies which Internet applications are headed is web service. Web service technology has become standart because of being

XML based, using a quite common protocol such as HTTP, being flexible, modular, and platform independent.

In this thesis, it is examined how to implement remote management of ontologies, the main components of Semantic Web, via web services and an example application is shown.

Keywords: Semantic Web, Ontology, Web Services, Web Ontology Language.

TEŞEKKÜR

Tez çalışmam süresince ilgisi ve desteğinden dolayı değerli danışmanım Sayın Yrd.Doç.Dr. Zeynep ALTAN'a teşekkürlerimi sunarım.

Yüksek lisans öğrenimim boyunca bana yol gösteren Prof. Dr. M. Yahya KARSLIGİL, Prof. Dr. İlhami YAVUZ, Yrd.Doç.Dr. Gökhan SİLAHTAROĞLU ve diğer bilim insanlarına teşekkür ederim.

Hiçbir sorumu yanıtsız bırakmayan ve böylece birçok sorunu aşmama yardımcı olan HP Labratuvarları'ndan Ian DICKINSON'a teşekkür ederim.

Çalışma süresince desteklerini esirgemeyen aileme teşekkür ederim.

İÇİNDEKİLER

Sayfa No.

TABLOLAR LİSTESİ

Sayfa No.

ŞEKİLLER LİSTESİ

Sayfa No.

KISALTMALAR

API **:**Application Programming Interface

DCOM **:**Distributed Component Object Model

DTD **:**Document Type Definition

HTML **:**Hyper-Text Markup Language

HTTP **:**Hyper-Text Transfer Protocol

IDE **:**Integrated Development Environment

IIOP **:**Internet Inter-ORB Protocol

JAR **:**Java Archive

JRE **:**Java Runtime Environment

N3 **:**Notation3

OWL **:**Web Ontology Language

RDF **:**Resource Description Framework

RDF-S **:**RDF-Schema

RMI **:**Remote Method Invocation

RPC **:**Remote Procedure Call

SOA **:**Service-Oriented Architecture

SOAP **:**Simple Object Access Protocol

SPARQL **:**Sparql Protocol and RDF Query Language

UDDI **:**Universal Description Discovery and Integration

URI **:**Uniform Resource Identifier

URL **:**Uniform Resource Locator

W3C **:**World Wide Web Consortium

WAR **:**Web Application Archive

WSDL **:**Web Service Definition Language

WWW **:**World Wide Web

XML **:**Extensible Markup Language

XML-S **:**XML-Schema

1. GİRİŞ

Bu tez çalışması, Anlamsal Ağ'ın temel bileşeni olan ontolojilerin uzaktan yönetimi konusunu başarım, esneklik, modülerlik, standartlara ve çeşitli platformlara uyumluluk açısından ele almakta ve çözüm olarak ontoloji üzerindeki işlemleri gerçekleştiren bir web servisi ve bu web servisiyle etkileşen bir modülden oluşan bir sistemin anlatımı ve geliştirmesini içermektedir.

Öncelikle Anlamsal Ağ konusu ele alınacak, ardından ontoloji kavramı üzerinde durulacaktır. Sonrasında ise, World Wide Web Consortium'un (W3C) standart ontoloji geliştirme dili olarak belirlediği Web Ontology Language (OWL) dilinin temel kavramları ve özellikleri anlatılacak; bunlar örnek bir ontoloji üzerinde gösterilecektir. Bu bölümde, ontoloji uygulamaları konusunda yaygın biçimde kullanılan programlama arayüzlerinden Jena hakkında da bilgi verilecektir.

Sonraki bölümlerde, ontoloji ile ilgili tüm işlemlerin bir web servisi tarafından yapılması öngörüldüğü için, web servislerinin temel özellikleri ve yararları üzerinde durulacaktır.

Anlamsal Ağ, ontoloji, OWL ve Web servisleri konularının anlatımından sonra ise, ontolojilerin İnternet üzerinden yönetimi konusuna yönelik çalışmaların genel olarak neleri kapsaması, hangi özellikleri barındırması gerektiği belirlenecektir. Bu sayede, geliştireceğimiz çözümün neleri amaçladığı ortaya konacak; hangi araç ve teknolojileri kullanması ve sistem mimarisinin nasıl olması gerektiğine karar verilecektir.

Son bölümde ise çözümün bileşenleri adım adım geliştirilecektir.

Tez çalışması sonunda varılmak istenen hedefler şunlardır: Anlamsal Ağ, ontoloji, OWL ve Web servislerinin temel olarak anlatılması, ontolojilerin uzaktan yönetimine yönelik bir çözümün kapsamının, temel özelliklerinin ve kısıtlarının belirlenmesi ve belirlenen hedeflere uygun bir çözümün adım adım geliştirilmesi. Bu çözüm, sunucu tarafında çalışacak bir web servisi olan Ontoloji Yönetim Servisi ile bu servisi kullanan ve kullanıcı ile etkileşen bir modülü, Ontoloji Yönetim Modülünü içerecektir. Bu modül hem masaüstü uygulamalarda hem de bir web sayfasında

kullanılabilecek bir modül olacaktır. Bununla bağlantılı olarak hem istemci tarafında hem de sunucu tarafında erişilebilir olacaktır.

Çalışmada, Web 3.0 çağının başlıca aktörlerinden, ontolojiler, web servisleri ve Java applet'leri kullanılarak, bilgi paylaşımı ve yönetimi konusunda bir çözüme ulaşılmıştır.

2. ANLAMSAL AĞ

İnternet'in temellerini atan ve halen W3C başkanlığı yapan Tim Berners Lee, Anlamsal Ağ'ın temel amaçlarını aşağıdaki biçimlerde belirtmektedir:

"Web için bir hayalim var, öyle ki bilgisayarlar web üzerindeki bütün veriyi, içerikler, linkler ve insanlarla bilgisayarlar arasındaki bütün işlemler gibi, analiz etmeye muktedir olacaklar. Henüz ortaya çıkmamış olsa da, ortaya çıktığı zaman Semantic Web ticaretin günlük mekanizmaları, bürokrasi ve günlük yaşamlarımız birbiri ile konuşan makinalar tarafından yürütülecek. İnsanlığın asırlardır konuşup durduğu "akıllı ajanlar" nihayet gerçekleşecek." [1]

"İlk olarak web, daha fazla birlikteliğin sağlanacağı bir ortam haline gelecektir, ikinci olarak da işlemlerin bilgisayarlar tarafından yürütülmesi ile web daha anlaşılır olacaktır" [2]

Anlamsal Ağ, günümüzde İnternet ile ilgili sorunların çözümüne yönelik yeni bir yaklaşımdır. Bu yeni yaklaşımın ilgili olduğu başlıca uygulama alanlarını ise aşağıdaki gibi sıralayabiliriz: [3]

- Bilgi yönetimi,
- Doğal dil işleme,
- e-Ticaret,
- Zeki bilgi bütünleştirme,
- Bilgi elde edimi,
- Veritabanlarının bütünleştirimi,
- Biyo-bilişim,
- Eğitim.

Günümüzde, çok büyük boyutlara ulaşmış olan verilerle ilgili erişim, ilişkilendirme ve bütünleştirme sorunları yaşanmaktadır. Tüm bu sorunların

kökeninde verilerin sözdizimsel olarak tutulması yatmaktadır. Anlamsal Ağ ise İnternet ortamındaki verilerin sözdizimsel olarak değil anlamsal olarak oluşturulması, saklanması ve erişilmesi hedefindedir.

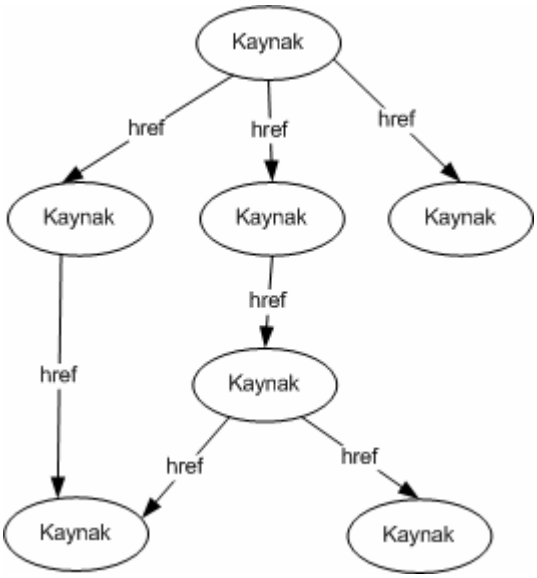
Sözdizimsel ağ için aşağıdaki işlemlerin gerçekleştirilmesi mümkün değildir. Bunlar aynı zamanda Anlamsal Ağ'ın başlıca hedefleridir: [4]

- Etki alanı bilgisi üzerinde çalışan karmaşık sorgulamalar,
- Veri depoları arasında belirli ilişkilere göre gezinmek,
- Web servislerini bulmak ve kullanmak,
- Web Temsilcilerine (Web Agent) görevler tayin edebilmek.

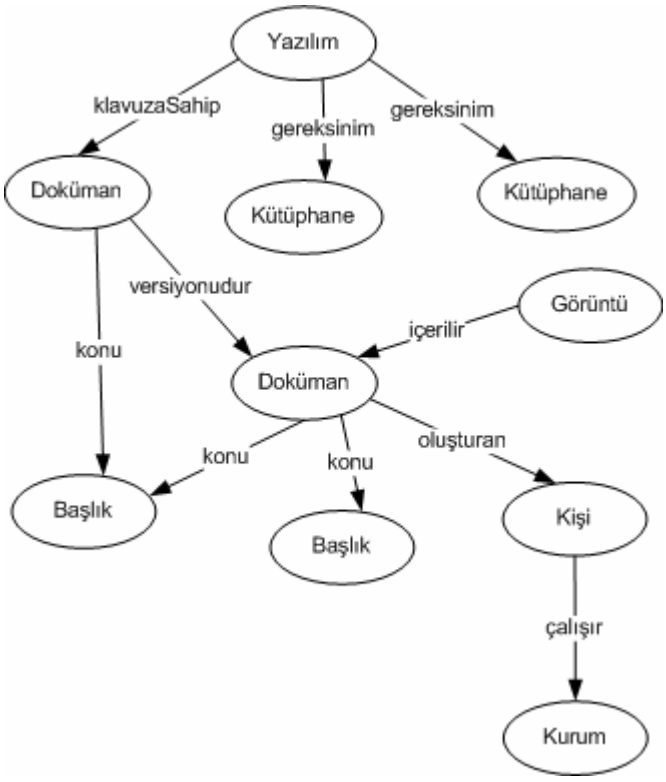
Tipik bir ağ sayfasında önemli olan verilerdir. Veriler belirli biçimlerde alt alta sıralanır. Önemli olan bu verilerin kullanıcının ekranına getirilmesi olduğu için verilerin anlamsal ilişkileri gözardı edilir. Aynı nedenden dolayı farklı sunucularda tutulan ilişkili verilere erişim ve bunlar üzerinde ortak bir işlem yapılması da son derece zordur. Çünkü verileri birbirine bağlayan anlamsal bağlar eksiktir.

Bir ağ sayfası içindeki veriler bilgisayarlar için değil, kullanıcılar için anlamlıdır. Bundan dolayı biçimlendirilmiş olmakla birlikte anlamsal bağlarla bağlanmamış verileri bilgisayarlar yorumlayamaz; yalnızca sözdizimsel olarak çözer ve kullanıcı önüne getirir.

Şekil 1.a'da Anlamsal Ağ öncesi web içeriğini oluşturan kaynakların Hyper-Text Markup Language (HTML) bağlantıları ile birbirine bağlanışı anlatılmaktadır. Bu bağlantılar hedef kaynak ile ilgili anlamsal bir bilgi vermez.



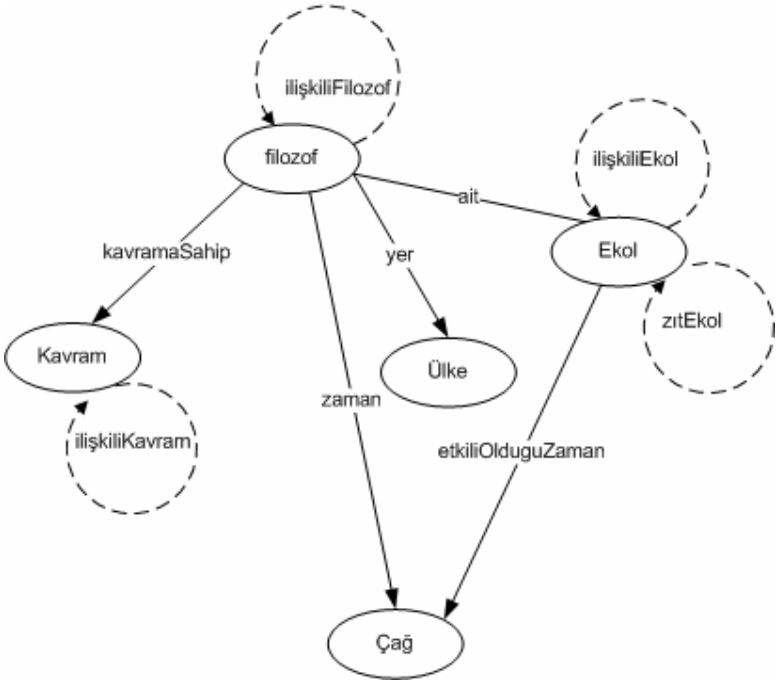
Şekil.1.a) Anlamsal Ağ öncesi web içeriği



Şekil.1.b) Anlamsal Ağ'da web içeriği

Anlamsal Ağ ise, Şekil 1.b'de gösterildiği gibi, verilerin içeriği ya da biçiminden çok anlamı, bağlamı ve diğer verilerle olan ilişkisi ile ilgilenir. Böylece

veriler hem tek bir web sitesi için, hem de tüm İnternet için anlamsal biçimde saklanır.



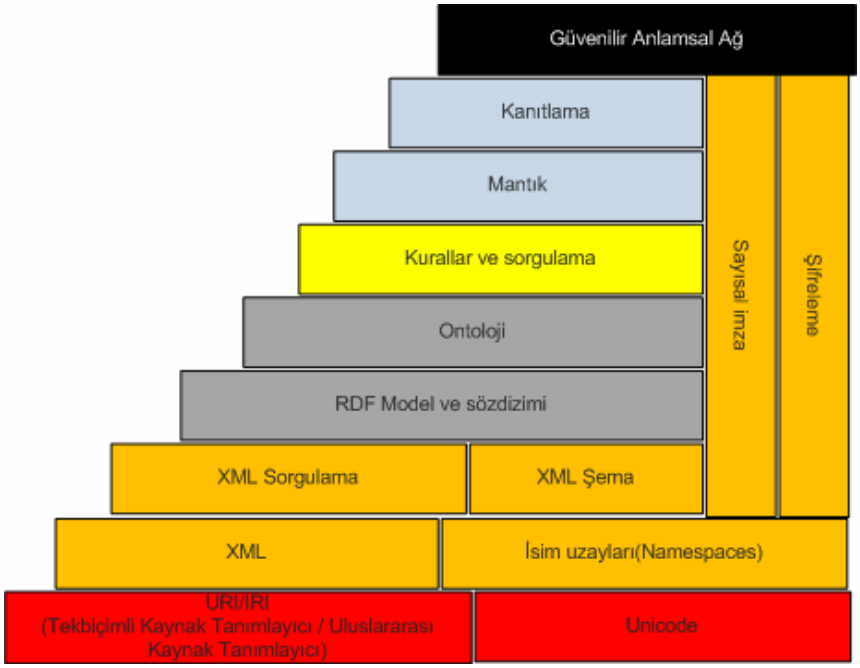
Şekil.2. Anlamsal Ağ'daki verilerin ilişkisine dair bir örnek

Şekil 2'de Felsefe konulu bir Anlamsal Ağ web uygulamasının içeriğinin anlamsal bağlarla düzenlenişi gösterilmiştir.

Anlamsal Ağ sayesinde veriler yalnızca kullanıcılar için anlamlı olmayıp, bilgisayarlar için de anlamlı olmaktadır. Bu köklü çözüm verilerle ilgili olarak erişim, ilişkilendirme ve bütünleştirme işlemlerini belirgin biçimde kolaylaştırmaktadır. Anlamsal Ağ, İnternet'in en önemli bileşeni olan arama motorlarından, web servislerine dek tüm İnternet'i derinlemesine değiştiren bir yaklaşımdır.

Anlamsal Ağ, köklü bir yaklaşım farklılığı getirmekle birlikte önceki web yaklaşımlarının bir uzantısı da sayılabilir. Anlamsal Ağ öncesinde de daha önce ifade edilen sorunların çözümüne ilişkin olarak Extensible Markup Language (XML) ve XML Schema (XML-S) gibi yenilikçi standartlar geliştirilmişse de bunlar daha çok, verilerin yapısıyla ilgilidir. Bununla birlikte temel sorun verilerin anlamının bilgisayarlar tarafından çözülebilmesi, ilişkilendirilmesi, bütünleştirilmesi ve kendiliğinden güncellenmesidir. Anlamsal Ağ bu temel sorunların çözümü için bir

çok ilke, çalışma grubu, yardımcı teknoloji, notasyon ve dilden yararlanan sıradüzensel bir çerçevedir.



Şekil.3. Anlamsal Ağ sıradüzeni

Şekil 3'te görüldüğü gibi Anlamsal Ağ, Extended Markup Language (XML), XML Schema, Resource Description Framework (RDF) , RDF Schema ve OWL bileşenlerini içerir. Bu bileşenler aşağıdaki gibi özetlenebilir: [5]

XML, yapılandırılmış belgeler için temel sözdizimsel kuralları belirtir. Ama daha önce belirtildiği gibi içeriğin anlamı ile ilgilenmez ve bu konuda herhangi bir kısıtlaması yoktur. XML Schema ise XML belgelerinin yapı ve içeriğini düzenlemeye yarayan bir dildir. RDF nesnelere (resources) ve bu nesnelerin nasıl ilişkili olduğuna işaret eden bir veri modelidir. RDF temelli model, XML sözdiziminde ifade edilebilir. RDF Schema, RDF kaynaklarının özelliklerini ve sınıflarını ifade etmeye yarayan sözcükler bütünü ve bunların genelleştirme sıradüzenleri için bir anlambiliminden oluşur. OWL, standart ontoloji geliştirme dili olarak verilerin özelliklerini, sınıflarını, kurallarını ve ilişkilerini betimlemek için kullanılan gelişmiş bir veri modelidir. [6]

3. ONTOLOJİ

Ontolojiler, Anlamsal Ağ'ın temel bileşenidir ve verilere anlamsal yaklaşımın merkezi kavramıdır.

Bilişim dünyasında yaklaşık 15 yıldır kullanılan bu terim felsefenin alt dallarından biri olan varlıkbilim ile eş anlamlıdır. Varlıkbilimin konusu ise varlıkların sıradüzeni, sınıflandırılması, tanımları ve ilişkileridir. Terim, bilişim dünyasında da hemen hemen aynı anlamda kullanılır.

Tim Berners Lee'nin tanımı şu biçimdedir:

"Web içeriğinin bilgisayarlar için anlamlı olan formu"

Gruber ise ontolojinin tanımını şöyle yapar:

"Ontoloji, paylaşılan bir kavramsallaştırmanın biçimsel ve net belirtimidir" [7]

Bu tanımdaki "kavramsallaştırma", "net bir belirtim", "biçimsel" ve "paylaşılan" sözcüklerinin her birinin ayrı ayrı tanımları ise aşağıdaki biçimde verilebilir: [8]

"Kavramsallaştırma", insanların dünyadaki varlıklar üzerine nasıl düşündüklerinin soyut bir modelini ifade eder. Bu soyut model genellikle özel bir konu alanı ile sınırlandırılmıştır.

"Net bir belirtim", soyut modeldeki kavramlara ve ilişkilere net isimler verildiği ve net tanımlar yapıldığı anlamına gelmektedir. Bir kavramın ya da ilişkinin tanımı, o terimin anlamının ifade edilmesidir. Bir başka deyişle, bir terimin diğer terimlerle ilişkisinin nasıl olacağını belirtir.

"Biçimsel", anlam tanımının biçimsel bir dille temsil edildiğini ve böylece tanım üzerindeki belirsizliklerin, farklı anlam çıkarma olasılıklarının ortadan kaldırıldığını ifade etmektedir. Bundan dolayı biçimsel temsil, kendiliğinden çıkarım yapma imkanını sağlamaktadır.

“Paylaşılan” ise, ontolojilerin farklı uygulamalar ve topluluklar arasında yeniden kullanımı amaçladıklarını ve desteklediklerini ifade etmektedir.

Anlamsal Ağ’ın verilere, anlamlarına göre yaklaşmaları nedeni ile, verilerin türleri, sıradüzen içindeki yerleri, tanımları ve diğer verilerle olan ilişkilerinin belirlenmesi Anlamsal Ağ’ın temel işlevidir.

Anlamsal Ağ yaklaşımında bilginin temsili, ontolojiler ile gerçekleştirilir. Bir ontoloji, bir çalışma alanındaki her şeyi bir varlık olarak kavramanın ve bu alandaki tüm varlıkları bir bütün olarak kavramsallaştırmanın yoludur.

Ontoloji kavramının ne olduğunu anlamak için bu kavramın veritabanları, nesne modelleri, XML Schema ve iş kuralları ile karşılaştırmakta yarar vardır: [9]

Ontolojiler, veritabanlarına benzerler; çünkü çalışma zamanında uygulamalar tarafından sorgulanırlar. Ontolojiler, geleneksel veritabanlarına benzemezler; çünkü ontolojilerde ilişkiler birinci sınıf yapıtaşlarıdır.

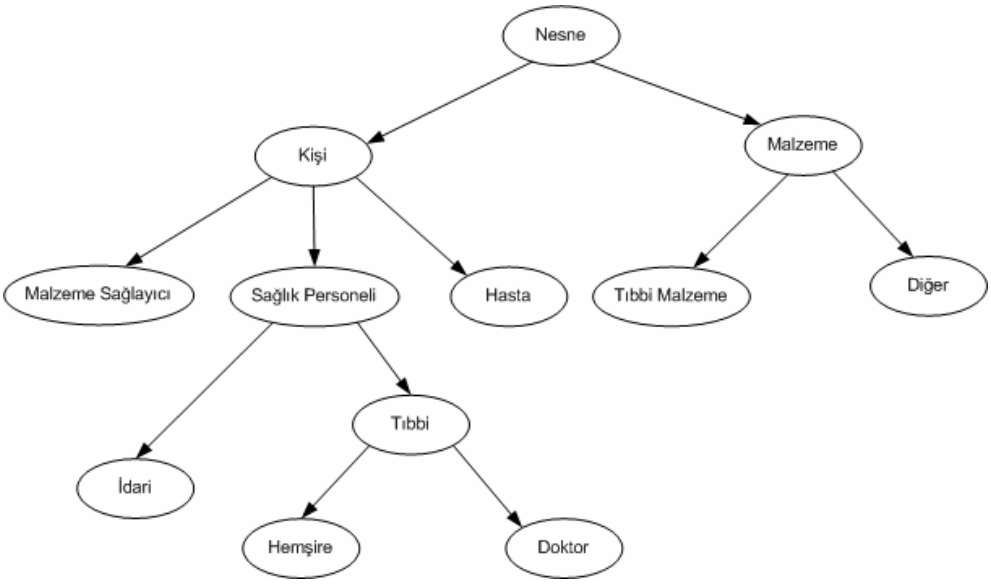
Ontolojiler, nesne modellerine benzerler; çünkü sınıfları ve niteliklerini tanımlarlar. Ontolojiler, nesne modellerine benzemezler; çünkü ontolojiler küme-temelli ve dinamiktirler.

Ontolojiler, XML Schema’ya benzerler; çünkü web için doğal yapılardır. (XML biçiminde serileştirilebilirler) Ontolojiler, XML Schema’ya benzemezler; çünkü ontolojiler çizge veri yapısı tipindedir, ağaç veri yapısı tipinde değillerdir.

Ontolojiler, iş kurallarına benzerler; çünkü olay-tabanlı davranışları kodlarlar. Ontolojiler, iş kurallarına benzemezler; çünkü ontolojiler kuralları ilk savları kullanarak düzenlerler.

Ontolojiler sayesinde çalışma alanındaki kavram, nesne ve ilişkiler tanımlanarak bu alandaki bilgiler modellenir.

Örneğin bir sağlık kurumu ile ilgili tüm bilgiler Şekil 4’te gösterilen biçimde modellenebilir:



Şekil.4. Örnek bir ontolojide sınıf sıradüzeni

Ontolojiler, varlıkların sıradüzenini belirttiğinden sınıfların türetilmesini de içerir. Bundan dolayı yeniden kullanılabilirlik ontolojilerin temel yararları arasındadır.

Ontolojiler sayesinde aynı çalışma alanında ortak bir model kullanılabilir; aynı alandaki farklı modeller birleştirilebilir, kesiştirilebilir ve aralarındaki farklar belirlenebilir.

Ontolojilerin Anlamsal Ağ için başlıca yararlarından biri bilgilere daha akıllı erişimi gerçekleştirmesidir. Anlamsal Ağ yaklaşımına uygun olarak tasarlanmış arama motorlarının, belirli bir çalışma alanındaki modeller üzerinde işlem yapması ile daha anlamlı sonuçlar üretilecektir.

Ontolojilerin içerdiği kavramlar aşağıda belirtilmektedir:

- Sınıflar (classes) bilgi alanlarındaki genel kavramların tanımlarıdır.
- Örnekler (instances) sınıfların özelliklerini taşıyan öğelerdir.
- İlişkiler (relationships) öğeler arasındaki bağlantıları tanımlar.
- Özellikler (properties) sınıfların ve örneklerin özellikleri tanımlanır.
- Fonksiyonlar (functions) öğelerle ilgili fonksiyonlardır.

- Kısıtlar (constraints) ontolojide tanımlanan nesne ve sınıflarla ilgili kısıtlardır.

- Kurallar (rules) ilişkiler ve özellikler arası kısıtları, alanların alabileceği değerleri (range) vb. tanımları içerir.

3.1. Web Ontology Language (OWL)

Standart ontoloji dilleri arasında, W3C tarafından sunulan en son dil OWL'dir. OWL, Anlamsal Ağ bilgisi için genel bir yol sağlar ve XML, XML-S, RDF ve RDF-S gibi web dillerinin bir devamı niteliğindedir. OWL belgeleri, XML tabanlı olarak saklanır ve işlenir. Bundan dolayı platformlara uyumluluk konusunda bir sorun yaratmaz.

OWL, RDF'deki kavramları içerdiği gibi ona yeni özellikler eklemiş ve böylece Anlamsal Ağ'ın temel teknolojisi haline gelmiştir.

W3C, RDF ve XML'e göre daha gelişmiş bir ontoloji diline gereksinim duyulduğunu aşağıdaki biçimde dile getirmiştir:

"Farklı kurumlar arasında veri değişimi veya araması yapan uygulamalar için ontolojiler çok önemlidir. Her ne kadar XML, Document Type Definition (DTD) ve XML Şemalar daha önceden anlaşılmış veri yapıları üzerinde veri değişimini mümkün kılsa da, bu dillerde anlamsallık olmadığından yeni bir XML kelime haznesi tanımlandığında bunun bilgisayar tarafından algılanıp kullanılması mümkün değildir. Bu sorun basit seviyede anlamsallığı ifade etmemize olanak sağlayan RDF ve RDF Şemalar ile belli bir miktarda çözülmüştür, ancak çok sayıda otonom olarak gelişen ve yönetilen şemalar arasında birlikte çalışılabilirliği sağlamak daha zengin bir anlamsallık gerektirir. Örneğin RDF Şemalarda bir kişinin erkek veya kadın olabileceğini, bir babanın bir kişi ve bir erkek olabileceğini tanımlayabiliriz; ancak bir babanın kadın olamayacağını ifade etme olanağımız yoktur" [10].

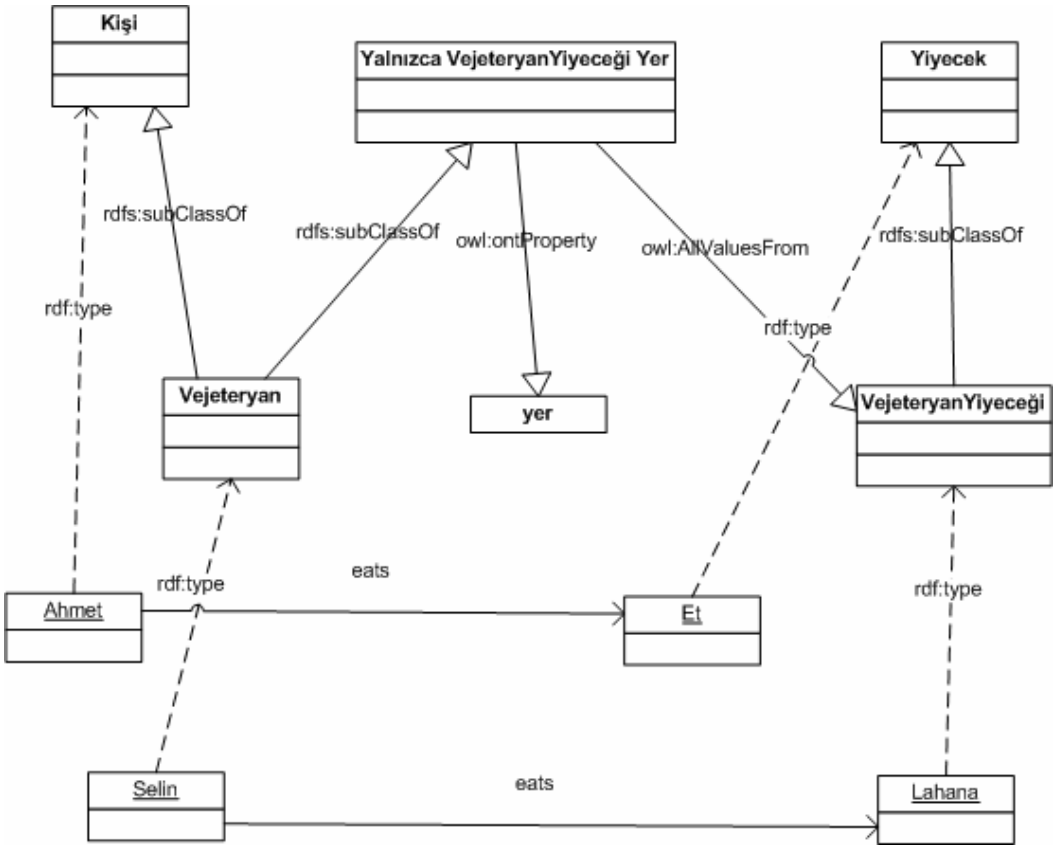
OWL'nin temel bileşenleri sınıflar (classes), bireyler (individuals) ve özelliklerdir (attributes).

Bireyler, çalışma alanında içerilen herhangi bir varlıktır.

Özellikler ise bireyler arasındaki ilişkileri belirlememizi sağlar. Özellikler veri özellikleri ve nesne özellikleri olarak ikiye ayrılır. Diğer bir deyişle bir bireyin bir özelliği ya onu tanımlayan bir veri ile ilgilidir ya da bir başka nesne ile ilgilidir. İlk durumda bu özellik veri özelliği, ikinci durumda ise nesne tipi özelliktir. Örneğin herhangi bir A bireyinin "name" özelliği metin tipinde, dolayısıyla veri tipindedir. Aynı nesnenin "isParentOf" gibi değer olarak bir başka nesneyi alan özellikleri ise nesne tipi özelliktir.

Sınıflar ise bireyleri kapsayan kümelerdir. Sınıflar türetilebilir, ilişkilendirilebilir ve onlar için kimi sınırlamalar belirtilebilir. OWL'de sınıflar ayrık ya da alt küme olarak tanımlanabilir. Bir sınıf diğerinin eşiti olarak tanımlanabileceği gibi, "ve", "veya", "içerir" gibi işleçler kullanılarak kurala dayalı mantıksal sınıflar da elde edilebilir.

OWL'nin güçlü yanı nesneler arasında mantıksal ilişkilerin kurulmasını sağlamasıdır. Bu, kurallar ve sınırlamalar ile yapılır. Örneğin, özellikler için kaynak (domain) ve hedef (range) belirtilerek yalnızca anlamlı bilgilerin girilmesi, dolayısıyla modelin tutarlılığı sağlanır. Buna göre yalnızca A sınıfı ile B sınıfının bireyleri için anlamlı olan özellikler tanımlanabilir. Örneğin ulaşım araçlarını konu edinen bir modelde "motor tipi" gibi bir özellik, yalnızca "Araç" isimli bir sınıfa ait bir varlık ile "Motor" isimli bir sınıfa ait bir varlığı ilişkilendirmek için kullanılabilir. Diğer bir deyişle "motor tipi" özelliğinin kaynağı (domain) "Araç" sınıfı; hedefi ise (range) "Motor" sınıfı olacaktır. Veri tipindeki özellikler için de hedefler tanımlanabilir. Veri tipindeki özellikler hedef , verinin tipi (metin, sayı, tarih vb.) olacaktır.



Şekil.5. OWL sınırlamalarını örnekleyen bir model

Örneğin, Şekil 5'te gösterilen OWL modelinde [11] aşağıdaki sınıflar, alt sınıflar, özellik ve sınırlamalar içerilmektedir:

- "Kişi" sınıfı ve onun "Vejeteryan" alt sınıfı,
- "Yiyecek" sınıfı ve onun "VejeteryanYiyeceği" alt sınıfı,
- "Kişi" sınıfına ait "Ahmet" varlığı,
- "Vejeteryan" sınıfına ait "Selin" varlığı,
- "Yiyecek" sınıfına ait "Et" varlığı,
- "VejeteryanYiyeceği" sınıfına ait "Lahana" varlığı,
- "Vejeteryan" sınıfı ile "VejeteryanYiyeceği" sınıfları arasındaki "allValuesFrom" tipindeki "yer" özelliği.

Bu modelde, “Vejeteryan” sınıfının “yer” özelliği, yalnızca “VejeteryanYiyeceği” sınıfının bir varlığı ile ilişkili olacak biçimde sınırlandırılmıştır.

OWL, Anlamsal Ağ sıradüzeninindeki her bileşen gibi Uniform Resource Identifier (URI) isimlendirmesini destekler. Bu, bir ontolojideki her varlığın bir kimliği diğer bir deyişle URI bilgisi olduğu anlamına gelir. OWL, bu sıradüzende yer alan XML biçeminde yapılandırılır ve saklanır. OWL, bu sıradüzendeki bir alt katman olan RDF modelini de destekler. Örneğin, URI belirtimi RDF sözdizimine göre yapılır.

3.2. Örnek bir Ontoloji Tasarımı

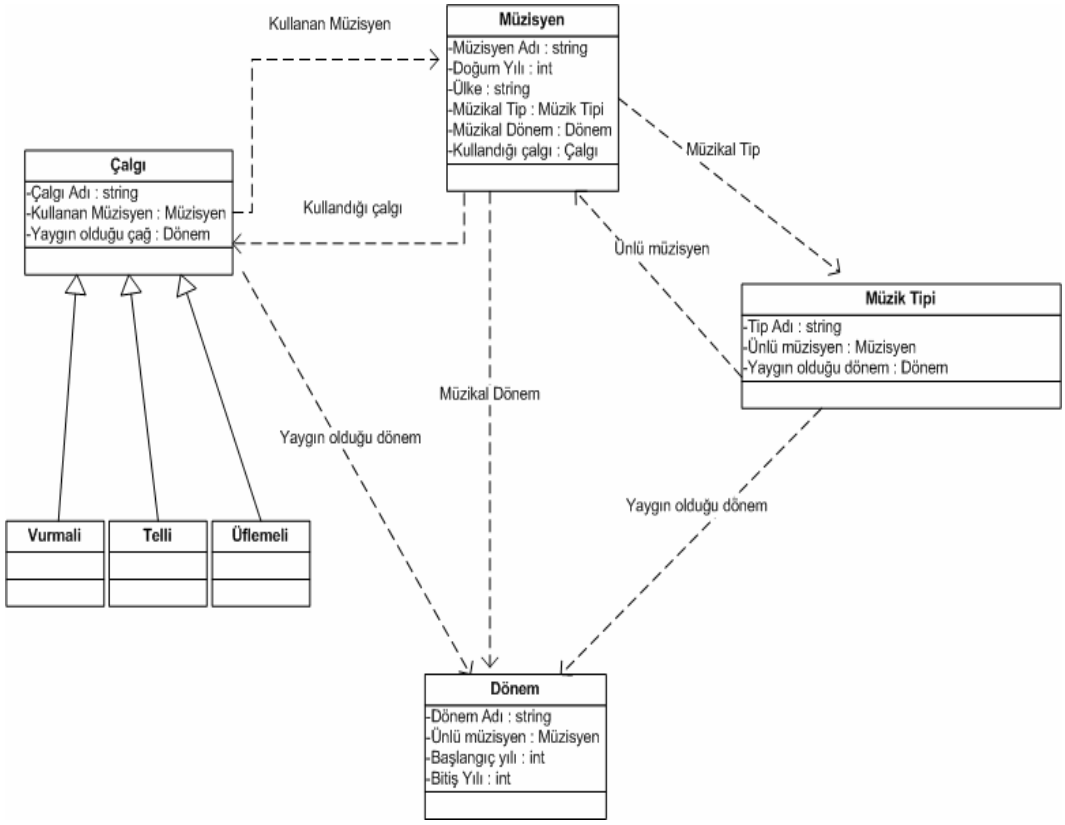
Bir ontoloji oluşturmak için öncelikle bir çalışma alanı seçmek ve bu alandaki sınıfları belirlemek gerekir.

Örneğin, çalışma alanı olarak Müzik konusu seçildiğinde bu alandaki varlıklar olarak müzisyenler, enstrümanlar, müzik tipleri ve müzikal dönemler alınabilir. Bunlar örnek ontolojinin sınıfları olarak düşünülebilir. Bu sınıflardan bazıları ise alt sınıflara ayrılmış olabilir. Örneğin “Enstrümanlar” sınıfı, “Telli”, “Vurmalı”, “Üflemeli” gibi alt sınıflara ayrılabilir.

Sonraki adım olarak bu sınıfa ait bireylerin ne gibi özellikleri olabileceği düşünülebilir: Müzisyenler için “Ad”, “Doğum Yılı”, “Ülke”, “Müzikal tip” ve “Ait olduğu müzikal dönem”, “Kullandığı enstrüman” gibi bir müzisyeni betimleyen temel özellikler düşünülebilir. Enstrümanlar için “Kullanan müzisyen”, “En çok kullanıldığı müzikal dönem” özelliklerini; “Müzikal dönem” sınıfı için “İçerdiği müzisyen”, “Başlangıç yılı”, “Bitiş yılı” özelliklerini; “Müzikal tip” sınıfı için “İçerdiği müzisyen” ve “En yakın müzikal dönem” gibi özellikler düşünülebilir. Bu özelliklerin kimisi (örneğin “Doğum yılı”, “Ad” vb.) veri tipi kimisi ise (“Kullanan müzisyen”, “İçerdiği müzisyen” vb.) nesne tipi özelliktir. Veri özellikleri için verinin tipi, nesne tipi özellikler için ise kaynak sınıfı ve hedef sınıfı belirlenmelidir. Örneğin “İcra ettiği müzikal tip” özelliği için kaynak sınıf “Müzisyen”, hedef sınıf ise “Müzikal tip” olacaktır.

Sınıflar ve özellikler belirlendikten sonra sınıflara ait bireylerin girişi yapılabilir. Örneğin, “Muzisyen” sınıfı için “Bach” ve “Beethoven” bireyleri eklenebilir.

Şekil 6, bu çalışmada örneklenen modelin sınıflarını, özelliklerini ve bireylerini göstermektedir:



Şekil.6. Örnek bir ontoloji modeli

Örneklenen model, bir ontoloji düzenleyicisi ile oluşturulup kaydedildiğinde “.owl” uzantılı bir dosya ortaya çıkacaktır.Bu dosya XML tabanlı bir metin dosyasıdır ve RDF/XML-Abbrev biçimindeki içeriği Ek-1’de gösterilmiştir.

Bu OWL dosyasının içeriği incelendiğinde ontoloji, sınıf, altsınıf, özellik ve bireylerin aşağıdaki şekilde tanımlandıkları görülebilir:

Ontoloji tanımı:

```
      <owl:Ontology
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl"/
>
```

Sınıf tanımı:

```
      <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#M
uzisyen"/>
```

Alt sınıf tanımı:

```
        <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#V
urmali">
          <rdfs:subClassOf
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Calgi"/>
        </owl:Class>
```

Veri tipi özellik tanımı:

```
        <owl:DatatypeProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#B
itisYili">
      <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Donem"/>
      <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
      </owl:DatatypeProperty>
```

Nesne tipi özellik tanımı:

```
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#K
ullandigiCalgi">
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Calgi"/>
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
        </owl:ObjectProperty>
```

Birey tanımı:

```
        <j.0:Muzisyen
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#B
ach">
          <j.0:Ad
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Johann Sebastian Bach</j.0:Ad>
          <j.0:Muzikal_Donemi>
            <j.0:Muzikal_Donem
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#B
arok"/>
          </j.0:Muzikal_Donemi>
        </j.0:Muzisyen>
```

3.3. Java için Anlamsal Ağ Çerçevesi:Jena

Jena, W3C tarafından standart kabul edilen Kaynak Tanımlama Çerçevesi (RDF, RDF-S), Ağ Ontoloji Dili (OWL) ve Ontoloji Sorgulama Dili (Sparql Protocol and RDF Query Language -SPARQL) için programlama ortamıdır. Jena Anlamsal Ağ uygulamaları geliştirmek için kullanılır ve açık kaynak kodludur.

Jena çalışma ortamı aşağıdaki bileşenleri içerir: [12]

- RDF Application Programming Interface (API),
- RDF, RDF/XML, Notation3 (N3) ve N-Triples biçiminde RDF okuma ve yazma,
- OWL API,
- Bellek içi ve kalıcı saklama,
- SPARQL sorgulama motoru.

Jena sınıfları RDF, RDF-S, ve OWL kavramlarını, bir sıradüzen içinde kapsar. Salt RDF ile ilgili işlemler yapılacağı gibi RDF sınıflarından türetilmiş sınıflar kullanılarak ontoloji işlemleri de yapılabilir. Örneğin Model sınıfı RDF modeli ile ilgili işlevleri kapsar; ondan türetilmiş OntModel sınıfı ise ontoloji modelleri ile çalışmamıza olanak sağlar.

Jena işlevleri, üzerinde çalıştığı dosyaların biçiminden bağımsız olarak işler. Bununla birlikte model, istenilen biçimde kaydedilebilir.

Jena ile ilgili çalışmalara HP Anlamsal Ağ Çalışma Grubu [1] liderlik etse de açık kaynak kodlu bir çalışma ortamı olduğu için genişletilebilirdir.

[1] http://www.hpl.hp.com/semweb/

4. WEB SERVİSLERİ

Bilişim dünyasının başlıca sorunlarından biri bilgisayarların haberleşmesi ve uygulamaların bütünleştirilmesi olmuştur. Remote Procedure Call (RPC), Distributed Component Object Model (DCOM), Internet Inter-ORB Protocol (IIOP) ve Java Remote Method Invocation (RMI) bu sorunun çözümüne yönelik olarak geliştirilmiştir. Fakat bu çözümlerden hiçbiri bilgisayarların iletişimi konusunda yeterli olamamıştır.

Web servisleri ise ortaya çıktığı 2000 yılından bu yana İnternet'i şekillendiren birçok kuruluş tarafından desteklenmiş ve sonuçta uygulama bütünleştirme konusunda bir standart ortaya çıkmıştır.

W3C, web servislerinin web ortamında ve kurumsal yazılımlarda kullanılan mesaj tabanlı bir tasarıma işaret ettiğini ve Hyper-Text Transfer Protocol (Hyper-Text Transfer Protocol), XML, Simple Object Access Protocol (SOAP), Web Service Definition Language (WSDL), SPARQL gibi teknolojiler üzerine kurulduğunu belirtir. [13]

Web servisleri için, ağ üzerinde birer Uniform Resource Locator (URL) ile tanımlanan ortamdan bağımsız program parçaları da denilebilir. Günümüzde sistemlerin birbirleriyle haberleşmesini gerçekleştiren en yaygın teknoloji web servisleridir. Web servislerinin bu denli kabul görmesinde aşağıdaki nedenler etkili olmuştur:

- Standart veri formatı olan XML'i kullanması,
- En yaygın ağ erişim protokolü olan HTTP üzerinden işlemesi,
- DCOM, RMI ve IIOP gibi nesne modeline özel erişim protokoküne gereksinim duymaması,
- Mevcut ağ mimarisinde herhangi bir değişikliğe gereksinim duyulmaması,

- Taşıdığı mesajları HTTP üzerinden aktardığı ve bu mesajlar XML biçimli basit metin dosyaları olduğu için, web servisi iletişiminin ateş duvarlarına (firewall) takılmaması,

- İşletim sistemlerinden bağımsız olması,

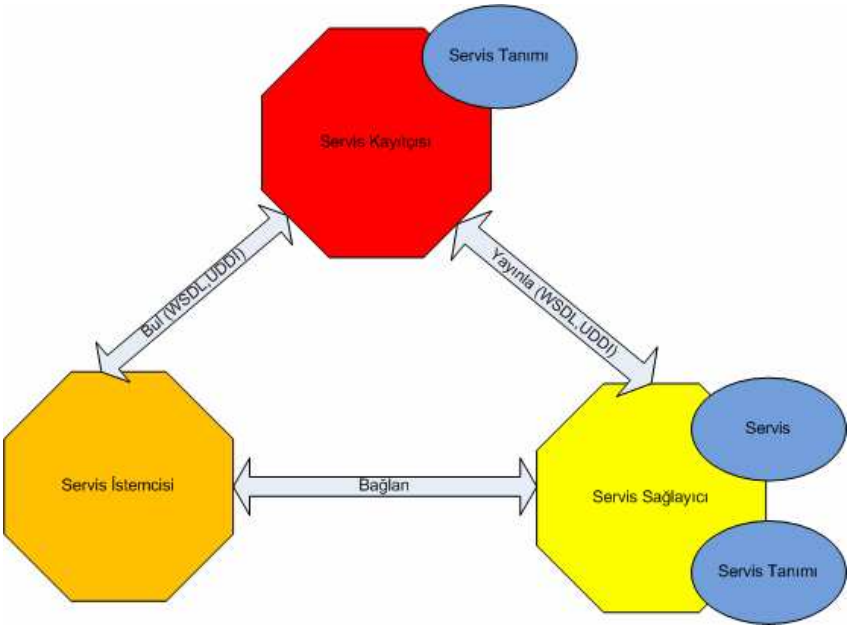
- Programlama dillerinden bağımsız olması.

Web servislerinin işleyişinde üç temel birim sözkonusudur.Bunlar servis sağlayıcısı, servis istemcisi ve servis kayıt birimi olarak sınıflandırılır.

Servis sağlayıcısı, istemcilerin sağlayıcıda bulunan servislere erişimini sağlar. Aynı zamanda kendi sitesinde bulunan web servislerinin tanımlarını servis kayıt birimine kaydederek bu servislerin nasıl çağrılacağını belirtir.

Servis istemcileri ise, servis sağlayıcısında bulunan web servislerini çağırarak kullanan istemci uygulamalardır. Web servislerini nasıl çağıracaklarını, ilgili parametreleri servis kayıt biriminden elde eder.

Servis kayıt birimi ise, servisle ilgili tanım bilgilerini servis sağlayıcısından alıp kaydeder ve istek yapıldığında servis istemcisine iletir.



Şekil.7. Web servis modeli.

Şekil 7 Web servisi teknolojisinin üç temel birimi arasındaki ilişkiyi betimlemektedir.

Web servisleri açık İnternet standartlarına dayanır. Bu standartlar şunlardır: SOAP, Web Service Definition Language (WSDL), ve Universal Description Discovery and Integration (UDDI).

4.1. Simple Object Access Protocol (SOAP)

SOAP, uygulama bütünleştirmesi için geliştirilmiş, XML biçimindeki verilerin iletimini sağlayan bir protokoldür. Bir SOAP XML belgesi, SOAP mesajı ya da SOAP zarfı olarak adlandırılır. Bu mesaj çeşitli ağ protokolleri ile gönderilebilir. Bununla birlikte SOAP mesajlarını iletmenin en yaygın yolu HTTP protokolüdür. SOAP mesajının gövdesi, çağrılacak metot ile bu metota aktarılacak parametre bilgilerini içerir.

SOAP belgeleri aslında Web servis istemcisi ile web servis sağlayıcısı arasındaki iletişimi sağlar. Servis istemcisi bir SOAP istek mesajı gönderir, bu istek mesajı SOAP sunucusundaki uç noktalardan birisine iletilir, SOAP sunucusu ilgili servisi çağırdıktan sonra SOAP yanıt mesajını hazırlar ve bu mesaj istemciye yanıt olarak iletilir.

Aşağıda bir SOAP istek ve yanıt mesajı örneği bulunabilir. [14]

SOAP istek mesajı:

```
      POST /InStock HTTP/1.1
Host: www.example.org
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPrice>
    <m:StockName>IBM</m:StockName>
  </m:GetStockPrice>
</soap:Body>
</soap:Envelope>
```

SOAP yanıt mesajı:

```
      HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">

<soap:Body xmlns:m="http://www.example.org/stock">
  <m:GetStockPriceResponse>
    <m:Price>34.5</m:Price>
  </m:GetStockPriceResponse>
</soap:Body>

</soap:Envelope>
```

Yukarıdaki SOAP mesajlaşması incelendiğinde istemcinin "StockName" parametresine "IBM" değerini geçirerek, "GetStockPrice" metoduna çağrı yaptığı anlaşılır. Servis sağlayıcısı ise yanıt olarak, "34.5" değerinde olan "Price" parametresini döndürmüştür.

4.2. Web Service Definition Language (WSDL)

Bir uygulamanın bir web servisini kullanması için, hangi protokolün kullanılacağını, web servisinin metot ve parametre tanımlarını ve web servis sağlayıcısının adresini bilmesi gerekir. Bu bilgiler belirli bir biçimde tutulmalıdır, aksi takdirde web servislerinin kulllanımı standart olmayacaktır.

WSDL, web servisi ile ilgili bilgilerin ortak bir biçimde tutulmasını sağlayan bir araçtır. Bir WSDL belgesi, WSDL Schema sözdizimine uygun bir XML belgesidir.

WSDL bilgisi, web servislerinin kullanımındaki temel aşamaların tümünde kullanılır. Bunlar yayınlama (publish), bulma (find) ve bağlama'dır (bind). Web servis sağlayıcısı, servisin tanımını bir veya daha fazla servis kayıt birimine kaydeder, servisin tanımı bu birimlerce yayınlanır, istemciler ise servisleri bu birimler sayesinde bulur ve onlara bağlanır.

Bir WSDL belgesinde üç temel bilgi bulunur:

- Servis ne amaçla geliştirilmiştir ve servisin sağladığı metotlar nelerdir?
- Servise erişim için kullanılacak protokoller ve veri formatları nelerdir?
- Servis adresi (URL) nedir?

Bir WSDL belgesinin temel bileşenleri şöyle özetlenebilir:

- <Types>:Web servisi tarafından kullanılan tipleri belirtir.
- <Messages>:Web servisi tarafından kullanılan mesajları belirtir.
- <PortType>: Web servis tarafından gerçekleştirilen işlemleri belirtir.
- <Binding>: Web servisinin kullandığı iletişim protokollerini gösterir.

Aşağıda örnek bir WSDL içeriği verilmektedir: [15]

```
<definitions name="HelloService"
targetNamespace="http://www.examples.com/wsdl/HelloService.wsdl"
         xmlns="http://schemas.xmlsoap.org/wsdl/"
         xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
         xmlns:tns="http://www.examples.com/wsdl/HelloService.wsdl"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema">

         <message name="SayHelloRequest">
            <part name="firstName" type="xsd:string"/>
         </message>
         <message name="SayHelloResponse">
            <part name="greeting" type="xsd:string"/>
         </message>

         <portType name="Hello_PortType">
            <operation name="sayHello">
               <input message="tns:SayHelloRequest"/>
               <output message="tns:SayHelloResponse"/>
            </operation>
         </portType>

         <binding name="Hello_Binding" type="tns:Hello_PortType">
         <soap:binding style="rpc"
            transport="http://schemas.xmlsoap.org/soap/http"/>
         <operation name="sayHello">
            <soap:operation soapAction="sayHello"/>
            <input>
               <soap:body
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
```

```
                  namespace="urn:examples:helloservice"
                  use="encoded"/>
            </input>
            <output>
               <soap:body

encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                  namespace="urn:examples:helloservice"
                  use="encoded"/>
            </output>
         </operation>
         </binding>

         <service name="Hello_Service">
            <documentation>WSDL File for
HelloService</documentation>
            <port binding="tns:Hello_Binding" name="Hello_Port">
               <soap:address
                  location="http://www.examples.com/SayHello/">
            </port>
         </service>
      </definitions>
```

Yukarıdaki WSDL içerği incelendiğinde aşağıdaki bilgilere ulaşılabilir: Servisin tanımı “HelloService”, kullandığı tipler XML şemada tanımlı tipler, içerdiği mesajlar “firstName” parametresi alan “sayHelloRequest” ve “greeting” parametresi döndüren “sayHelloResponse” mesajlarıdır. Servisin port tipleri istek ve yanıt mesajları olan “sayHello” operasyonu adresi “http://www.examples.com/SayHello/” ve bağlama tipi SOAP HTTP aktarım protokolüdür.

4.3. Universal Description Discovery and Integration (UDDI)

UDDI, web servisleri için kayıt defteri hizmeti vermek üzere tasarlanmış ve büyük kuruluşların bir çoğu tarafından kabul görmüş (Microsoft, IBM…), Service Oriented Architecture (SOA) bileşenidir.

UDDI, Web servisi sağlayan kuruluşlarla ve sundukları servislerle ilgili genel bilgileri tutar ve bu bilgilere paylaşımı sağlar. UDDI, kuruluşların sağladıkları servislerin detaylı bilgisini, standartlarını, kısaltmaları, tipleri ve kimlik bilgilerini kütükler halinde saklar.

UDDI, web servislerini temel alan SOA için temel altyapı bileşenlerinden biridir; çünkü sayıları hızla artan servislere ulaşmak ve onları yönetmek gün geçtikçe önemi artan bir sorun durumuna gelmiştir.

5. ONTOLOJİLERİN UZAKTAN YÖNETİMİ

Anlamsal Ağ siteleri ya da uygulamaları için ontolojilerin uzaktan yönetimi temel bir konudur. Ontolojilerin uzaktan yönetimine gereksinim duyulma nedenleri aşağıdaki gibi özetlenebilir:

- Ontoloji üzerinde zaman ve mekan kısıtlaması olmaksızın değişiklik yapılabilmesinin sağlanması,
- Kullanıcıların ontolojileri genişletebilmesi, değiştirebilmesi ya da yeni ontolojiler oluşturabilmesinin sağlanması,
- Ontoloji üzerinde yapılacak işlemlerin denetlenmesi için güvenlik politikalarının uygulanması.

Ontolojilerin uzaktan yönetimi ile bir web sitesinin ya da uygulamasının içeriği kullanıcılar tarafından belirlenebildiği gibi, kategoriler, özellikler, ilişkiler ve kurallar da kullanıcılar tarafından eklenebilir. Örneğin, bir kullanıcı otomobilleri konu edinen bir siteye otomobillerle ilgili yeni bir özellik ekleyebilir. Böylece otomobilleri betimleyen yeni bir özellik elde edilmiş olur.

Anlamsal Ağ yaklaşımına göre tasarlanmış bir web sitesinde, bir konu ile ilgili okunacak ya da girilecek veri alanlarını, içeriğin sıradüzenini, kuralları ve özellikleri de kullanıcılar belirleyebilir. Hatta kullanıcılar yeni bir çalışma alanı ya da ontoloji açarak içeriğe yepyeni bir boyut katabilir. Özetle, önceki web yaklaşımlarında yalnızca bilgi ekleyebilen kullanıcılar, anlamsal bir web sitesine yapı da ekleyebilir.

Anlamsal Ağ uygulamalarında, uygulamanın içeriği ve bu içeriğin yapısı bir içerik yöneticisine gereksinim duyulmaksızın kendiliğinden genişleyebilir. Tüm bunlar için web uygulamasının ontolojilerden oluşması ve bu ontolojilerin de uzaktan yönetilebilmesi gerekir.

5.1. Ontoloji Yönetiminin Temel Özellikleri

- Geliştirilecek çözüm varolan ağ standartlarına dayanmalı yeni protokollere, katmanlara vb. gereksinim duymamalıdır.

- Ontoloji yönetimine katılması beklenen kullanıcıların temel ontoloji bilgisi dışında ayrıntılı teknik bilgiye sahip olması gerekmemelidir.

- Kullanıcıların sisteme erişebilmesi için kurulum vb. herhangi özel bir çabaya gereksinim olmamalıdır.

- Kullanıcıların ontoloji yönetim sisteminin kullanımı için özel araçlara başvurması gerekmemelidir.

- Ontoloji ile ilgili işlevselliğin sunucu tarafında olması gerekir; aksi durumda kullanıcıların sisteminde ontoloji ile ilgili kütüphanelerin yüklü olması gerekir. Bu durum ise kütüphanelerin güncelliği, kütüphanelerin doğru biçimde yüklenip yüklenmediği, gereksinim duyulan bir aracın kullanıcı tarafından silinmesi veya değiştirilmesi gibi bir çok soruna yol açar.

- Ontoloji Yönetim Modülü işletim sistemi, web tarayıcı gibi araçlardan herhangi birine bağımlı olmamalı, platform ve uygulamadan bağımsız olmalıdır.

- Ontoloji ile ilgili işlemleri gerçekleştiren modül, çeşitli ontoloji yönetim modülleri tarafından kullanılabilir olmalı, diğer bir deyişle istemci modülünden bağımsız olmalıdır.

- Ontoloji Yönetim Modülü ile ontoloji işlemlerini gerçekleştiren modül arasındaki iletişim, ateş duvarları gibi engellere takılmamalıdır.

- Ontoloji Yönetim Modülü, hem masaüstü hem de web uygulamaları tarafından kolaylıkla kullanılabilmeli, diğer bir deyişle bu modül kullanılarak farklı uygulamalar yazılabilmelidir.

- Hem Ontoloji Yönetim Modülü, hem de ontoloji işlemlerini gerçekleştiren modüle İnternet ortamından erişilebilmelidir. Diğer bir deyişle Ontoloji Yönetim Modülüne bir web sayfasından erişilebilmeli, ontoloji işlemlerini gerçekleştiren modül ile ilgili bilgiler de standart bir biçimde İnternet ortamında saklanmalı ve erişime açık olmalıdır.

5.2. Ontoloji İşlemlerini Gerçekleştiren Web Servisi Modülü

Yukarıda açıklanan özellikler gözönüne alındığında ve aşağıdaki nedenlerden dolayı, bu çalışmada ontoloji işlemlerini yapan modülün bir web servisi olmasına karar verilmiştir:

- Ontoloji işlemlerinin, istemci ortamından bağımsız bir modül tarafından gerçekleştirilmesi gerekir. Web servisleri esnek ve modülerdir. Aynı web servisini kullanan çok çeşitli uygulamalar yazılabilir.

- Bu modülün bir sunucuda bulunması, bu modülün yaygın ve ateş duvarlarına karşı korunaklı bir kanal ile iletişimi sağlaması gerekir. Web servisleri en yaygın iletişim protokolü olan ve ateş duvarları gibi engellere takılmayan HTTP protokolünü kullanır; bundan başka özel bir protokole gereksinim duymaz.

- Standart bir veri biçiminde bilgi alımı ve gönderimi gerekir. Web servislerinin istemci-servis sağlayıcı arasındaki SOAP mesajları XML tabanlıdır ve bilindiği gibi XML, günümüz web dünyasının standart veri formatıdır.

- Geliştirilecek çözümün, platform ve uygulamadan bağımsız olması nedeni ile, istemci ortamında herhangi özel bir bileşene gereksinim duymaması gerekir. Web servisleri XML ve HTTP tabanlı olduğu için herhangi bir işletim sistemi ya da yazılıma bağımlı değildir.

- İşlevselliğine ilişkin bilgilerinin standart bir biçimde İnternet ortamından yayınlanması gerekir. Web servisleri, tanım bilgilerini WSDL denilen standart bir biçimde İnternet'ten yayınlar.

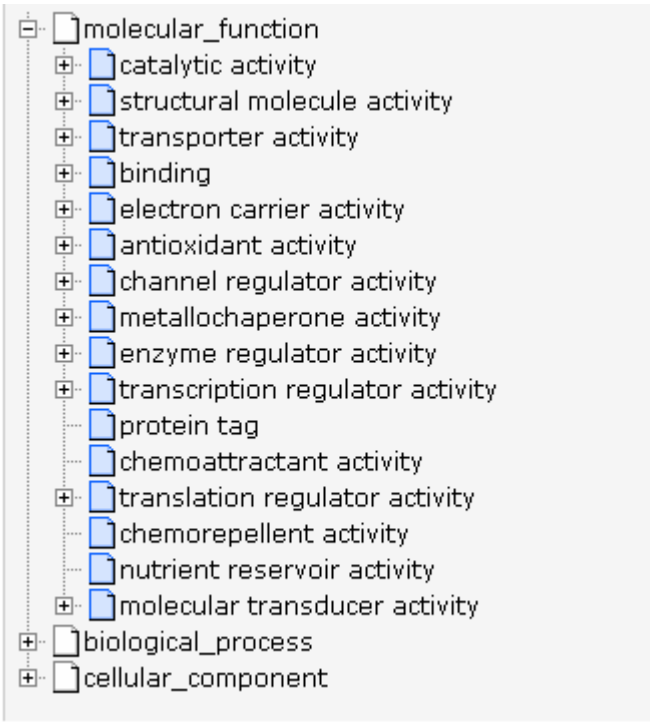
5.3. Ontoloji Servisi Örnekleri

Ontoloji paylaşımı ya da uzaktan yönetimi için web servislerinin kullanımı, çok yaygın olmamakla birlikte günümüzde kullanılan bir yöntemdir.

Aşağıda bu konudaki örnek uygulama ve çalışmalara ilişkin tanıtıcı bilgiler bulunmaktadır.

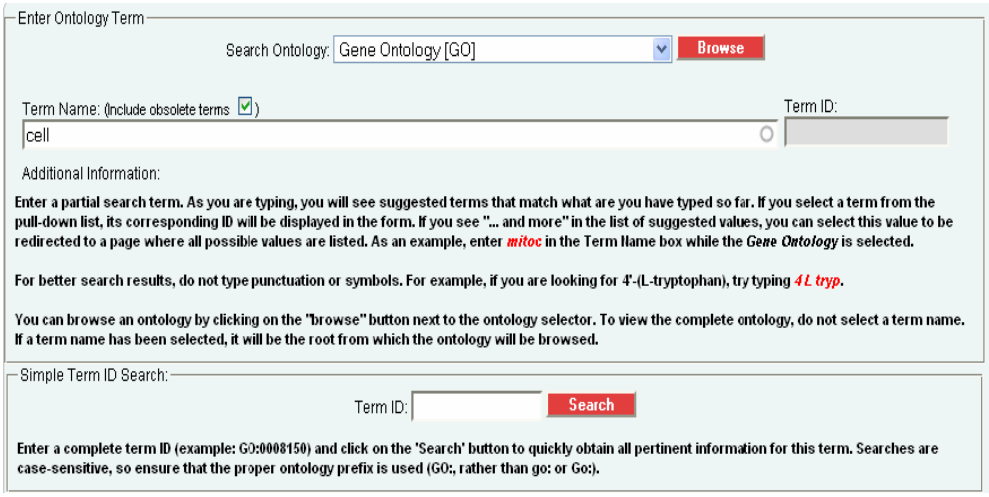
Ontology Lookup Service (OLS)

OLS, sisteme kayıtlı ontolojilerin paylaşımına yönelik açık kaynak kodlu bir uygulamadır. OLS ile ontolojiler ağaç yapısı biçiminde görünütülenebildiği gibi, ontoloji içinde varlık araması da yapılabilir. "Gene" ontolojisinin ağaç biçimindeki gösterimi Şekil 8.a'da verilmiştir. [16]



Şekil.8.a) OLS ile ontoloji gösterimi

Şekil 8.b ise "Gene" ontolojisinde terim (varlık) aramasını göstermektedir.



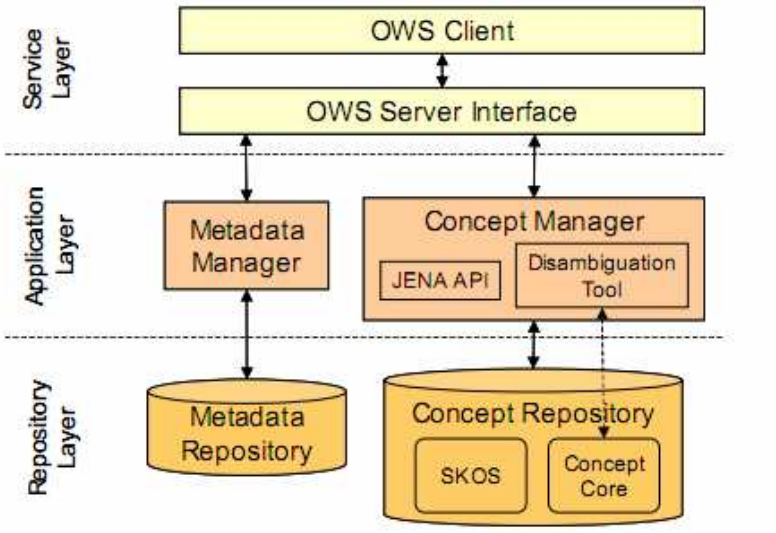
Şekil.8.b) OLS ile ontoloji sorgulama

OLS web servisi ile bilgilere, OLS'nin WSDL belgesi yardımıyla erişilebilir.[17]

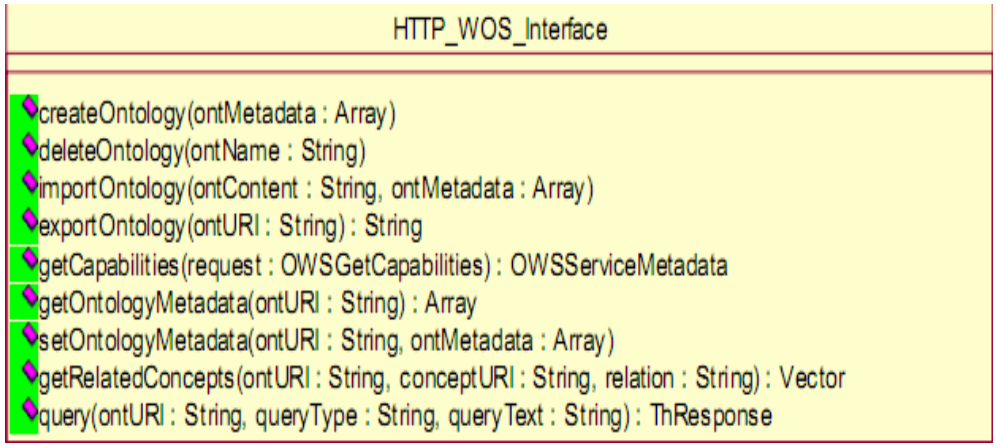
Uzamsal veri altyapısı için temel bir bileşen: Web Ontology Service (WOS)

WOS, uzamsal verilerin standarta kavuşturulması ve paylaşımı için ontololojilerin kullanılmasına ve bu ontolojilerin bir web servisi ile paylaşılmasına ve genişletilmesine yönelik bir projedir. [18]

Şekil 9.a'da projenin mimarisi, Şekil 9.b'de ise web ontoloji servisinin arayüz metotları verilmektedir.



Şekil.9.a) WOS mimarisi

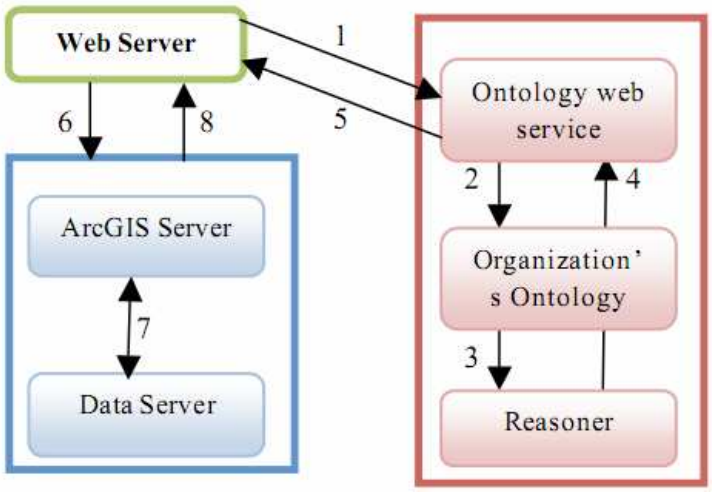


Şekil.9.b) WOS arayüz metotları

Tahran'daki kurumların anlamsal veri paylaşımını artırmak için ontoloji temelli Geographic Information System (GIS) web servisi

GIS web servisi, Tahran'da sondaj yapan kurumların uzamsal verileri standart bir biçimde paylaşabilmesini ve genişletebilmesini amaçlayan ontoloji temelli bir web servisi projesidir. [19]

Şekil 10'dan da anlaşılacağı üzere; bu projede ontoloji web servisi, belediyenin web sunucusundaki uygulama sayesinde, kullanıcının isteğini ilgili kuruluşun ontolojisine iletir ve ontolojiden edindiği bilgiyi web sunucusuna, diğer bir deyişle kullanıcıya gönderir.



Şekil.10. GIS sisteminin yordamı

5.4. Ontoloji Yönetim Modülü olarak Applet

Appletler, Java Programlama Dili ile yazılan ve bir web sayfasına kolayca gömülebilen programlardır. Java Runtime Environment'ın (JRE) yüklü olduğu bir bilgisayarda, Java desteği olan bir web tarayıcısı ile, applet içeren bir web sayfası çağrıldığında, applet kodları bilgisayara yüklenir ve burada çalıştırılır. Diğer bir deyişle appletler istemci tarafında çalışan Java programcıklarıdır. Appletler, Java ile geliştirilen programcıklar olduğu için rahatlıkla bir Java uygulamasında da kullanılabilir. Appletlerin geliştiriciler için en önemli yararı, Java Sanal Makinelerinden dolayı işletim sistemi ve web tarayıcılarından bağımsız olmalarıdır.

Aşağıda sıralanan nedenlerden dolayı, Ontoloji Yönetim Modülünün bir applet olarak geliştirilmesine karar verilmiştir:

- İstemci tarafında, günümüzde oldukça yaygın bir biçimde kullanılan Java Çalışma Ortamı dışında, özel yazılım bileşenlerine gereksinim duymaması,
- Hem masaüstü uygulamalarda hem de bir web sayfasında rahatlıkla kullanılabilmeleri,
- Açık kaynak kodlu Jena çerçevesi ile aynı dilde (Java) geliştirilebilir olması,
- Platform ve web tarayıcıdan bağımsız olması,
- Appletlerin, Java Web Start Application teknolojisinin çekirdeği olması. Java Web Start Application teknolojisi sayesinde appletler tipik bir yazılım gibi istemci tarafına kendiliğinden kurulur ve yazılımlar gibi kendiliğinden güncellenebilir, program grubu oluşturulur ve program oluşturulan kısayol aracılığı ile çalıştırılabilir.

6. SİSTEMİN GELİŞTİRİLMESİ

6.1. Sistemin Tasarımı

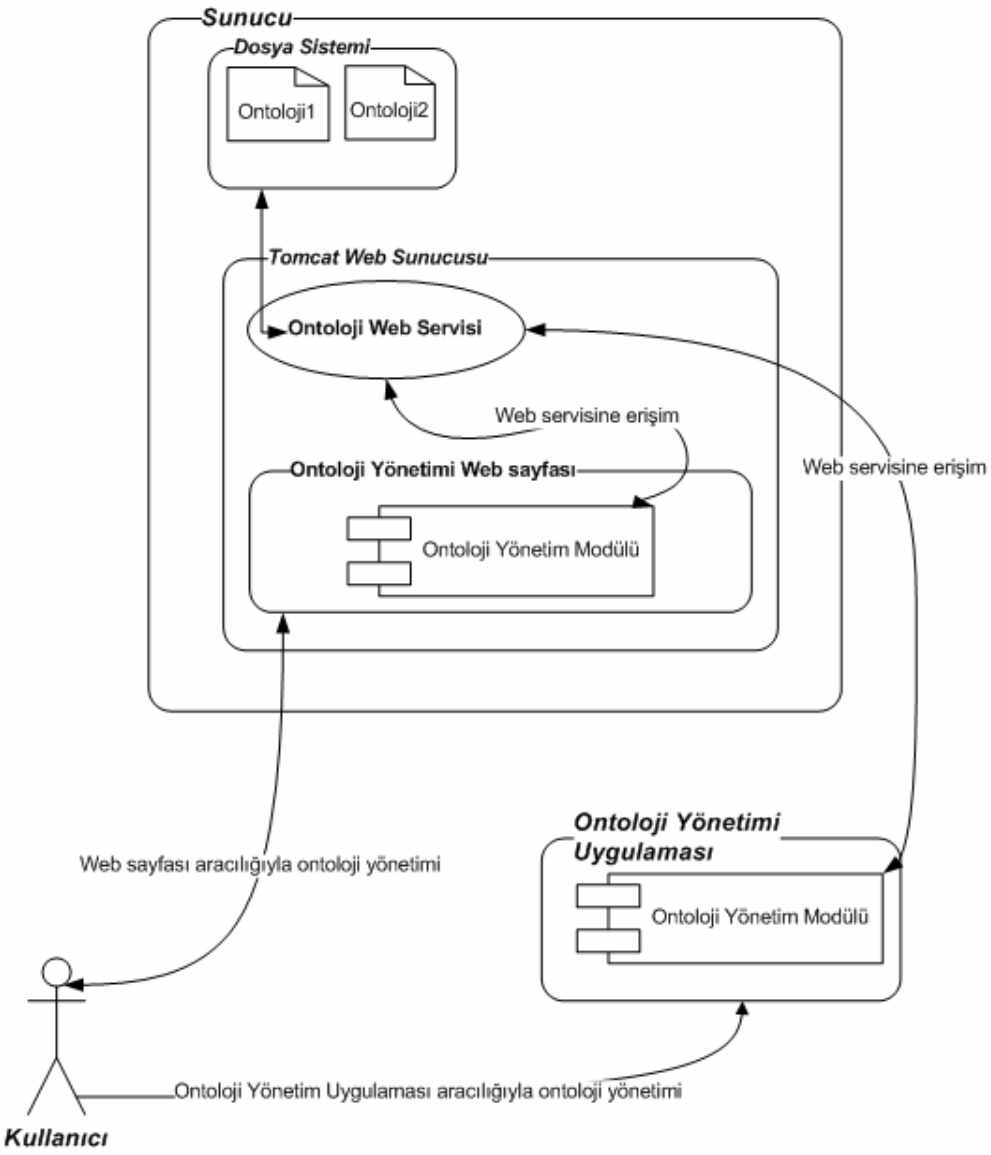
Bu çalışmanın yapıtaşını oluşturan uzaktan ontoloji yönetimine ilişkin çözüm için izlenecek adımlar şunlardır:

- Sistemin bileşenlerinin belirlenmesi,
- Sistem tasarımı,
- Kullanılacak araçların seçimi,
- Ontoloji Servisinin geliştirilmesi,
- Ontoloji Yönetim Modülünün geliştirilmesi,
- Sistemin testi, iyileştirmeler ve eklemeler.

Bu çalışmada tasarlanan sistemde, Ontoloji Yönetim Modülü kullanıcıların isteklerini bir web sunucusu tarafından sunulan Ontoloji Servisine iletecek, Ontoloji Servisi ise istenen işlemi gerçekleştirecek ve sonucu istemciye gönderecektir. Geliştirilen sistemin temel bileşenleri aşağıda belirtilmiştir:

- Ontoloji işlemlerini gerçekleştirecek bir Ontoloji Web Servisi,
- Web servisinin konumlandırılacağı Web Sunucusu,
- Ontoloji Yönetim Modülü ve bunu içeren bir web sayfası ya da uygulama.

Geliştirilecek olan sistem dosya sisteminde bulunan ontolojilerin sunucu tarafındaki web servisi ve web sayfası yoluyla yönetimini sağlamaya yöneliktir. Web sayfasının içerdiği applet, Tomcat web sunucusundaki web servisine istek gönderir, web servisi dosya sistemindeki ontolojilere erişir ve istenilen görevi yapar. Web servisi ile iletişen applet, bir web sayfasında kullanılabileceği gibi başka bir Java uygulamasında da kullanılabilir. Şekil 11, tasarlanan Ontoloji Yönetim Sistemi mimarisini göstermektedir.



Şekil.11. Ontoloji Yönetim Sistemi mimarisi

Bu tasarıma göre, Ontolojiler web sunucusunun dosya sisteminde saklanacaktır. Ontolojilerle ilgili işlemleri bir web servisi gerçekleştirecektir. Kullanıcı ile web servisi arasındaki ilişkiyi Ontoloji Yönetim Modülü sağlayacaktır. Sistemin bileşenleri ve bu bileşenlerin birbirleri ile etkileşimleri Applet→Web Servisi→Ontoloji sırasında gerçekleşecektir.

6.2. Çalışmada Kullanılan Araçlar

Sistemin geliştirilmesi ve kullanılması için gereken araçlar, sunucu tarafında konumlandırılacak bileşenler ve istemci tarafında konumlandırılacak bileşenler olarak iki gruba ayrılır.

Sunucu tarafındaki bileşenler

- Ontolojilerin yer aldığı sunucu dosya sistemi,
- Web sunucusu,
- Web sunucusunun yayınlayacağı
    - Ontoloji Web Servisi,
    - Ontoloji Yönetim Modülünü içeren web sayfası

olarak sınıflandırılır.

İstemci tarafındaki bileşenler

- Java Çalışma Ortamı,
- Web tarayıcısı,
- Ontoloji Yönetim Modülünü içeren masaüstü uygulama

olarak incelenir.

Masaüstü uygulaması sistemin çalışması için zorunlu bir bileşen değildir. Bu uygulama, Ontoloji Yönetim Modülünün masaüstü uygulamalar tarafından da kullanılabileceğini göstermek için geliştirilmektedir.

Burada dikkat edilmesi gereken konu, Ontoloji Yönetim Modülünün, appletlerin çalışma biçiminden dolayı istemci tarafına yüklenmesi ve istemci tarafında çalışmasıdır. Appletler, ağdan erişilmekle birlikte istemci tarafında çalışan tüm bileşenler gibi (örneğin ActiveX'ler) güvenlik nedeniyle çeşitli kısıtlamalara bağlıdır ve sistemin geliştirilmesi aşamasında bu kısıtların gözetilmesi gerekir.

Sistem tasarımına göre sistemin geliştirilmesi ve test edilmesi için gerekenler şunlardır:

- Örnek ontolojiler geliştirmek için bir ontoloji düzenleyicisi,

- Ontoloji Servisi ve Ontoloji Yönetim Modülünü içerecek olan web sayfasını yayınlayan bir web sunucusu,
- Java geliştirme ortamı,
- Herhangi bir web tarayıcı.

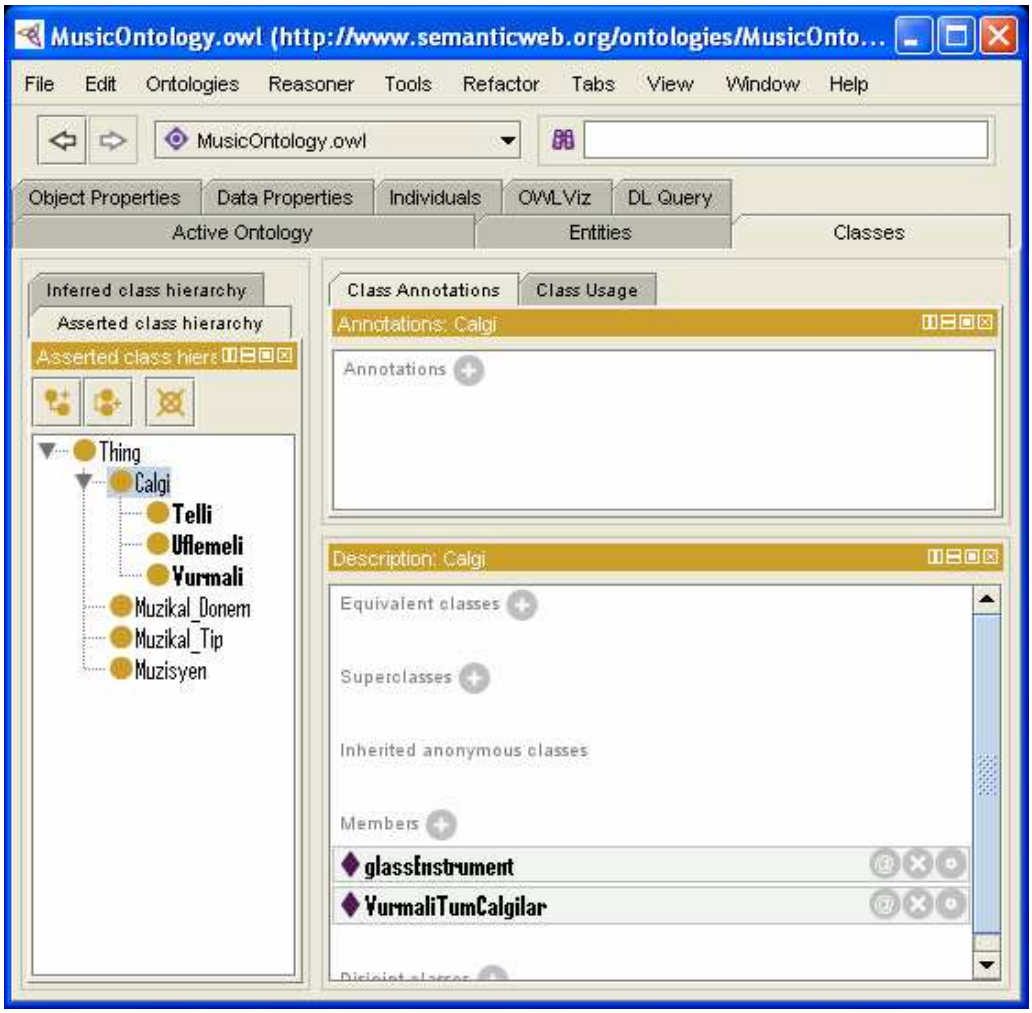
6.2.1. Ontoloji Geliştirme Aracı:Protege

Ontolojiler, XML tabanlı düz metin dosyaları olduğu için, bir ontoloji oluşturmak için herhangi bir metin düzenleyicisi yeterlidir. Bununla birlikte metin düzenleyicisi ile ontoloji geliştirmek hem çok fazla hata yapmamıza neden olur; hem de çok fazla zamanımızı alır. Ayrıca, ontoloji geliştirmek için tasarlanmış uygulamalarda içerilen birçok kolaylığı ve gelişmiş özelliği kullanamayız.

Ontoloji düzenleyicileri, kullanıcıların kolaylıkla ontoloji geliştirmesini sağlayan, hata riskini azaltan ve ontolojilerle ilgili doğrulama, sorgulama gibi gelişmiş işlevleri gerçekleştiren uygulamalardır.

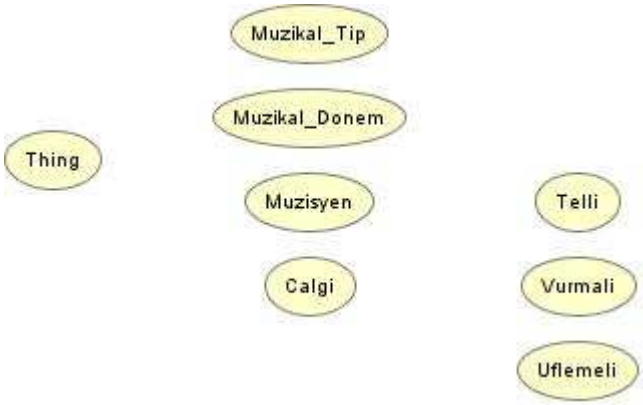
Sistemin test edilmesi ve gereken ontolojileri oluşturmak için bir ontoloji düzenleyicisine gereksinim vardır. Bununla birlikte sistemin geliştirilmesi tamamlandıktan sonra, ontoloji ile ilgili temel işlemleri Ontoloji Servisi üstleneceği için ontoloji düzenleyicisine gereksinim duyulmayacaktır.

Çalışmada, ontoloji düzenleyicisi olarak Protege seçilmiştir. Protege, Java ile geliştirilmiş açık kaynak kodlu ve ücretsiz bir uygulamadır. Protege, sağladığı ontoloji API'si sayesinde genişletilebilir esnek bir uygulamadır. Protege'in en önemli yararlarından biri ise ontolojileri görselleştirebilmesi, böylece sınıf sıradüzeninin gözden geçirilmesine olanak vermesidir. Şekil 12.a'da Protege ontoloji geliştirme ortamı gösterilmektedir.



Şekil.12.a) Protege ontoloji geliştirme ortamı

Şekil 12.b'de ise, örneklenmiş olan Müzik konulu ontolojiye ait sınıfların, Protege ile elde edilmiş görsel gösterimi bulunmaktadır.



Şekil.12.b) Protege ile müzik ontolojisinin görsel gösterimi

6.2.2. Web Servisi Sunucusu:Apache Tomcat

Apache Tomcat, Java Servlet ve Java Server Pages çalıştıran açık kaynak kodlu ücretsiz bir yazılımdır. Çalışmada Tomcat 6 versiyonu kullanılmıştır. Tomcat, sistem tasarımında Ontoloji Servisininin barındırılması ve Ontoloji Yönetim Modülünü içeren web sayfasınının sunulması için kullanılır.

6.2.3. Ontoloji Uygulama Programlama Arayüzü:Jena

Çalışmada, ontoloji uygulama programlama arayüzü olarak Jena tercih edilmiştir. Jena kütüphaneleri, ontoloji web servisinin geliştirilmesinde kullanılmıştır.

Ontoloji Yönetim Modülü olarak geliştirilen applet ontoloji ile ilgili herhangi bir kod içermemektedir, bundan dolayı Jena yalnızca Ontoloji Servisi geliştirilirken kullanılmıştır.

6.2.4. JAVA Geliştirme Ortamı:Netbeans

Netbeans, Sun Microsystem tarafından geliştirilen, açık kaynak kodlu çok yaygın bir Java geliştirme ortamıdır. Netbeans geliştirme ortamında, kod yazma, derleme, hata giderme gibi temel Integrated Development Environment (IDE) işlevlerinin yanısıra çeşitli Java uygulamaları için taslak kodlar üreten sihirbazları da vardır.

Netbeans, web servisi ve servis istemcilerinin geliştirilmesi konusunda geliştiricilere büyük kolaylıklar sağlar. Örneğin Tomcat, Glassfish gibi web sunucuları ile bütünleşik biçimde çalışabilir. Web servisleri için Web Application Archive (WAR) dosyasını ve web istemcisi için Java Archive (JAR) dosyasını üretir.

Bir başka Netbeans kolaylığı ise, appletler için JNLP dosyasının üretilmesi ve appletlerin kendiliğinden imzalanmasıdır.

Çalışmada Netbeans 6.8 kullanılmıştır.

6.3. Ontoloji Web Servisi

6.3.1. Ontoloji Servisinde Olması Gereken Temel Metotlar

Çalışmanın amacı, ontoloji ile ilgili temel işlemleri yapan bir web servisi ve bu servise erişerek kullanıcıların ontoloji oluşturması, silmesi ve değiştirmesi işlemlerini sağlayan bir istemci modülü geliştirmek olduğu için, Ontoloji Servisinin içermesi gereken metotlar ontoloji ile ilgili temel işlemleri yerine getirecektir.

Bununla birlikte ontolojilerle ilgili önemli bir başka konu da ontolojilerin birleştirilmesi, kesiştirilmesi ve farklarının alınması işlemleridir. Bu işlemleri sağlayan metotlar, Ontoloji Servisinde yer alması zorunlu olmamakla birlikte, çalışmanın kapsamı içine alınmıştır.

Ontoloji Servisinin sağlaması gereken işlevler aşağıdadır:

- Ontoloji listesini göndermek,
- Ontoloji oluşturmak,
- Ontoloji silmek,
- Ontolojinin ismini değiştirmek,
- URL'si verilen bir ontolojinin sunucu dosya sistemine indirmek,
- Verilen iki ontoloji için,
    - Birleştirme,
    - Kesişim,
    - Fark alma,

  işlemlerinin yapılması,
- İsmi verilen ontolojinin,
    - Sınıf sıradüzenini,

- Özelliklerini (Niteliklerini),
- Varlıklarını (Bireylerini) ve bu varlıkların özelliklerine verilen değerleri

içeren metin dizisini göndermek,

- Sınıf oluşturmak ve silmek,
- Özellik oluşturmak ve silmek,
- Varlık oluşturmak ve silmek,
- Bir varlığın bir özelliğine değer vermek ve verilen değeri silmek.

Ontoloji Servisi geliştirilirken gözetilen önemli konular şunlardır:

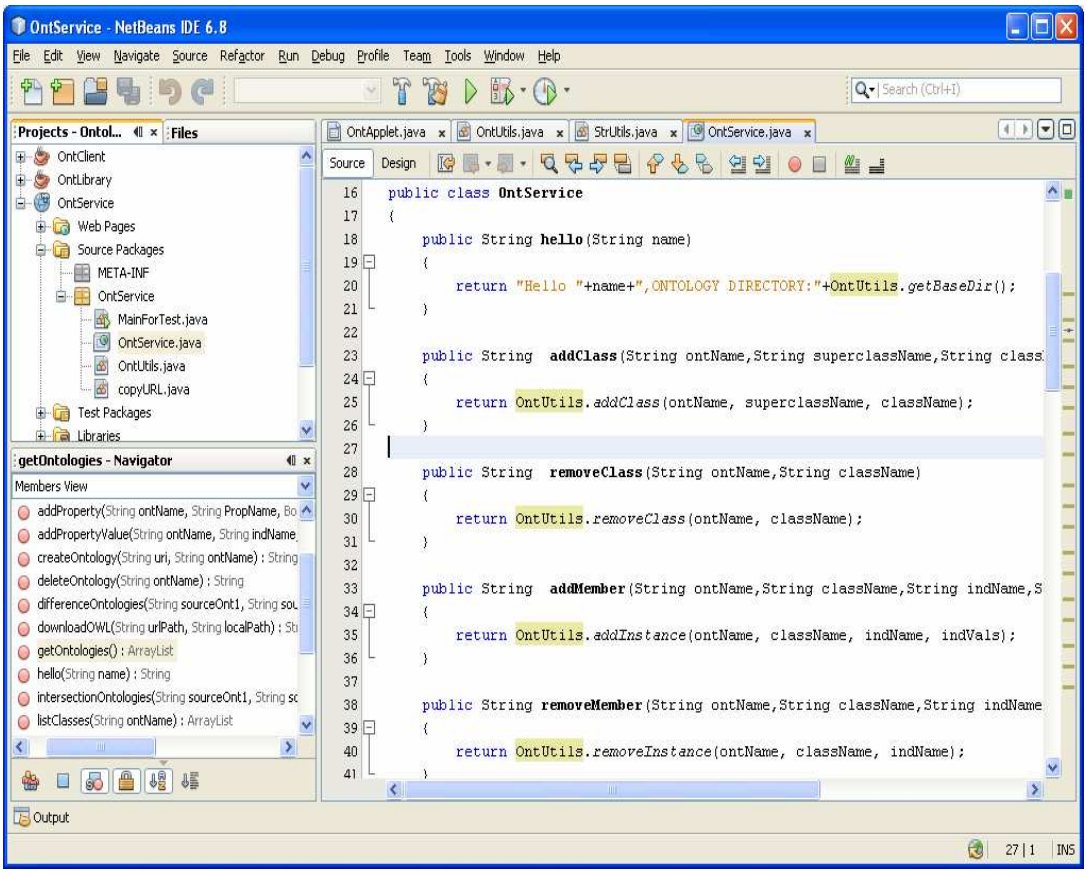
- Ontoloji Servisi, istemcisinin bağlantı testi yapabilmesi için bir test metodu barındırmalıdır. Böylece Ontoloji Yönetim Modülünde öncelikle servisle olan bağlantı test edilebilir.
- Ontoloji Servisi, isteklerin tümü için bir sonuç dönmelidir. Çalışmada dönüş tipi olarak string (metin) tipi seçilmiştir. Bunun nedeni istek yerine getirilirken bir hata olduğunda, hata ile ilgili açıklamanın istemciye iletilebilmesidir.
- Ontoloji Servisi ontolojiyi değiştiren herhangi bir işlemden sonra ontoloji dosyasını kaydetmelidir. Örneğin kullanıcının yeni bir sınıf oluşturma isteği bulunduğunda, istek yerine getirildikten hemen sonra ontoloji yeni haliyle kaydedilmelidir. Kaydetme işlemi bitirildikten sonra istemciye sonuç dönmelidir. Böylece kaydetme işleminde bir sorun çıktığında kullanıcı bundan haberdar olabilecektir.
- Ontoloji Servisi ile istemcisi arasında veri trafiğini azaltmak amacıyla ontolojiye ait bilgiler istemciye metin dizisi biçiminde gönderilmelidir. Ontoloji Yönetim Modülünde, sınıf, özellik, varlık ve değer listelenmesi bu metin dizisi işlenerek elde edilmelidir.

- Ontoloji Servisinin hangi dizindeki ontoloji dosyaları üzerinde işlem yapacağı parametrik olarak belirlenebilmelidir.

6.3.2. Ontoloji Servisinin Geliştirilmesi

Netbeans ortamında yeni bir web uygulaması oluşturarak web servisi geliştirilmeye başlandı. Uygulama oluşturma sihirbazında, sunucu olarak Tomcat 6 seçildikten sonra, Tomcat uyumlu bir web uygulaması için gerekli olan başlangıç dosyaları oluşturuldu.

Ontoloji Servisi, ontoloji ile ilgili işlemler için Jena OWL API'sini kullanacağı için, Jena kütüphaneleri Netbeans çalışma ortamına "global kütüphane" olarak eklendi. Uygulamaya bir web servisi eklemek için ise, Netbeans'in web servis sınıfı oluşturan sihirbazı kullanıldı. Bu sihirbaz ile web servisi adı ve paketi gibi bilgiler girildikten sonra servisin metotlarının yazılmasına başlandı. Şekil 13, Netbeans geliştirme ortamını göstermektedir.



Şekil.13. Netbeans Geliştirme ortamı

Ontoloji web servisindeki metotlar

Aşağıdaki tabloda Ontoloji Servisindeki metotlar ve açıklamaları bulunabilir:

Tablo.1. Ontoloji Servisindeki metotlar

Konu	İşlev (Komut ismi)	Çağırdığı Web Servisi Metodu
Web servisi testi	Web Servis test.	*hello*
Ontoloji	Ontolojileri listele	*getOntologies*
	Yeni ontoloji oluştur	*createOntology*
	Ontolojiyi sil	*deleteOntology*
	İsmini değiştir	*renameOntology*
	Birleşim	*mergeOntologies*
	Kesişim	*intersectionOntologies*
	Fark	*differenceOntologies*
Sınıf	Sınıflar	*listClasses*
	Yeni sınıf oluştur	*addClass*
	Sınıfı sil	*removeClass*
Nitelik	Yeni nitelik oluştur	*addProperty*
	Niteliği sil	*removeProperty*
Varlık	Yeni varlık oluştur	*addMember*
	Varlığı sil	*addMember*
Değer	Yeni değer oluştur	*addPropertyValue*
	Değeri sil	*removeMember*

Kullanılan Jena arayüzleri ve metotları

Ontoloji Servisi geliştirilirken, ontoloji oluşturma, yükleme, kaydetme, değiştirme ve karşılaştırma işlemleri için ModelFactory, OntModel, Ontology, RDFNode, OntClass, Individual, OntProperty, DataTypeProperty ve ObjectTypeProperty gibi temel Jena arayüzleri (interface) kullanıldı.

Tablo 2, kullanılan Jena arayüzlerini (interface) ve metotlarını ve bunların ne amaçla kullanıldıklarını listelemektedir:

Tablo.2. Kullanılan Jena arayüzleri ve metotları

Arayüz	Metot ismi	Kullanım amacı
ModelFactory	*createOntologyModel*	Ontoloji Model nesnesi oluşturmak
OntModel	*getOntClass*	Ontolojideki bir sınıfa erişmek
	createClass	Sınıf oluşturmak
	getIndividual	Ontolojideki bir varlığa erişmek
	createIndividual	Varlık oluşturmak
	getOntProperty	Ontolojideki bir özelliğe erişmek
	createObjectProperty	Nesne tipi özellik oluşturmak
	createDatatypeProperty	Veri tipi özellik oluşturmak
	createOntology	Modelde yeni bir ontoloji oluşturmak
	difference	Bir ontolojinin diğerinden farkını bulmak
	listOntologies	Bir modeldeki ontolojileri listelemek
	intersection	İki ontolojinin kesişimini elde etmek
	listNamedClasses	Ontolojinin sınıflarını listelemek
	Union	İki ontolojiyi birleştirmek
	Read	Modeli OWL dosyasından okumak
	Write	Modeli bir OWL dosyasına yazmak
Ontology	*getURI*	Ontolojinin URI değerini elde etmek
RDFNode	*getURI*	RDFNode nesnesinin URI değerini elde etmek
OntClass	*listDeclaredProperties*	Sınıfa ait özellikleri listelemek
	createIndividual	Sınıfa ait yeni bir varlık oluşturmak
	remove	Sınıfı silmek
	removeAll	Sınıfa ait bir özelliğin tüm değerlerini silmek
	listInstances	Sınıfa ait tüm varlıkları listelemek
	getLocalName	Sınıfın yerel adını elde etmek
	listSubClasses	Sınıfa ait alt sınıfları listelemek
	hasSuperClass	Sınıfın bir üstsınıfı olup olmadığını öğrenmek
Individual	*addProperty*	Varlığın belirtilen özelliğine değer vermek
	listPropertyValues	Varlığın değerlerini listelemek
	removeProperty	Varlığın bir değerini silmek
	remove	Varlığı silmek
	getLocalName	Varlığın yerel ismini elde etmek
OntProperty	*isDatatypeProperty*	Veri tipi özellik olup olmadığını öğrenmek
	getRange	Özelliğin Hedef sınıfı ve tipini öğrenmek
	getLocalName	Özelliğin yerel ismini öğrenmek
	isObjectProperty	Nesne tipi özellik olup olmadığını öğrenmek
	remove	Özelliği silmek
DataTypeProperty, ObjectProperty	*setDomain*	Bir özelliğin kaynak sınıfını belirlemek
	setRange	Bir özelliğin hedef sınıfını belirlemek

Ontoloji Servisi ile ilgili ek bilgiler

Web uygulamasının ve onun içerdiği web servisinin adı “OntService”dir.

Web servisinin ana sınıfı “OntService” bu sınıfın ontoloji işlemlerini gerçekleştirmesi için çağırdığı sınıf ise “OntUtils” sınıfıdır. Ontoloji Servisi bunların dışında; Web servisleri için METRO kütüphanelerini, Jena kütüphanelerini, bazı metin işlemleri için geliştirilen “StrUtils” adında bir sınıf içeren “OntLibrary” kütüphanesini, OntUtils sınıfındaki metotları denemek için yazılan “MainForTest” sınıfını, İnternet ortamındaki bir ontolojiyi indirmek için geliştirilen “copyURL” sınıfını kullanmaktadır.

Web servisinde listeleme işlemlerini gerçekleştiren *getOntologies* ve *listClasses* metotlarının dönüş tipi arrayList, diğer tüm metotların dönüş tipi ise string’tir.

Web servisinin, hangi dizindeki ontoloji dosyalarını işleyeceği bilgisi, uygulamaya “Config.properties” dosyasındaki “baseDir” parametresinin değeri ile bildirilir.

Web servisi ana sayfasının tasarlanması

Netbeans kullanılarak bir web projesi oluşturulduğunda, web uygulamasının ana sayfası olarak, “index.jsp” dosyası üretilir. Bu dosya web uygulamasının giriş sayfasıdır. “index.jsp” dosyası, HTML kullanılarak istenilen duruma getirilmiştir. Bu dosyaya erişmek için gereken web adresi, sonraki bölümde açıklanacaktır.

Web servis uygulamasının Tomcat’e kurulması

Netbeans, web projeleri derlediğinde, “dist” adlı dizinde, uygulamanın web sunucusuna yüklenebilmesi için gereken “war” uzantılı dosyayı oluşturur.

WAR dosyaları, bir web uygulamasını oluşturan JSP (Java Server Page), servlet, Java class, XML vb. dosyaların dağıtımı için kullanılan bir JAR (Java Archive) dosyasıdır.

Tomcat’e bir web uygulaması kurmak için, uygulamanın “war” dosyasının, “webapps” dizinine konulması yeterlidir.

Web uygulaması Tomcat’e yüklendiğinde, uygulama dizini aşağıdaki desene uygun olacaktır:

[Tomcat dizini]\[webapps]\[uygulama ismi]

Bu durumda; Tomcat dizini “D:\Tomcat6”, uygulama ismi “OntService” olduğunda, web uygulamasının dizini aşağıdaki gibi olacaktır:

“D:\Tomcat6\webapps\OntService”

Web servisine ise,

http://[sunucu ismi]:[Tomcat erişim portu]/[web uygulamasının ismi]

ya da

http://[sunucu ismi]:[Tomcat erişim portu]/[web uygulamasının ismi]/index.jsp

biçimindeki adreslerden erişilebilir.

Örneğin, sunucu ismi “localhost”, Tomcat erişim portu “8090” ve web uygulama ismi “OntService” olduğunda, web uygulamasına giriş adresi,

“http://localhost:8090/OntService”

ya da

“http://localhost:8090/OntService/index.jsp”

olacaktır.

Ontoloji web servisinin WSDL içeriği

Tomcat, web servislerinin WSDL içeriklerini dosya olarak tutmak yerine, istendiğinde üretir ve gösterir.

Bir web uygulamasının WSDL içeriği Tomcat’ten aşağıdaki biçimde istenir:

[Web uygulaması giriş adresi]/[Uygulama ismi]?wsdl

Örneğin, erişim portu 8090 olan yerel Tomcat sunucusuna kurulmuş olan OntService uygulamasının WSDL içeriğine aşağıdaki adres ile ulaşılabilir:

“http://localhost:8090/OntService/OntService?wsdl”

Ontoloji Servisinin WSDL içeriği Ek-2’de bulunabilir.

6.4. Ontoloji Yönetim Modülü

Daha önce de belirtildiği gibi Ontoloji Yönetim Modülü Java appleti olarak geliştirilmiştir. Bu applet yönetim modülünün kullanımı açısından esneklik sağlamaktadır. Bu çalışmada, Ontoloji Yönetim Modülü Web sayfası içinde, Java uygulaması içinde ve Java Web Start uygulaması olarak üç farklı biçimde kullanılacaktır.

6.4.1. Ontoloji Yönetim Modülünün Geliştirilmesi

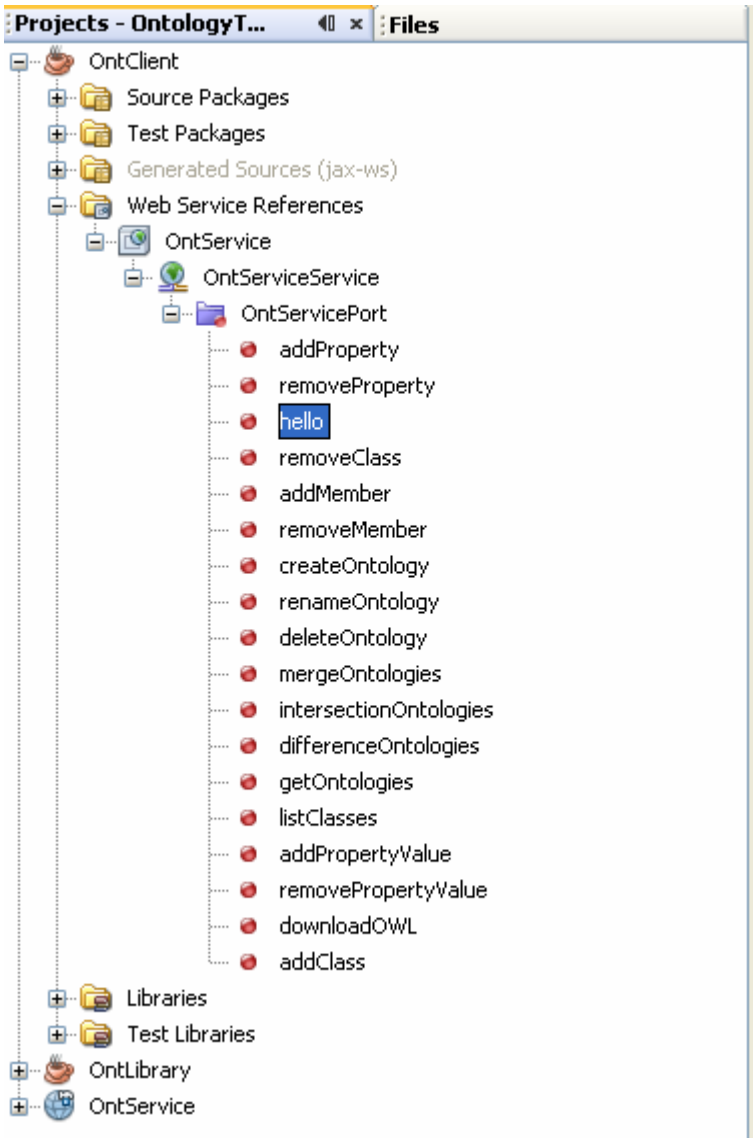
Ontoloji Servisinin geliştirilmesi ile ontoloji yönetiminin sunucu tarafı geliştirilmiş oldu.

Ontoloji Yönetim Modülü ise amaçlanan sistemin istemci tarafına örnek olarak geliştirilmiştir. Bundan dolayı Ontoloji Yönetim Modülü geliştirilirken, görsel tasarımdan daha çok, Ontoloji Servisindeki temel ontoloji işlemlerinin yerine getirilmesi önemsenmiştir.

İstemci uygulamasının oluşturulması

Ontoloji Yönetim Modülünün geliştirilmesine, yeni bir Java Projesi oluşturularak başlandı. Bu modülün temel amacı Ontoloji Servisi ile iletişmek olduğu için, projeye Ontoloji Web Servisi projesi ile bağlantılı olduğu belirtilen bir web servis istemcisi eklendi, böylece web servisi ile ilgili referanslar projeye eklendi ve web servisine erişen metotların yazımına hazır duruma gelindi.

Şekil 14, istemci projesine kendiliğinden eklenen Ontoloji Servisi metotlarını göstemektedir.

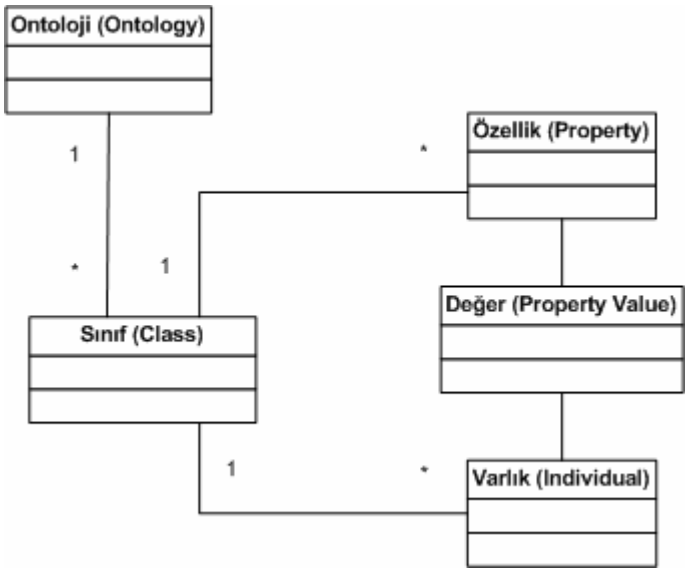


Şekil.14. Web servis istemcisi projesindeki Ontoloji Servisi metotları

Sonraki adım ise, Ontoloji Yönetim Modülünün bir applet olması kararlaştırıldığı için uygulamaya bir applet sınıfı eklemek olmuştur.

Ontoloji Yönetim Modülünün ekran tasarımı

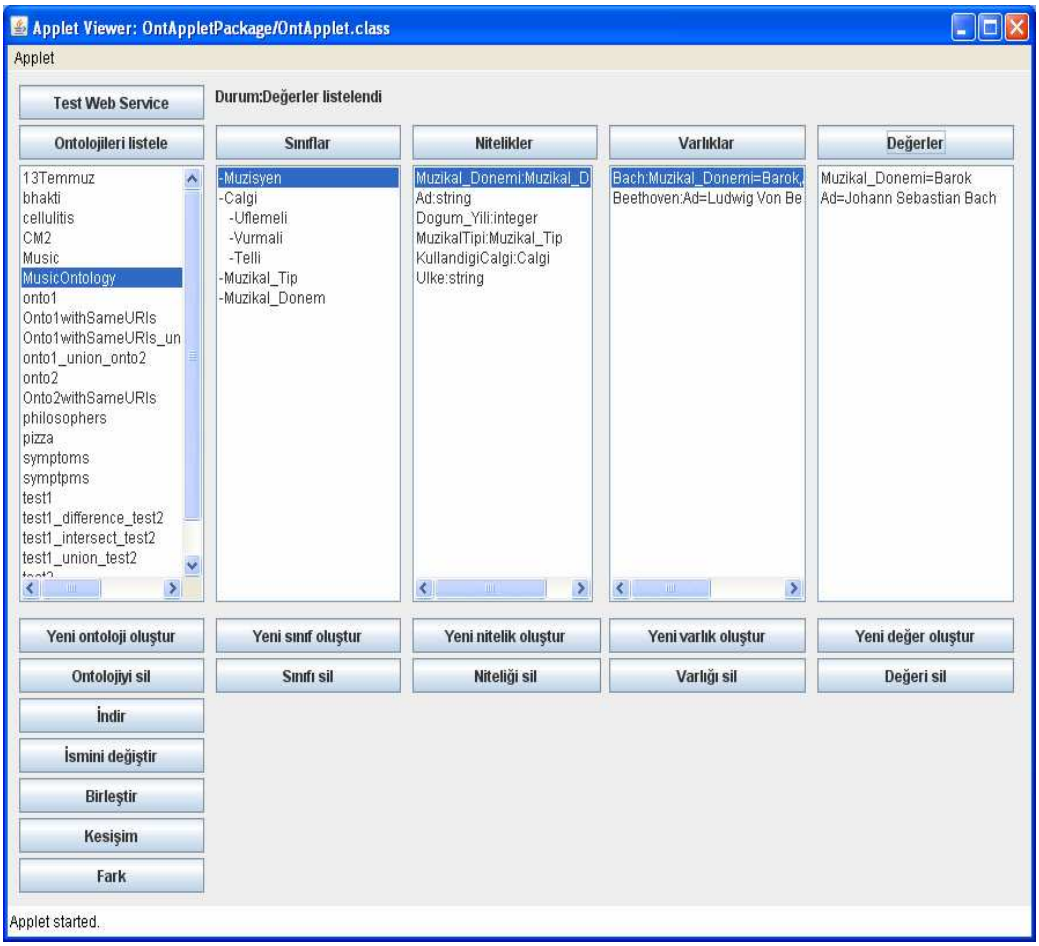
Ontoloji Yönetim Modülü komutlarının ekran üzerindeki yerleşimi, ontolojideki kavram sıradüzenine uygun olarak Şekil 15'te gösterilen biçimdedir:



Şekil.15. Ontoloji bileşenlerinin ilişkileri

Ontoloji Yönetim Modülündeki komutlar, konularına göre aşağıda listelenmiştir:

- Ontoloji ile ilgili komutlar: Ontolojilerin listelenmesi, ontoloji oluşturma, silme, ad değiştirme, ontoloji indirme ve iki ontoloji arasındaki birleştirme, kesişim ve fark işlemleri.
- Sınıflarla ilgili komutlar: Sınıfların listelenmesi, sınıf oluşturma, silme.
- Özelliklerle ilgili komutlar: Özelliklerin listelenmesi, özellik oluşturma, silme.
- Varlıklarla ilgili komutlar: Varlıkların listelenmesi, varlık oluşturma ve silme.
- Değerlerle ilgili komutlar: Değerlerin listelenmesi, değer oluşturma ve silme.



Şekil.16. Ontoloji Yönetim Modülü arayüzü

Şekil 16, Ontoloji Yönetim Modülü arayüzünü göstermektedir.

Arayüz, beş bloktan oluşmakta ve her bir blok belirli bir konuda (ontoloji, sınıf, özellik, varlık, değer) yukarıdan aşağıya doğru dizilmiş bileşenlerden oluşmaktadır. Bu bileşenler listeleme butonu, konuyla ilgili nesneleri içeren liste ve konuyla ilgili yapılacak işlemler için kullanılan butonlardır.

Geliştirilen sınıflar ve kütüphaneler

Ontoloji Yönetim Modülü için, applet sınıfı (OntApplet), web servisindeki "listClasses" metodunun çağrılması sonucunda dönen metin dizisini işleyen EntityListUtils sınıfı ve modülü Java uygulaması olarak kullanabilmek için "OntClientApp" sınıfı geliştirilmiştir. Ontoloji Servisinin geliştirilmesinde de kullanılan "OntLibrary" kütüphanesindeki "StrUtils" sınıfı ise metin işleme için kullanılmıştır.

6.4.2. Ontoloji Yönetim Modülünün Farklı Biçimlerde Kullanımları

Ontoloji Yönetim Modülü bir applet olduğu için bir web sayfasında ve tipik bir java uygulmasında kullanıldığı gibi, bir Java Web Start uygulaması olarak da kullanılabilir.

Ontoloji Yönetim Modülü için geliştirilen proje derlendiğinde, Netbeans geliştirme oramı, "dist" dizini içinde aşağıdaki dosyaları üretir:

- OntClient.jar: Projede kullanılan tüm sınıfları ve diğer dosyaları içeren Java arşivi,
- index.html:Web servisi istemci uygulamasının giriş sayfası,
- launch.jnlp: Java uygulamalarının Java Web Start uygulaması olarak kullanılabilmesi için gereken JNLP dosyası.

Ontoloji Yönetim Modülünün web sayfasında kullanılması

Bir Java appletini web sayfasında kullanmak için "applet" etiketi kullanılır.

Aşağıda JNLP dosyası kullanılarak Ontoloji Yönetim Modülünün bir web sayfasına nasıl yerleştirildiği gösterilmektedir:

```
      <applet width="900" height="650">
          <param name="jnlp_href" value="launch.jnlp"/>
          <param name="WSDLAddress"
value="http://localhost:8090/OntService/OntService?wsdl"/>
      </applet>
```

Yukarıdaki koddan da anlaşılacağı üzere applet için iki parametre kullanılmıştır:

"jnlp_href" parametresi, oluşturulan JNLP dosyasının dosya yolunu belirtmek için kullanılmıştır ve değeri "launch.jnlp"dir. Bu parametre kullanılmak zorundadır.Diğer parametre olan "WSDLAddres" parametresi ise, applete WSDL

adresini belirtmek için eklenmiştir. [1] Modülün içerildiği web sayfasının kullanılabilmesi için OntClient uygulamasının bir web sunucusuna konulması gerekir. Bu amaçla proje derlendikten sonra oluşturulan “dist” dizininin adı “OntClient” olarak değiştirilmiş ve Tomcat “webapps” dizini altına konmuştur.

Ontoloji Yönetim Modülünü içeren web sayfasına erişmek için kullanılması gereken adres deseni,

http://[sunucu ismi]:[Tomcat erişim portu]/OntClient/

ya da

http://[sunucu ismi]:[Tomcat erişim portu]/OntClient/index.html

biçiminde olacaktır.

Sunucu ismi “localhost”, Tomcat erişim portu “8090” ve web uygulama adı “OntClient” olduğunda, Ontoloji Yönetim Modülünü kullanan web sayfasına erişim adresi,

“http://localhost:8090/OntClient/”

ya da

“http://localhost:8090/OntClient/ /index.html”

olacaktır.

Ontoloji Yönetim Modülünün Java uygulaması olarak kullanılması

Bu amaçla geliştirilen “OntClientApp” sınıfının kodları aşağıdadır:

```
public class OntClientApp {
    public static void main(String[] args) {
        OntApplet myOntApplet=new OntApplet();
        myOntApplet.init();
        myOntApplet.start();
```

[1] WSDL Adres bilgisinin seçimli olmasının nedeni ise web servisinin sunucusunun ya da portunun değişme olasılığıdır.

```
            JFrame window = new JFrame("Ontoloji Yönetim
Uygulaması");
            window.setSize(900,650);
            window.setContentPane(myOntApplet);
            window.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
            window.setVisible(true);
            window.setResizable(false);
        }
    }
```

Yukarıdaki koddan anlaşılacağı üzere, “main” sınıfı bir applet örneği yaratmakta ve bunu bir “Jframe” nesnesi içine gömmektedir.

Proje derlendiğinde, OntClientApp sınıfı, OntClient.jar dosyasında içerileceğinden Ontoloji Yönetim Modülünü kullanan Java uygulamasının çalıştırılabilmesi için, konsoldan,

“Java -jar OntClient.jar” komutunun verilmesi gerekir.

Uygulamayı başlatmak için gereken bu türden komutlar, genellikle bir komut dosyası içine (“.bat” dosyası gibi) yazılır.

Ontoloji Yönetim Modülünü bir Java uygulaması olarak çalıştırmak için kullanılan dizinde “OntClient.jar”, “Lib” dizini ve uygulamayı başlatmak için oluşturulmuş komut dosyası (“Run.bat”) bulunmaktadır.

Bunların dışında, web sayfasındaki gibi applete WSDL adres bilgisini (“WSDLAddress=http://localhost:8090/OntService/OntService?wsdl”) belirtmek için “Config.Properties” dosyası da kullanılmaktadır.

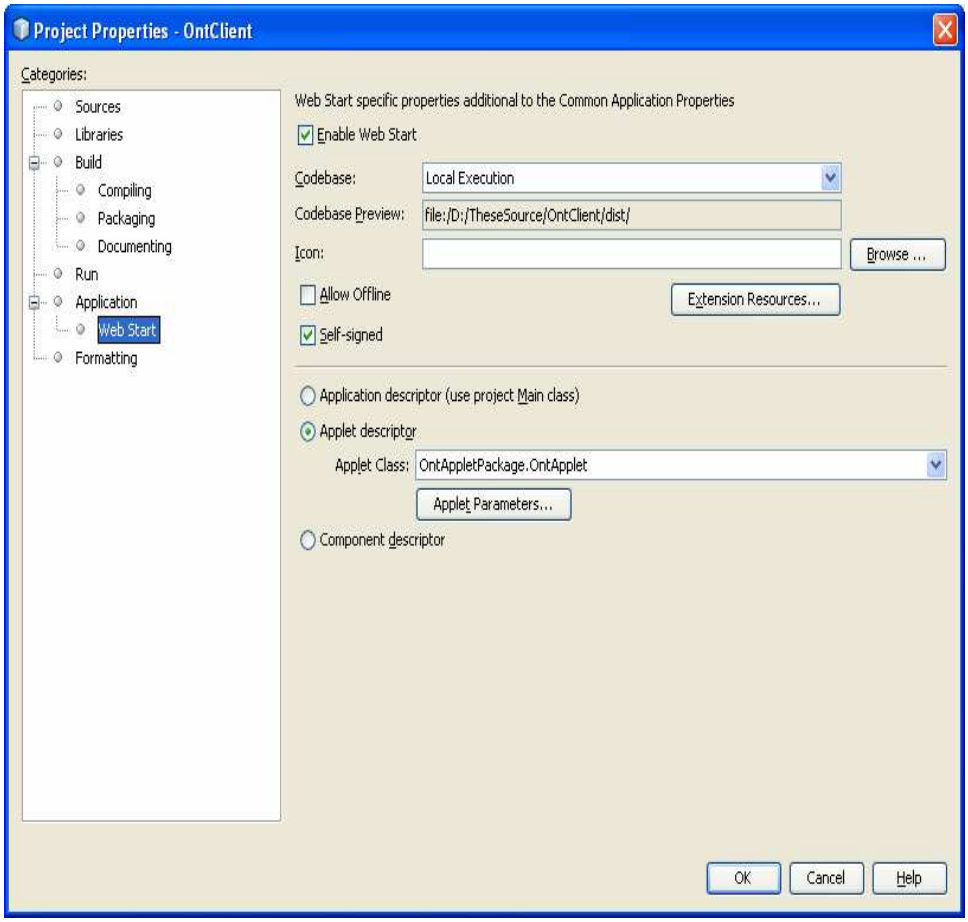
Hazırlanan .komut dosyasını çalıştırıldığında Ontoloji Yönetim Modülü masaüstü uygulaması olarak ekrana gelecektir.

Ontoloji Yönetim Modülünün Java Web Start uygulaması olarak kullanılması

Java Web Start, Sun firması tarafından geliştirilmiş olan ve ağ üzerinden uygulama, yükleme ve kullanmamızı sağlayan bir teknolojidir. Bu teknoloji web tarayıcısından ve web sunucusundan bağımsız çalıştığı için yalnızca Java çalışma ortamına gereksinim duyar. Java Web Start Uygulamaları, sunucudaki uygulama ile bilgileri içeren JNLP dosyalarını kullanarak çalışır ve istenirse tipik bir yazılım gibi

yerel bilgisayara kurulur. Kurulum yapılsın ya da yapılmasın, Java Web Start uygulamaları yerel bilgisayarda çalışır.

Bir Java projesinin Java Web Start uygulaması olarak kullanılabilmesi için proje özellikleri ayarlanmalıdır. Bu amaçla OntClient projesinin Web Start özellikleri, Netbeans çalışma ortamında, Şekil 17'de gösterildiği gibi ayarlanmıştır.



Şekil.17. Ontoloji Yönetim Modülünün Web Start Uygulaması olarak ayarlanması

Bu özellikler ayarlandıktan sonra proje derlendiğinde projenin "dist" dizininde "launch.jnlp" dosyası üretilir.

Hatırlanacağı gibi, "dist" dizini Tomcat'in "webapps" dizini altına "OntClient" adı ile kopyalanmış, bu sayede Ontoloji Yönetim Modülünü içeren web sayfası erişilebilir duruma getirilmişti. Bu web sayfasının adresi, sunucu "localhost", Tomcat erişim portu "8090" ve uygulama dizini "OntClient" olduğunda,

"http://localhost:8090/OntClient/ /index.html" olarak belirtilmişti.

“Launch.jnlp” dosyası ile “index.html” dosyası aynı dizinde olduğundan Java Web Start uygulamasının erişim adresi aşağıdaki gibi olacaktır:

“http://localhost:8090/OntClient/launch.jnlp”

6.4.3. Ontoloji Yönetim Modülünün Kısıtları

Ontoloji Yönetim Modülü, Ontoloji Servisine erişerek ontoloji ile ilgili temel işlemlerin yapılmasını sağlayan örnek bir modül olduğu için kimi kısıtlara sahiptir.

Bunlardan başlıcaları aşağıda sıralanmıştır:

- Sınıflar listelenirken yalnızca adlandırılmış sınıflar gözününde bulundurulur.
- Özellik oluştururken kaynak ve hedef sınıf belirtilmesi zorlanmıştır.
- Veri tipi özellikler için seçilebilecek tipler mantıksal (boolean), metin (string)ve sayısal (integer) tipler olarak kısıtlanmıştır.

6.5. Ontoloji Servisi ve İstemci Uygulamaların Konumsal İlişkisi

Bu çalışmada Ontoloji Servisi ile istemci uygulaması aynı sunucu üzerinde konumlandırılmıştır. Bununla birlikte Ontoloji Servisi ile istemci uygulaması başka sunucularda bulunabilir. Diğer bir deyişle istemci uygulamaları ile Ontoloji Servisi bağımsız modüllerdir ve farklı yerlerde konumlandırılabilirler. Nitekim, Ontoloji Yönetim Modülünün web sayfası olarak değil de yerel bilgisayar üzerinden çalıştırıldığı durumda, istemci uygulamanın sunucuya yüklenmesinin zorunlu olmadığı açıktır.

Çalışmada, Ontoloji Yönetim Modülünün Web servisinin konumundan bağımsız olarak çalışabilmesi, servisin konum bilgisinin applete parametre olarak geçilebilmesi sayesinde sağlanmıştır. Bu parametrenin değeri, Ontoloji Servisinin WSDL adresidir.

Ontoloji Yönetim Modülünün kullanım biçime bağlı olarak bu parametrenin applete geçirilmesi de farklı biçimlerde sağlanmıştır. Ontoloji Yönetim Modülünün

web sayfası olarak kullanıldığı durumda bu parametre (WSDLAddress), applet etiketleri içinde “param” deyimi ile geçirilmiş, masaüstü uygulaması için ise “Config.properties” dosyasında aynı isimli özelik kullanılmıştır.

7. SİSTEMİN TEST EDİLMESİ

Ontoloji Yönetim Sistemi'nin çalışması için sağlanması gereken koşullar şunlardır:

- Tomcat sunucusunun çalışması,
- Sunucuda yüklü Ontoloji Web Servisinin çalışması,
- Ontoloji Servisine erişen bir uygulamanın çalışması. Bu uygulama aşağıdakilerden biri olabilir:
    - Ontoloji Yönetim Modülünü içeren web sayfası,
    - Ontoloji Yönetim Modülünü kullanan web start uygulaması,
    - Ontoloji Yönetim Modülünü içeren masaüstü uygulaması.

Bu bölümde öncelikle Tomcat sunucusu ve Ontoloji Servisinin, sonrasında ise bu servise erişen uygulamaların testi yapılacaktır.

Tüm testlerde, yerel sunucu (localhost), 8090 portu kullanılacaktır. Ontoloji Servisinin sunucudaki dizini "OntService", Ontoloji Yönetim Modülünün dizini ise "OntClient" olarak belirlenmiştir.

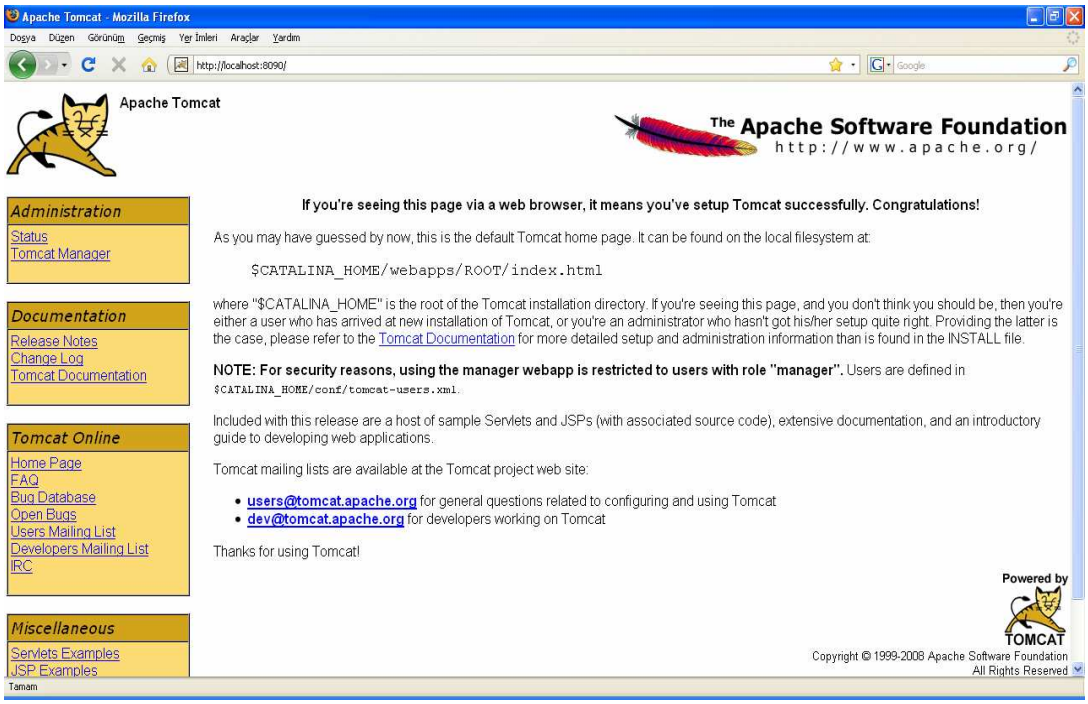
7.1. Ontoloji Servisinin Test Edilmesi

Servisin çalışabilmesi için öncelikle Tomcat sunucusuna erişim sağlanmalıdır. Bunun için aşağıdaki adres kullanılmıştır.

"http://localhost:8090/"

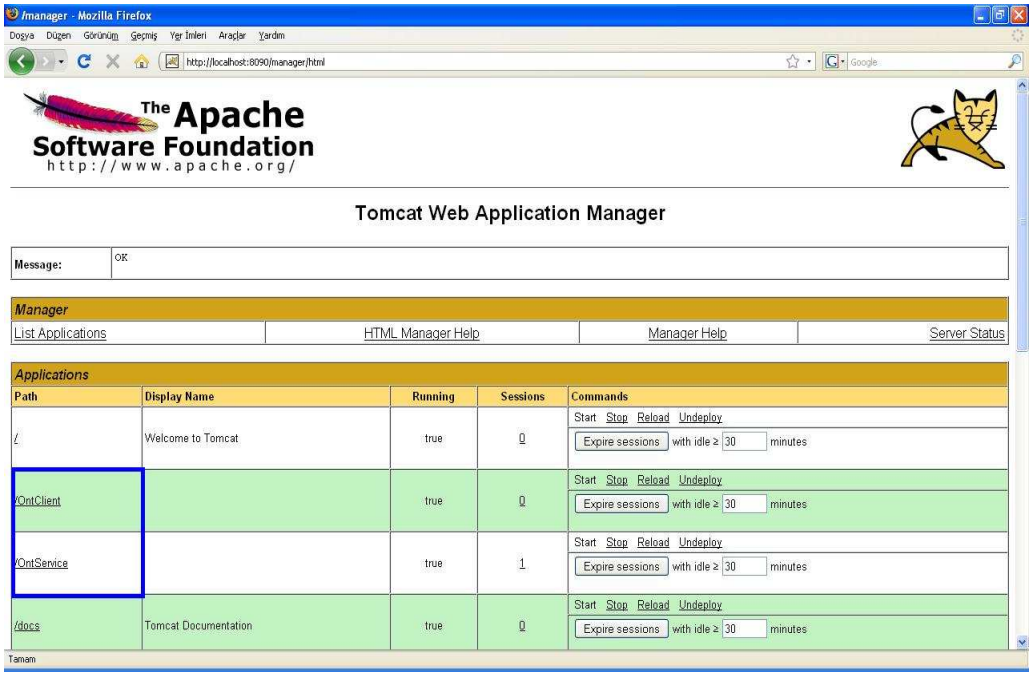
Belirtilen adrese erişim sağlandığında Tomcat sunucusunun ana sayfası ekrana gelmiştir.

Şekil 18.a, Tomcat sunucusunun ana sayfasını göstemektedir.



Şekil.18.a) Tomcat sunucusu

Bu sayfadaki Tomcat Manager bağlantısı tıklanarak yönetim ekranı açılabilir. Yönetim ekranı sayesinde Ontoloji Servisi ve Ontoloji Yönetim Modülünü içeren web uygulamasının sunucuya yüklenip yüklenmediği anlaşılabilir.



Şekil.18.b) Tomcat yönetim ekranı

Şekil 18.b'de mavi çerçeve içinde gösterildiği gibi, Ontoloji Yönetim Modülü (OntClient) ve Ontoloji Servisi (OntService) web uygulamaları, Tomcat'e yüklenmiş durumdadır.

Bu sayfada OntService bağlantısına tıklandığında, Şekil 18.c'de gösterildiği gibi, Ontoloji Servisi ana sayfası gelecektir:

Şekil.18.c) Ontoloji Servisi ana sayfası

Bu sayfadaki bağlantıların hedefleri ve adresleri ise aşağıda listelenmiştir:

- Ontoloji Servisinin WSDL içeriği:
  "http://localhost:8090/OntService/OntService?wsdl"

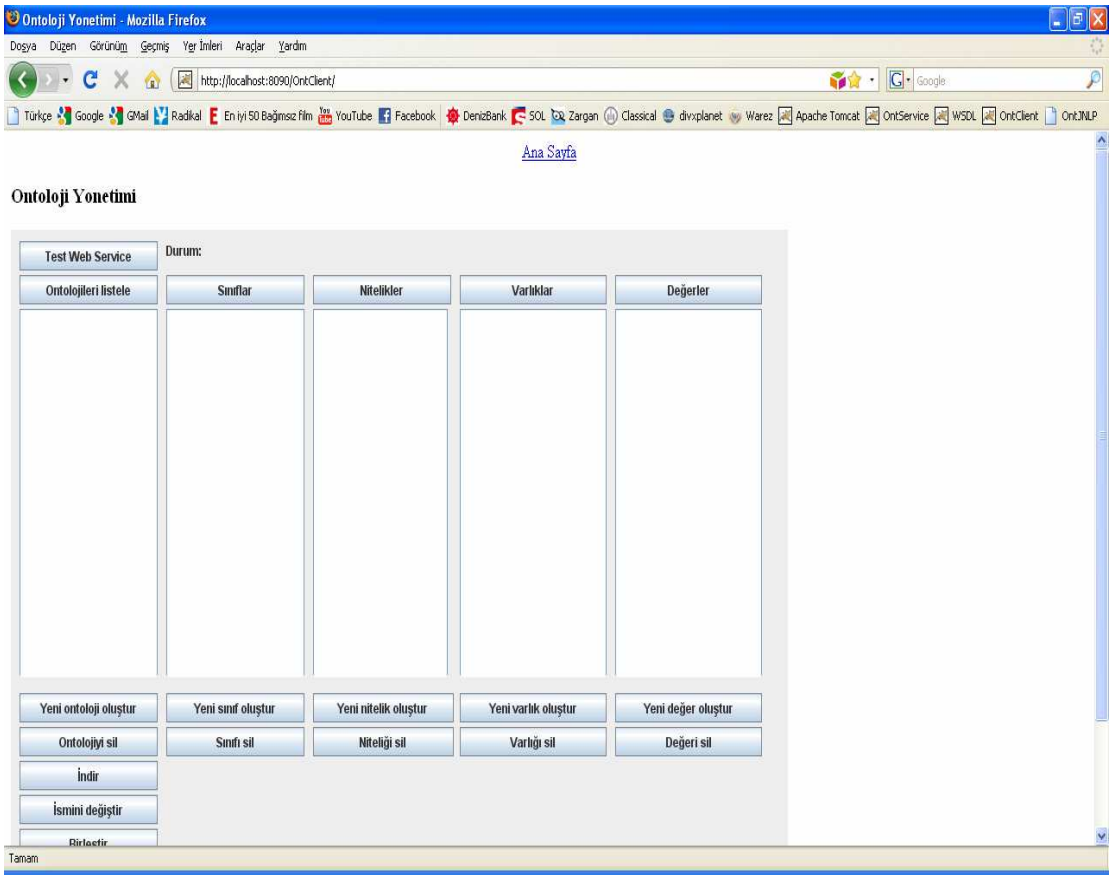
- Ontoloji Yönetimi Web Sayfası:

  "http://localhost:8090/OntClient/"

- Ontoloji Yönetimi Web Start uygulaması:
  "http://localhost:8090/OntClient/launch.jnlp"

7.2. Ontoloji Yönetim Sayfasının Test Edilmesi

Yukarıda belirtilen adrese erişim sağlandığında, Şekil 18.d'de gösterildiği gibi, Ontoloji Yönetim Modülünü içeren bir web sayfası ekrana gelecektir.



Şekil.18.d) Ontoloji Yönetim Sayfası

Bu sayfa, tasarımımıza uygun olarak Ontoloji Servisi ana sayfasına bir bağlantı ("Ana Sayfa") ve Ontoloji Yönetim Modülü olarak geliştirilen appletten oluşmaktadır.

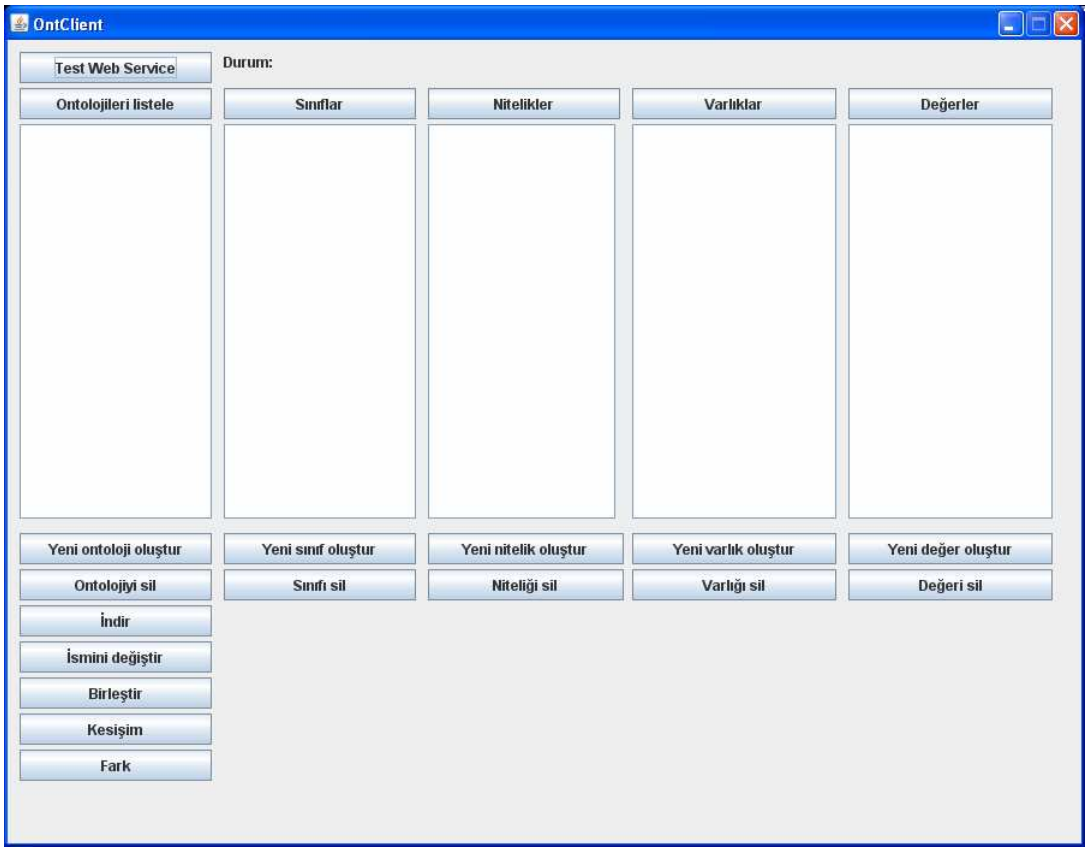
7.3. Ontoloji Yönetimi Web Start Uygulamasının Test Edilmesi

Ontoloji Servisi ana sayfasındaki ilgili bağlantı kullanıldığında ya da doğrudan "http://localhost:8090/OntClient/launch.jnlp" adresine erişim sağlandığında, Java Çalışma Ortamı, sunucudaki "OntClient" uygulaması ile ilgili olarak, sayısal imza kullanılmamış bir uygulamaya erişim yaptığımız konusunda bizi uyarır ve uygulamayı çalıştırıp çalıştırmayacağımızı sorar. Şekil 18.e, bu güvenlik uyarısını göstermektedir.

Şekil.18.e) Ontoloji Yönetim Modülü yüklenirken sayısal imzanın denetlenmesi

"Çalıştır" komutu verildiğinde, Ontoloji Yönetim Modülü yerel bilgisayar üzerinde çalıştırılır. Java ayarları uygun bir biçimde ayarlanmışsa uygulama tipik bir yazılım gibi yerel bilgisayara yüklenir, uygulama programlar menüsüne eklenir ve kısayolu masaüstüne konulur. Bu işlemlerden sonra Ontoloji Yönetimi Web Start uygulaması, herhangi bir program gibi çalıştırılabilir.

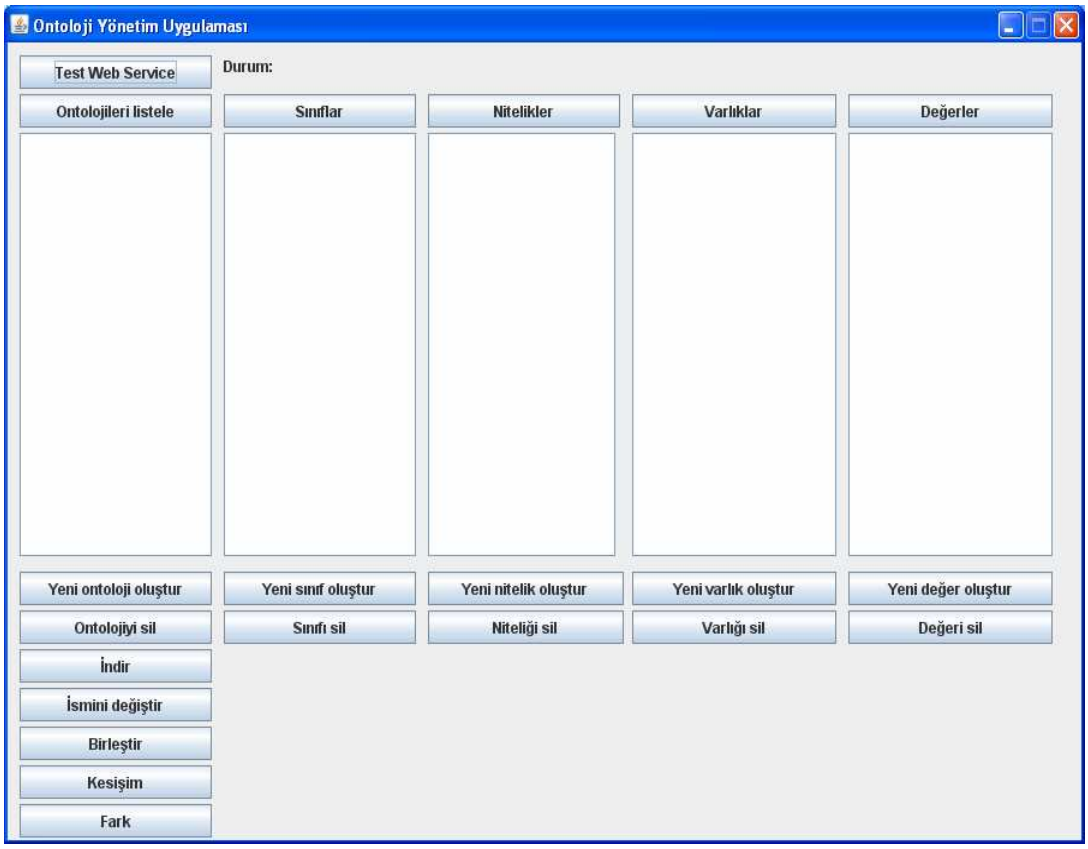
Şekil 18.f, Ontoloji Yönetimi Web Start uygulamasını göstermektedir.



Şekil.18.f) Ontoloji Yönetimi Web Start uygulaması

7.4. Ontoloji Yönetimi Masaüstü Uygulamasının Test Edilmesi

Bu işlem için daha önce hazırlanan komut dosyasını çalıştırmak yeterli olacaktır. Şekil 18.g, Ontoloji Yönetimi Masaüstü Uygulaması'nı göstermektedir.



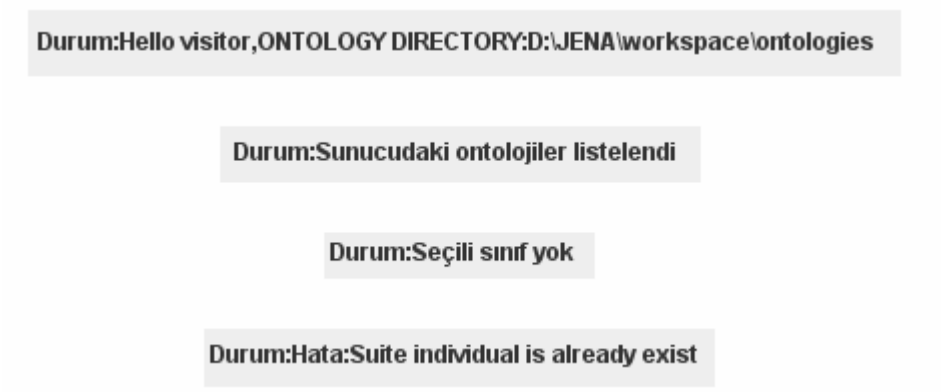
Şekil.18.g) Ontoloji Yönetimi Masaüstü uygulaması

8. ONTOLOJİ YÖNETİM MODÜLÜNÜN KULLANIMI

8.1. Genel Bilgiler

Durum bilgisi

Yönetim modülünde, durum bilgisi sayesinde, kullanıcı yapılan işlemlerin ne aşamada olduğu ya da işlemin sonucu hakkında bilgilendirilir. Durum bilgisi kullanıcıyı uyarmak için de kullanılır. Şekil 19'da durum bilgilerine ait örnekler bulunabilir.



Şekil.19. Ontoloji Yönetim Modülünde örnek durum bilgileri

Kullanılan listeler

Ontoloji Yönetim Modülünde soldan sağa doğru, ontoloji sıradüzeni ile uyumlu olarak, ontoloji, sınıf, nitelik, varlık ve değer listeleri kullanılır. Her bir listenin doldurulması için kullanılması gereken buton listelerin hemen üstünde yer alır. Her bir listenin altında ise listenin ilişkili olduğu ontoloji kavramları (ontoloji, sınıf, nitelik, varlık, değer) ile ilgili komutlar yer almaktadır.

Kullanılan listeler ontolojiyi değiştiren her hangi bir komuttan sonra (ekleme, silme, isim değiştirme) kendiliğinden yenilenir. Soldaki bir liste yenilendiğinde bu listenin sağındaki listeler boşaltılacaktır. Örneğin ontoloji listesi yenilendiğinde, ontoloji listesinin sağında yer alan sınıf, nitelik, varlık ve değer listelerinin tümü boşaltılacaktır. Aynı biçimde sınıf listesi "Sınıflar" butonu tıklanarak yenilendiğinde, bu listenin sağındaki nitelik, varlık ve değer listesi yenilenecektir.

Boşluk içeren kaynak (resource) isimlerinin düzeltilmesi

Yönetim modülü, kullanıcının sınıf, nitelik ve varlık ismi olarak belirlediği boşluk içeren metinleri, boşluk karakteri yerine "_" karakteri kulanarak düzeltir. Örneğin, kullanıcı yeni oluşturacağı bir sınıf için "Related Age" ismini belirtmesine karşın sistem "Related_Age" isimli bir sınıf oluşturacaktır.

8.2. Web Servisi Bağlantı Testi

Yönetim modülü web servisi ile kullanıcı arasındaki arayüz olduğu için öncelikle web servisine başarılı bir biçimde bağlanılması gerekir. Bunun için "Test Web Service" butonu kullanılır.

Web servisine erişilemediği durumda ilgili hata mesajı durum bilgisi ("Durum:Hata:java.net.ConnectException:Connection refused:connect") olarak ekrana yazılır. Web servisine başarılı bir biçimde bağlanıldığında ise durum bilgisi aşağıdaki gibi olacaktır:

"Durum:Hello visitor, ONTOLOGY DIRECTORY:[Ontoloji Dizini]"

8.3. Ontolojilerle İlgili İşlemler

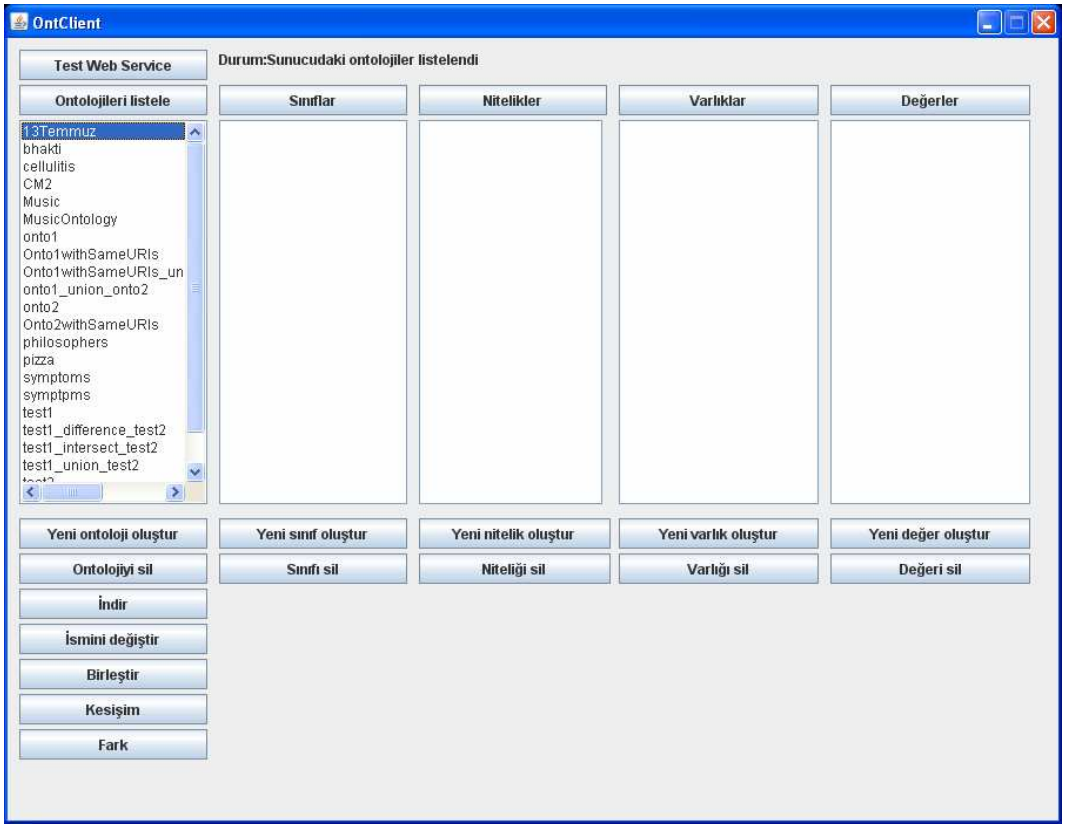
Web servisi testi başarılı olduğu durumda ontolojilerle ilgili aşağıdaki işlemler yapılabilir:

- Ontolojilerin listelenmesi,
- Yeni ontoloji oluşturulması,
- Seçili ontolojinin silinmesi,
- Seçili ontolojinin isminin değiştirilmesi,
- Seçili ontoloji ile bir başka ontolojinin birleştirilmesi,
- Seçili ontoloji ile bir başka ontolojinin kesiştirilmesi,
- Seçili ontoloji ile bir başka ontolojinin farkının alınması.

Ontolojilerin listelenmesi

Ontolojilerin listelenmesi için "Ontolojileri listele" butonu kullanılır.

Şekil 20.a'daki ekran görüntüsü "Ontolojileri listele" metodunun çağrılması sonucu elde edilmiştir.

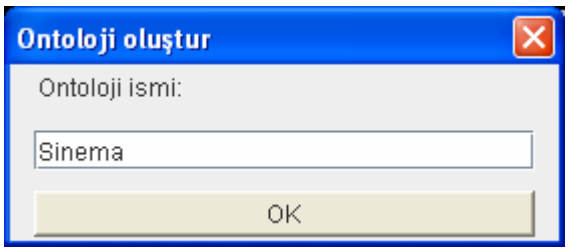


Şekil.20.a) Ontolojilerin listelenmesi

Yeni ontoloji oluşturma

Yeni ontoloji oluşturmak için "Yeni ontoloji oluştur" butonu kullanılır.

Şekil 20.b, “Yeni ontoloji oluştur” komutu verildikten sonra ekrana gelen diyalog ekranını göstermektedir.



Şekil.20.b) Yeni ontoloji oluşturma

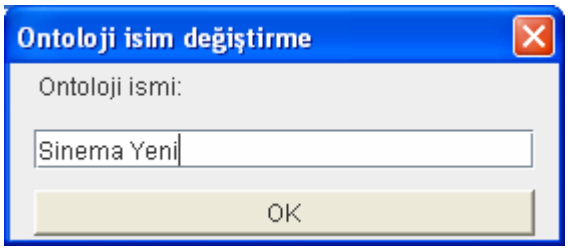
Ontoloji ismi girildikten ve bilgi giriş ekranında onay butonuna tıklandıktan sonra ontoloji oluşturulacaktır.

Ontoloji silme

Bir ontolojiyi silmek için ise silinmesi istenen ontolojinin seçilmesi ve “Ontolojiyi sil” komutunun verilmesi yeterlidir.

Ontoloji ismini değiştirme

Ontoloji ismini değiştirmek için, ismi değiştirilmek istenen ontoloji seçilir ve “İsmini değiştir” komutu verilir. Bu işlemlerden sonra Şekil 20.c’de gösterilen bilgi giriş ekranı görüntülenir.

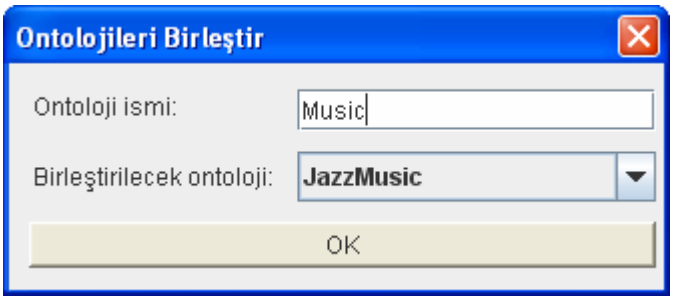


Şekil.20.c) Ontoloji ismini değiştirme

Bilgi giriş ekranı onaylandıktan sonra seçili ontolojinin ismi değiştirilecektir.

İki ontolojiyi birleştirme

İki ontolojiyi birleştirmek için, birleştirilmek istenen ontolojilerden biri seçilir ve "Birleştir" komutu verilir. Diğer ontolojinin seçilmesi için görüntülenecek olan bilgi giriş ekranı Şekil 20.d'de gösterilmiştir.



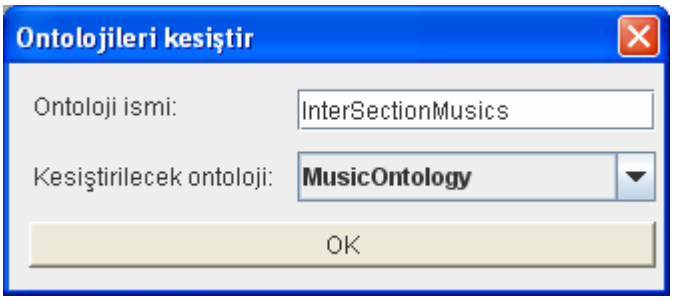
Şekil.20.d) İki ontolojiyi birleştirme

Bu ekranda ontoloji ismi bölümüne, iki ontolojinin birleştirilmesi sonucu elde edilecek ontolojinin ismi girilir. "Birleştirilecek ontoloji" listesi kullanılarak ise birleştirilmek istenen ontoloji seçilir.

Burada dikkat edilmesi gereken nokta, birleştirme işleminde iki kaynağın (sınıf, özellik, varlık yada değer) aynı nesne olarak değerlendirilmesi için bu nesnelerin URI değerlerinin aynı olması gerektiğidir. Örneğin iki sınıfın yerel isimleri (local name) aynı olsa bile bu iki sınıf birbirinden farklı olarak değerlendirilir. Bu durumda iki ayrı sınıf olarak ele alınıp, yeni oluşturulan ontolojide (birleştirme sonucu elde edilen ontoloji) aynı adlı iki sınıf bulunacaktır. Bu durum kesişim ve fark işlemleri için de geçerlidir.

İki ontolojiyi kesiştirme

İki ontolojiyi kesiştirmek için, kesiştirilmek istenen ontolojilerden biri seçilir ve "Kesişim" komutu verilir. Diğer ontolojinin seçilmesi için görüntülenecek olan bilgi giriş ekranı Şekil 20.e'de gösterilmiştir.

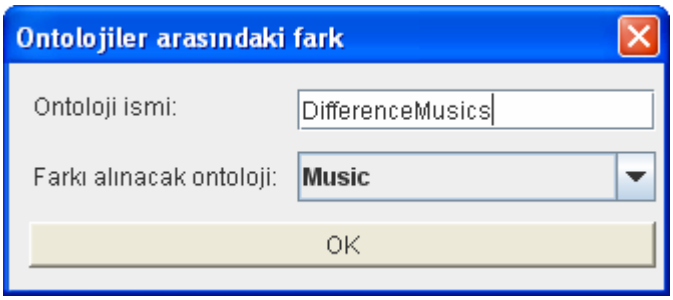


Şekil.20.e) İki ontolojiyi kesiştirme

Bu ekranda ontoloji ismi bölümüne, iki ontolojinin kesiştirilmesi sonucu elde edilecek ontolojinin ismi girilir. Kesiştirilecek ontoloji bölümü kullanılarak ise kesiştirilmek istenen ontoloji belirtilir.

İki ontoloji arasındaki farkların bulunması

İki ontoloji arasındaki farkları bulmak için, ontolojilerden biri seçilir ve "Fark" komutu verilir. Diğer ontolojinin seçilmesi için görüntülenecek olan bilgi giriş ekranı Şekil 20.f'de gösterilmiştir.



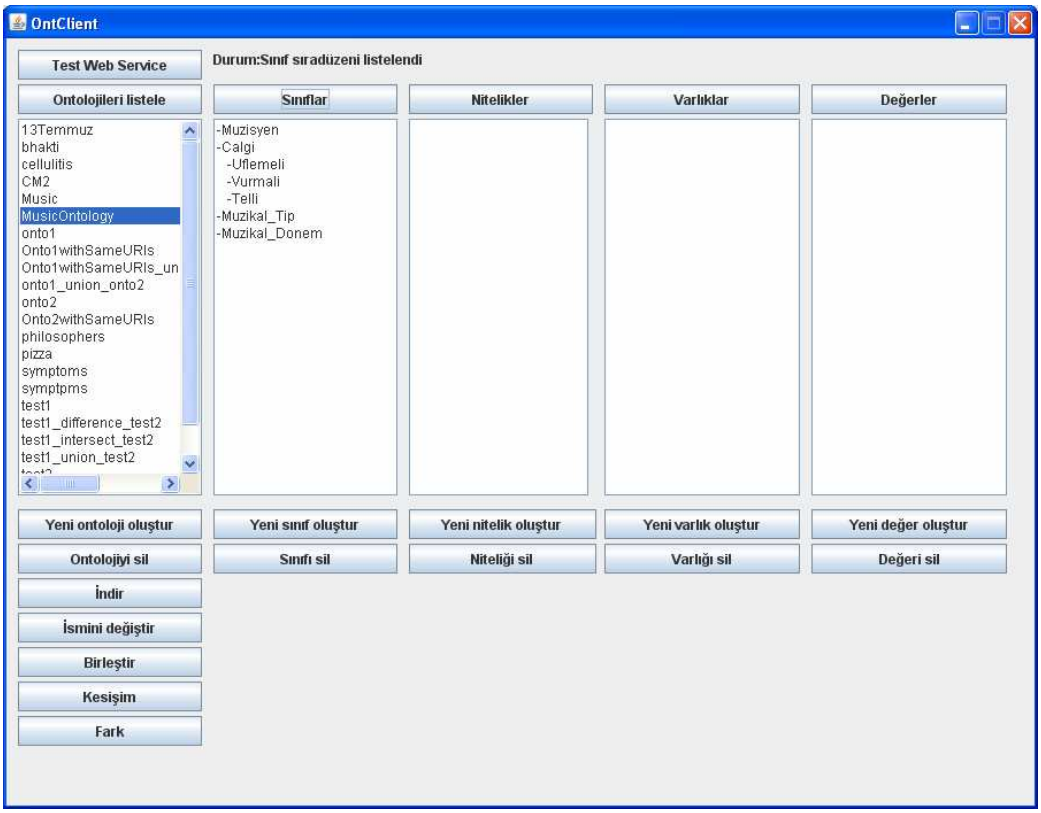
Şekil.20.f) Ontolojiler arasındaki farkların bulunması

Bu ekranda ontoloji ismi bölümüne, iki ontolojinin farkının alınması sonucu elde edilecek ontolojinin ismi girilir. Farkı alınacak ontoloji bölümünden ise farkı alınmak istenen ontoloji seçilir.

8.4. Sınıflarla İlgili İşlemler

Sınıfların listelenmesi

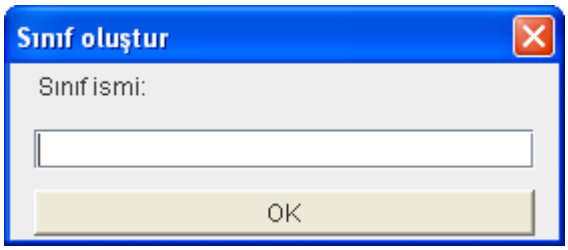
Sınıfların listelenmesi için "Sınıflar" butonu kullanılır. Sınıfların listelenmesi sonucu bu butonun altındaki sınıf listesi, seçili ontolojinin içerdiği sınıfları listeleyecektir. Şekil 21.a'daki görüntü sınıflar listelendikten sonra elde edilmiştir:



Şekil.21.a) Sınıfların listelenmesi

Yeni sınıf oluşturma

Seçili ontolojiye ait yeni bir sınıf oluşturmak için "Yeni sınıf oluştur" butonu kullanılır. Komut verildikten sonra görüntülenecek bilgi giriş ekranı Şekil 21.b'de gösterilmiştir.



Şekil.21.b) Yeni sınıf oluşturma

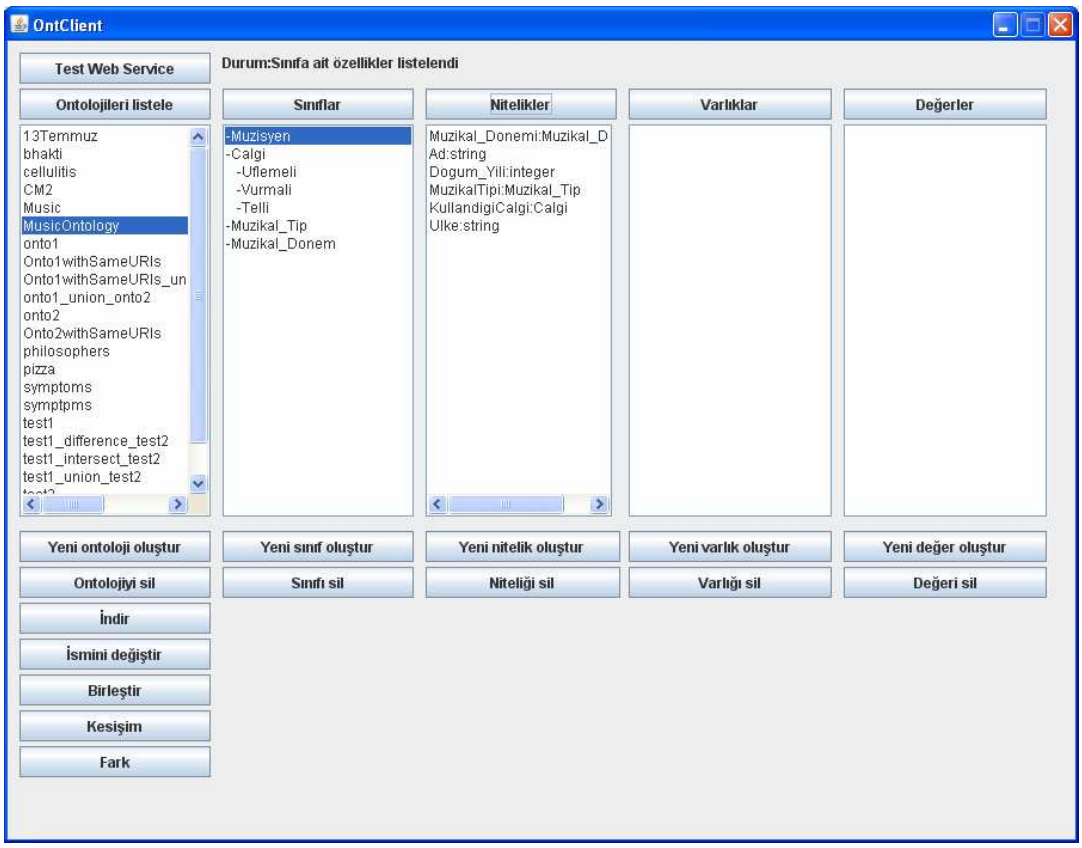
Sınıf silme

Seçili sınıfı silmek için "Sınıfı sil" butonu kullanılır. Sınıf silindikten sonra, bu sınıfa ait tüm nitelik, varlık ve değerler de silinecektir.

8.5. Niteliklerle İlgili İşlemler

Niteliklerin listelenmesi

Seçili sınıfın niteliklerinin silinmesi için “Nitelikler” butonu kullanılır. Komut verildikten sonra bu butonun altındaki nitelik listesi seçili sınıfa ait niteliklerle doldurulacaktır. Şekil 22.a’daki görüntü nitelikler listelendikten sonra elde edilmiştir:



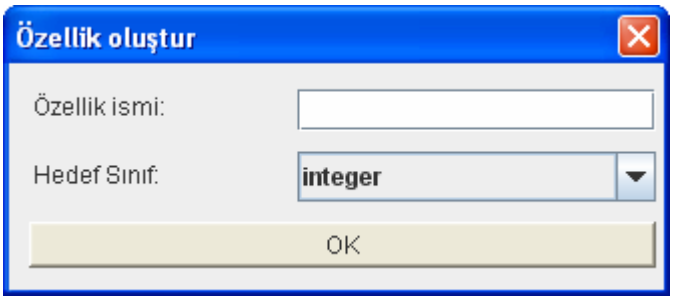
Şekil.22.a) Niteliklerin listelenmesi

Nitelik listesindeki nitelikler aşağıdaki biçemde listede yer alacaktır:

[Nitelik ismi]:[Niteliğin tipi]

Yeni nitelik oluşturma

Seçili sınıfa ait yeni bir nitelik oluşturmak için “Yeni nitelik oluştur” butonu kullanılır. Komut verildikten sonra görüntülenecek bilgi giriş ekranı Şekil 22.b’de gösterilmiştir.



Şekil.22.b) Yeni nitelik oluşturma

Bu işlem için niteliğin ismi ve tipi bu giriş ekranı sayesinde belirtilir.

Yeni nitelik için gereken parametre ve bu parametrelerin değerleri şunlardır:

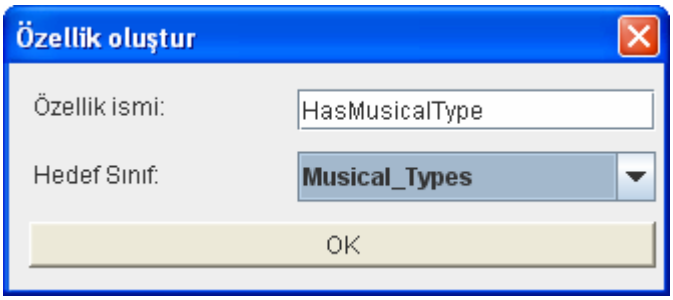
Özelliğin (niteliğin) kaynak tipi (domain type)**:** Seçilen sınıf

Özelliğin ismi**:** Girilen isim

Özelliğin hedef tipi (Range type): Hedef sınıf olarak seçilen tip.

Ontoloji Yönetim Modülü, hedef tipi olarak üç temel tipi (sayısal, metin, mantıksal) ile ontolojideki diğer sınıfların seçimini destekler. Diğer bir deyişle, hedef tipi olarak sayısal, metin ve mantıksal tiplerin yanı sıra ontolojideki bir başka sınıf da kullanılabilir.

Şekil 22.c’deki ekran görüntüsü niteliklerin hedef tipinin belirlenmesini göstermektedir.



Şekil.22.c) Niteliklerin hedef tipinin belirlenmesi

Niteliğin silinmesi

Seçili sınıfa ait bir niteliğin silinmesi için, silinmek istenen nitelik seçilir ve "Niteliği sil" butonu kullanılır. Nitelik silindikten sonra bu nitelik için girilmiş tüm değerler de silinir.

8.6. Bireysel Varlıklarla İlgili İşlemler

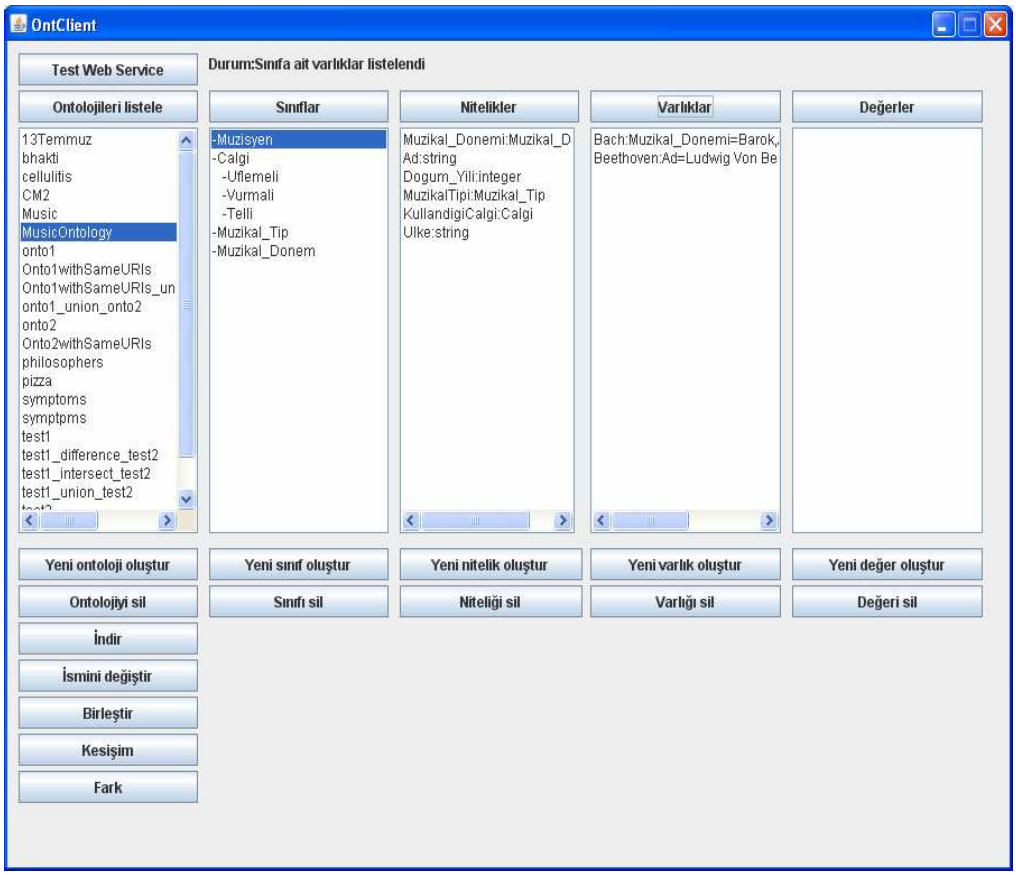
Varlıkların listelenmesi

Seçili sınıfa ait bireysel varlıkların listelenmesi için "Varlıklar" butonu kullanılır. Komut verildikten sonra bu butonun altındaki varlık listesi, seçili sınıfa ait bireysel varlıklar ile doldurulacaktır.

Varlık listesindeki varlıklar aşağıdaki biçemde listede yer alacaktır:

[Varlık ismi]:[Nitelik1:Değer1], [Nitelik2:Değer2], …,[Nitelik N:Değer N]

Şekil 23.a, varlıkların listelenmesini göstermektedir.



Şekil.23.a) Varlıkların listelenmesi

Yeni varlık oluşturma

Seçili sınıfa ait yeni bir varlık oluşturmak için “Yeni varlık oluştur” butonu kullanılır.Komut verildikten sonra görüntülenecek olan bilgi giriş ekranı Şekil 42’de gösterilmiştir.

Şekil.23.b) Yeni varlık oluşturma

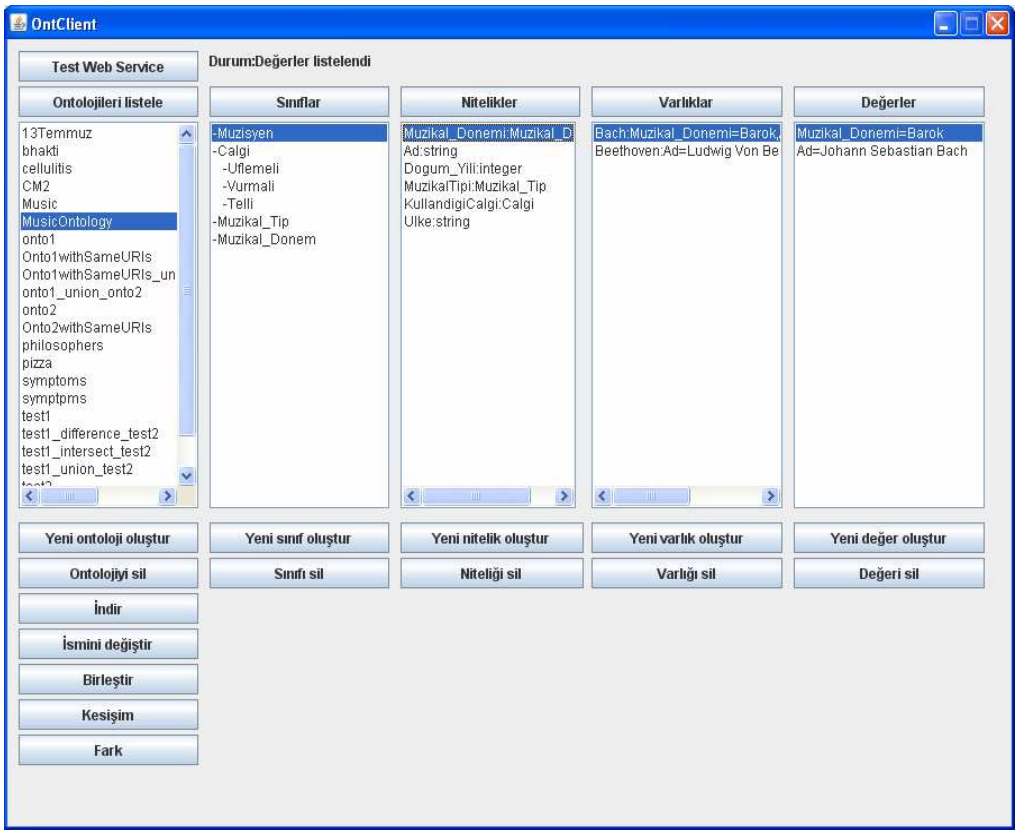
Varlığın silinmesi

Seçili bireysel varlığı silmek için silinmek istenen varlık seçilir ve “Varlığı sil” butonu kullanılır.

8.7. Değerlerle İlgili İşlemler

Değerlerin listelenmesi

Seçili varlığın değerlerini görmek için “Değerler” butonu kullanılır. Komut verildikten sonra değerler listesi, seçili varlığın nitelikleri (seçili sınıfın nitelikleri) için girilmiş olan değerleri gösterir. Şekil 24.a’daki görüntü değerler listelendikten sonra elde edilmiştir.



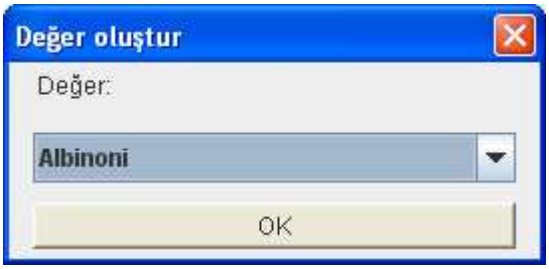
Şekil.24.a) Değerlerin listelenmesi

Değerleri listesindeki değerler listede aşağıdaki biçemde yer alır:

[Nitelik ismi]=[Değer]

Yeni değer oluşturma

Seçili varlığa ait yeni bir değer oluşturmak için bir nitelik ve bir varlık seçilmelidir. Gerekli seçimler yapıldıktan sonra “Yeni değer oluştur” komutu verilir. Komut verildikten sonra görüntülenecek olan bilgi giriş ekranı Şekil 24.b’de gösterilmiştir.



Şekil.24.b) Yeni değer oluşturma

Yeni değerin oluşturulması için gereken parametrelerin değerleri şunlar olacaktır:

Varlık: Seçili varlık

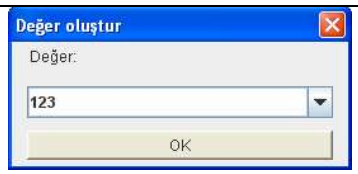
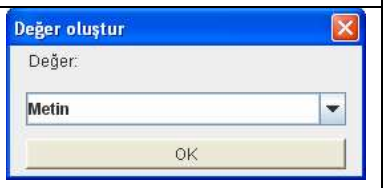
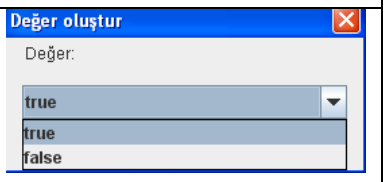
Nitelik: Seçili nitelik

Değer: Bilgi giriş ekranında seçilen yada girilen değer.

Yeni değer oluşturmak için görüntülenen bilgi giriş ekranı seçili niteğin tipine göre değişecektir.

Tablo 3, niteliğin tipine göre görüntülenecek olan bilgi giriş ekranlarını göstermektedir:

Tablo.3. Niteliğin tipine göre görüntülen giriş ekranları

Veri tipi	Açıklama	Bilgi giriş ekranı
Sayısal	Sayısal değer girilmelidir	
Metin	Metin girilmelidir.	
Mantıksal	true yada false değeri seçilmelidir	
Sınıf tipi	Seçili niteliğin hedef tipi bir sınıf ise, seçme kutusu sınıfa ait varlıklar ile doldurulur. Yeni değer olarak bu seçme kutusundan bir varlık seçilmelidir.	

9. SONUÇ

Bu çalışmada Anlamsal Ağ ve ontoloji kavramları üzerinde durulmuş, Anlamsal Ağ için önemli bir konu olan, ontolojilerin uzaktan yönetimi için uygulanabilirlik, standartlara uygunluk, başarım, esneklik ve modülerlik bakımından en uygun çözüm aranmıştır. Ontoloji işlemlerinin bir web servisi tarafından yapılması gerektiği, yukarıdaki ölçütlere göre karar verilmiştir. Bu servise erişecek olan istemci modülü için de benzer ölçütler kullanılmış ve bu modülün bir applet olmasına karar verilmiştir.

Çalışmanın uygulama bölümünde ise; Ontoloji Servisi ile Ontoloji Yönetim Modülünden oluşan sistemin temel bileşenlerinin belirlenmesine, web servisi ve istemcinin geliştirilmesine, bunların bir web sunucusuna yüklenmesine ve sistemin değişik biçimlerde kullanılmasına ilişkin örneklere yer verilmiştir.

Çalışmanın şu anki kapsamı bir web sitesinin ya da uygulamasının belirli bir çalışma alanında, ontoloji oluşturma, değiştirme ve genişletmesine olanak sağlamaktadır. Geliştirilen Ontoloji Servisi kullanılarak farklı ontolojiler üzerinde çalışan farklı web sayfaları ya da uygulamaları geliştirilebilir. Sağlanan modüler yapıdan dolayı, bu uygulamaların ontoloji ile ilgili herhangi bir kodlama içermesine gerek yoktur.

Bu çalışma, ontoloji ile ilgili temel işlemleri kapsadığı için Ontoloji Servisinin sunabileceği birçok işlev (sorgulama, çıkarım, görselleştirme, biçimsel dönüştürme, versiyonlama) eksik kalmıştır. Bu işlevler, Ontoloji Servisine eklenerek kapsam genişletilebilir.

Sistemin canlı ortamda kullanıldığı durumda, ontoloji yönetim sistemine eklenmesi zorunlu olan kimi işlevler de vardır. Bunlardan ilki, sisteme girişin denetlenmesidir. Web servisine erişim belirli kullanıcılar tarafından yapılacağı için bu zorunludur. Diğer bir konu ise kullanıcıların yetki düzeylerine göre servisi kullanabilmesidir. Örneğin, ontoloji silme yetkisi yalnızca yönetici tipindeki kullanıcılara verilebilir. Canlı ortamda sağlanması gerekenlerden biri de ontolojiler için kilitleme (lock) mekanizmasıdır. Farklı kullanıcılar, aynı anda aynı ontoloji üzerinde değişiklik yapabileceği için bu mekanizmanın geliştirilmesi zorunludur. Bu

mekanizma sayesinde ontolojiler, değiştirme işlemleri için kilitlenebilir ve geçerli kullanıcı, kilidi kaldırdıktan sonra bir başka kullanıcı aynı ontoloji üzerinde işlem yapabilir. Diğer bir konu ise, ontolojiler için tutarlılık denetimidir. Herhangi bir değişiklikten sonra ontolojinin tutarlılığının korunup korunmadığı denetlenebilir, korunmuyorsa yapılan işlem iptal edilebilir.

Bu çalışmada, günümüzün İnternet dünyasının iki temel konusu olan Anlamsal Ağ ve Web Servisleri birlikte ele alınmış ve Web Servisi teknolojisi Anlamsal Ağ'ın temel bileşeni olan ontolojilerin yönetimi için kullanılmıştır. Ontolojilerin paylaşımı ve yönetimi için web servislerini kullanan benzer çalışmalar (bkz. 5.2.2 Ontoloji Servisi Örnekleri) incelenerek farklı olanaklar keşfedilebilir ve farklı yaklaşımlar geliştirilebilir.

KAYNAKLAR

[1] Berners-Lee, T., Hendler, J.,Lassila, O., "The Semantic Web". Scientific American Magazine, Mayıs, 2001

[2] Daconta, M.C., Obrst, L.J., Smith, K.T. *The Semantic Web*, 2003

[3] Gómez-Pérez, A., Fernández-López, M.., Corcho,O., "Ontological Engineering: with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web", Springer, 2004.

[4] Horrocks, I., Bechhofer, S., *Reasoning with Expressive Description Logics-Theory and Practice*, http://www.cs.man.ac.uk/~horrocks/Slides/Innsbruck-tutorial/pt1-swintro.ppt

[5] W3C, OWL Web Ontology Language, http://www.w3.org/TR/owl-features/ , 2010

[6] Wikipedia, *Semantic Web*, http://en.wikipedia.org/wiki/Semantic_Web

[7] Gruber, T.R., *A translation approach to portable ontology specifications*, Knowl. Acquis. 5, 199-220, 1993.

[8] Uschold, M., Gruninger, M., *Ontologies and semantics for seamless connectivity*, SIGMOD Rec. 33, 58-64, 2004.

[9] Davis, M., *Semantic Wave 2006*, http://www.scribd.com/doc/27811/SemWave2006P1

[10] W3C, June 2003, http://www.w3.org, 2008

[11] Allemang, D., Hendler, J., *Semantic Web for the Working Ontologist*, Morgan Kaufmann, 2008.

[12] SourceForge, *Jena – A Semantic Web Framework for Java*, http://jena.sourceforge.net/

[13] W3C, *Web of Services*, http://www.w3.org/standards/webofservices/, 2010

[14] W3Schools, *A Soap Example*, http://www.w3schools.com/SOAP/soap_example.asp

[15] Tutorials Point, *WSDL Document Example*, http://www.tutorialspoint.com/wsdl/wsdl_example.htm

[16] Ontology Lookup Service, http://www.ebi.ac.uk/ontology-lookup/

[17] Ontology Lookup Service, WSDL Documentation, http://www.ebi.ac.uk/ontology-lookup/WSDLDocumentation.do

[18] Lacasta, J., Muro, P.R.,Nogueras, J., *Web Ontology Service, A Key Component of a Spatial Data Infrastructure*, http://www.ec-gis.org/Workshops/11ec-gis/papers/303lacasta.pdf

[19] Heydari, N., Mansourian, A., Taleai, M., Fallahi, G.R., *Ontology-based GIS web service for increasing semantic interoperability among organizations involving drilling in city of Tehran*, http://www.gsdi.org/gsdiconf/gsdi11/papers/pdf/163.pdf

EK-1: ÖRNEK ONTOLOJİNİN OWL İÇERİĞİ

```
      <rdf:RDF
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xmlns:j.0="http://www.semanticweb.org/ontologies/MusicOntology
.owl#"
      xmlns:owl="http://www.w3.org/2002/07/owl#"
      xmlns:daml="http://www.daml.org/2001/03/daml+oil#"
      xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
      <owl:Ontology
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl"/
>
      <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#M
uzikal_Donem"/>
      <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#M
uzikal_Tip"/>
        <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#T
elli">
          <rdfs:subClassOf>
            <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#C
algi"/>
          </rdfs:subClassOf>
        </owl:Class>
        <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#M
uzisyen"/>
        <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#V
urmali">
          <rdfs:subClassOf
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Calgi"/>
        </owl:Class>
        <owl:Class
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#U
flemeli">
          <rdfs:subClassOf
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Calgi"/>
        </owl:Class>
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#M
uzikal_Donemi">
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Donem"/>
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
        </owl:ObjectProperty>
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#o
rnekMuzisyen">
```

```
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Tip"/>
        </owl:ObjectProperty>
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#M
uzikalDonemi">
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Tip"/>
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Donem"/>
        </owl:ObjectProperty>
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#i
cerdigiMuzisyen">
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Donem"/>
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
        </owl:ObjectProperty>
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#K
ullandigiCalgi">
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Calgi"/>
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
        </owl:ObjectProperty>
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#M
uzikalTipi">
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Tip"/>
        </owl:ObjectProperty>
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#K
ullanlanDonem">
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Donem"/>
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Calgi"/>
        </owl:ObjectProperty>
        <owl:ObjectProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#K
ullananMuzisyen">
```

```
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Calgi"/>
          <rdfs:range
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
        </owl:ObjectProperty>
        <owl:DatatypeProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#A
d">
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
          <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        </owl:DatatypeProperty>
        <owl:DatatypeProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#B
itisYili">
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Donem"/>
          <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
        </owl:DatatypeProperty>
        <owl:DatatypeProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#B
aslangicYili">
          <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzikal_Donem"/>
        </owl:DatatypeProperty>
        <owl:DatatypeProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#U
lke">
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
          <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        </owl:DatatypeProperty>
        <owl:DatatypeProperty
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#D
ogum_Yili">
          <rdfs:range
rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
          <rdfs:domain
rdf:resource="http://www.semanticweb.org/ontologies/MusicOntology.ow
l#Muzisyen"/>
        </owl:DatatypeProperty>
        <j.0:Muzisyen
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#B
ach">
          <j.0:Ad
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Johann Sebastian Bach</j.0:Ad>
          <j.0:Muzikal_Donemi>
```

```
            <j.0:Muzikal_Donem
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#B
arok"/>
          </j.0:Muzikal_Donemi>
        </j.0:Muzisyen>
        <j.0:Muzisyen
rdf:about="http://www.semanticweb.org/ontologies/MusicOntology.owl#B
eethoven">
          <j.0:Ad
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Ludwig Von Beethoven</j.0:Ad>
        </j.0:Muzisyen>
      </rdf:RDF>
```

EK-2: ONTOLOJİ SERVİSİNİN WSDL İÇERİĞİ

```
      <?xml version='1.0' encoding='UTF-8'?>
      <definitions xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
xmlns:wsp="http://www.w3.org/ns/ws-policy"
xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://OntService/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="http://OntService/" name="OntServiceService">
      <types>
      <xsd:schema>
      <xsd:import namespace="http://jaxb.dev.java.net/array"
schemaLocation="http://localhost:8090/OntService/OntService?xsd=1"
/>
      </xsd:schema>
      <xsd:schema>
      <xsd:import namespace="http://OntService/"
schemaLocation="http://localhost:8090/OntService/OntService?xsd=2"
/>
      </xsd:schema>
      </types>
      <message name="addProperty">
      <part name="parameters" element="tns:addProperty" />
      </message>
      <message name="addPropertyResponse">
      <part name="parameters" element="tns:addPropertyResponse" />
      </message>
      <message name="removeProperty">
      <part name="parameters" element="tns:removeProperty" />
      </message>
      <message name="removePropertyResponse">
      <part name="parameters" element="tns:removePropertyResponse"
/>
      </message>
      <message name="hello">
      <part name="parameters" element="tns:hello" />
      </message>
      <message name="helloResponse">
      <part name="parameters" element="tns:helloResponse" />
      </message>
      <message name="removeClass">
      <part name="parameters" element="tns:removeClass" />
      </message>
      <message name="removeClassResponse">
      <part name="parameters" element="tns:removeClassResponse" />
      </message>
      <message name="addMember">
      <part name="parameters" element="tns:addMember" />
      </message>
      <message name="addMemberResponse">
      <part name="parameters" element="tns:addMemberResponse" />
      </message>
      <message name="removeMember">
      <part name="parameters" element="tns:removeMember" />
      </message>
```

```
      <message name="removeMemberResponse">
      <part name="parameters" element="tns:removeMemberResponse" />
      </message>
      <message name="createOntology">
      <part name="parameters" element="tns:createOntology" />
      </message>
      <message name="createOntologyResponse">
      <part name="parameters" element="tns:createOntologyResponse"
/>
      </message>
      <message name="renameOntology">
      <part name="parameters" element="tns:renameOntology" />
      </message>
      <message name="renameOntologyResponse">
      <part name="parameters" element="tns:renameOntologyResponse"
/>
      </message>
      <message name="deleteOntology">
      <part name="parameters" element="tns:deleteOntology" />
      </message>
      <message name="deleteOntologyResponse">
      <part name="parameters" element="tns:deleteOntologyResponse"
/>
      </message>
      <message name="mergeOntologies">
      <part name="parameters" element="tns:mergeOntologies" />
      </message>
      <message name="mergeOntologiesResponse">
      <part name="parameters" element="tns:mergeOntologiesResponse"
/>
      </message>
      <message name="intersectionOntologies">
      <part name="parameters" element="tns:intersectionOntologies"
/>
      </message>
      <message name="intersectionOntologiesResponse">
      <part name="parameters"
element="tns:intersectionOntologiesResponse" />
      </message>
      <message name="differenceOntologies">
      <part name="parameters" element="tns:differenceOntologies" />
      </message>
      <message name="differenceOntologiesResponse">
      <part name="parameters"
element="tns:differenceOntologiesResponse" />
      </message>
      <message name="getOntologies">
      <part name="parameters" element="tns:getOntologies" />
      </message>
      <message name="getOntologiesResponse">
      <part name="parameters" element="tns:getOntologiesResponse" />
      </message>
      <message name="listClasses">
      <part name="parameters" element="tns:listClasses" />
      </message>
      <message name="listClassesResponse">
      <part name="parameters" element="tns:listClassesResponse" />
      </message>
      <message name="addPropertyValue">
      <part name="parameters" element="tns:addPropertyValue" />
      </message>
```

```
      <message name="addPropertyValueResponse">
      <part name="parameters" element="tns:addPropertyValueResponse"
/>
      </message>
      <message name="removePropertyValue">
      <part name="parameters" element="tns:removePropertyValue" />
      </message>
      <message name="removePropertyValueResponse">
      <part name="parameters"
element="tns:removePropertyValueResponse" />
      </message>
      <message name="downloadOWL">
      <part name="parameters" element="tns:downloadOWL" />
      </message>
      <message name="downloadOWLResponse">
      <part name="parameters" element="tns:downloadOWLResponse" />
      </message>
      <message name="addClass">
      <part name="parameters" element="tns:addClass" />
      </message>
      <message name="addClassResponse">
      <part name="parameters" element="tns:addClassResponse" />
      </message>
      <portType name="OntService">
      <operation name="addProperty">
      <input
wsam:Action="http://OntService/OntService/addPropertyRequest"
message="tns:addProperty" />
      <output
wsam:Action="http://OntService/OntService/addPropertyResponse"
message="tns:addPropertyResponse" />
      </operation>
      <operation name="removeProperty">
      <input
wsam:Action="http://OntService/OntService/removePropertyRequest"
message="tns:removeProperty" />
      <output
wsam:Action="http://OntService/OntService/removePropertyResponse"
message="tns:removePropertyResponse" />
      </operation>
      <operation name="hello">
      <input wsam:Action="http://OntService/OntService/helloRequest"
message="tns:hello" />
      <output
wsam:Action="http://OntService/OntService/helloResponse"
message="tns:helloResponse" />
      </operation>
      <operation name="removeClass">
      <input
wsam:Action="http://OntService/OntService/removeClassRequest"
message="tns:removeClass" />
      <output
wsam:Action="http://OntService/OntService/removeClassResponse"
message="tns:removeClassResponse" />
      </operation>
      <operation name="addMember">
      <input
wsam:Action="http://OntService/OntService/addMemberRequest"
message="tns:addMember" />
```

```
      <output
wsam:Action="http://OntService/OntService/addMemberResponse"
message="tns:addMemberResponse" />
      </operation>
      <operation name="removeMember">
      <input
wsam:Action="http://OntService/OntService/removeMemberRequest"
message="tns:removeMember" />
      <output
wsam:Action="http://OntService/OntService/removeMemberResponse"
message="tns:removeMemberResponse" />
      </operation>
      <operation name="createOntology">
      <input
wsam:Action="http://OntService/OntService/createOntologyRequest"
message="tns:createOntology" />
      <output
wsam:Action="http://OntService/OntService/createOntologyResponse"
message="tns:createOntologyResponse" />
      </operation>
      <operation name="renameOntology">
      <input
wsam:Action="http://OntService/OntService/renameOntologyRequest"
message="tns:renameOntology" />
      <output
wsam:Action="http://OntService/OntService/renameOntologyResponse"
message="tns:renameOntologyResponse" />
      </operation>
      <operation name="deleteOntology">
      <input
wsam:Action="http://OntService/OntService/deleteOntologyRequest"
message="tns:deleteOntology" />
      <output
wsam:Action="http://OntService/OntService/deleteOntologyResponse"
message="tns:deleteOntologyResponse" />
      </operation>
      <operation name="mergeOntologies">
      <input
wsam:Action="http://OntService/OntService/mergeOntologiesRequest"
message="tns:mergeOntologies" />
      <output
wsam:Action="http://OntService/OntService/mergeOntologiesResponse"
message="tns:mergeOntologiesResponse" />
      </operation>
      <operation name="intersectionOntologies">
      <input
wsam:Action="http://OntService/OntService/intersectionOntologiesRequ
est" message="tns:intersectionOntologies" />
      <output
wsam:Action="http://OntService/OntService/intersectionOntologiesResp
onse" message="tns:intersectionOntologiesResponse" />
      </operation>
      <operation name="differenceOntologies">
      <input
wsam:Action="http://OntService/OntService/differenceOntologiesReques
t" message="tns:differenceOntologies" />
      <output
wsam:Action="http://OntService/OntService/differenceOntologiesRespon
se" message="tns:differenceOntologiesResponse" />
      </operation>
      <operation name="getOntologies">
```

```
      <input
wsam:Action="http://OntService/OntService/getOntologiesRequest"
message="tns:getOntologies" />
      <output
wsam:Action="http://OntService/OntService/getOntologiesResponse"
message="tns:getOntologiesResponse" />
      </operation>
      <operation name="listClasses">
      <input
wsam:Action="http://OntService/OntService/listClassesRequest"
message="tns:listClasses" />
      <output
wsam:Action="http://OntService/OntService/listClassesResponse"
message="tns:listClassesResponse" />
      </operation>
      <operation name="addPropertyValue">
      <input
wsam:Action="http://OntService/OntService/addPropertyValueRequest"
message="tns:addPropertyValue" />
      <output
wsam:Action="http://OntService/OntService/addPropertyValueResponse"
message="tns:addPropertyValueResponse" />
      </operation>
      <operation name="removePropertyValue">
      <input
wsam:Action="http://OntService/OntService/removePropertyValueRequest
" message="tns:removePropertyValue" />
      <output
wsam:Action="http://OntService/OntService/removePropertyValueRespons
e" message="tns:removePropertyValueResponse" />
      </operation>
      <operation name="downloadOWL">
      <input
wsam:Action="http://OntService/OntService/downloadOWLRequest"
message="tns:downloadOWL" />
      <output
wsam:Action="http://OntService/OntService/downloadOWLResponse"
message="tns:downloadOWLResponse" />
      </operation>
      <operation name="addClass">
      <input
wsam:Action="http://OntService/OntService/addClassRequest"
message="tns:addClass" />
      <output
wsam:Action="http://OntService/OntService/addClassResponse"
message="tns:addClassResponse" />
      </operation>
      </portType>
      <binding name="OntServicePortBinding" type="tns:OntService">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
style="document" />
      <operation name="addProperty">
      <soap:operation soapAction="" />
      <input>
      <soap:body use="literal" />
      </input>
      <output>
      <soap:body use="literal" />
      </output>
      </operation>
      <operation name="removeProperty">
```

```
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="hello">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="removeClass">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="addMember">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="removeMember">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="createOntology">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="renameOntology">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
```

```
</operation>
<operation name="deleteOntology">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="mergeOntologies">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="intersectionOntologies">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="differenceOntologies">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="getOntologies">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="listClasses">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
<soap:body use="literal" />
</output>
</operation>
<operation name="addPropertyValue">
<soap:operation soapAction="" />
<input>
<soap:body use="literal" />
</input>
<output>
```

```
      <soap:body use="literal" />
      </output>
      </operation>
      <operation name="removePropertyValue">
      <soap:operation soapAction="" />
      <input>
      <soap:body use="literal" />
      </input>
      <output>
      <soap:body use="literal" />
      </output>
      </operation>
      <operation name="downloadOWL">
      <soap:operation soapAction="" />
      <input>
      <soap:body use="literal" />
      </input>
      <output>
      <soap:body use="literal" />
      </output>
      </operation>
      <operation name="addClass">
      <soap:operation soapAction="" />
      <input>
      <soap:body use="literal" />
      </input>
      <output>
      <soap:body use="literal" />
      </output>
      </operation>
      </binding>
      <service name="OntServiceService">
      <port name="OntServicePort"
binding="tns:OntServicePortBinding">
      <soap:address
location="http://localhost:8090/OntService/OntService" />
      </port>
      </service>
      </definitions>
```

ÖZGEÇMİŞ

6 Mayıs 1978 tarihi, Osmaniye doğumluyum. İlkokulu Osmaniye'de, Orta okul ve liseyi İstanbul'da okudum. 1997 yılında Trakya Üniversitesi, Mühendislik-Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümüne kaydoldum. 2002 yılından bu yana özel sektörde Yazılım Mühendisliği yapmaktayım. 2006 yılında mezun olduktan sonra, 2008 yılında Beykent Üniversitesi, Bilgisayar Bilimleri Anabilim Dalında yüksek lisans eğitimine başladım.

Akademik ilgi alanlarım Yapay Zeka ve Anlamsal Ağ'dır. Yaşamımın ilerleyen yıllarında akademik alanda, bu konular üzerinde çalışmak istiyorum.

Özel ilgi alanlarım Felsefe, Müzik ve Tarih, yabancı dilim İngilizce olup, evliyim.

Aday:Eyyüphan Özdemir