EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

(DOKTORA TEZİ)

ANLAMSAL VEB UYUMLU ETMENLER İÇİN BİR ONTOLOJİ YÖNETİM SİSTEMİNİN GELİŞTİRİLMESİ

Oylum ALATLI

Tez Danışmanı: Doç. Dr. Rıza Cenk ERDUR

Bilgisayar Mühendisliği Anabilim Dalı

Sunuş Tarihi: 14.10.2015

Bornova-İZMİR
2015

Oylum ALATLI tarafından **DOKTORA TEZİ** olarak sunulan "**ANLAMSAL VEB UYUMLU ETMENLER İÇİN BİR ONTOLOJİ YÖNETİM SİSTEMİNİN GELİŞTİRİLMESİ**" başlıklı bu çalışma E.Ü. Lisansüstü Eğitim ve Öğretim Yönetmeliği ile E.Ü. Fen Bilimleri Enstitüsü Eğitim ve Öğretim Yönergesi'nin ilgili hükümleri uyarınca tarafımızdan değerlendirilerek savunmaya değer bulunmuş ve **14.10.2015** tarihinde yapılan tez savunma sınavında aday oybirliği/oyçokluğu ile başarılı bulunmuştur.

Jüri Üyeleri:	**İmza**
Jüri Başkanı: Doç. Dr. Rıza Cenk ERDUR	. . . . .
Raportör Üye: Prof. Dr. Oğuz Dikenelli	. . . . .
Üye: Doç Dr. Geylani Kardaş	. . . . .
Üye: Yrd. Doç. Dr. Selma Tekir	. . . . .
Üye: Yrd. Doç. Dr. Korhan Karabulut	. . . . .

EGE ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ

ETİK KURALLARA UYGUNLUK BEYANI

E.Ü. Lisansüstü Eğitim ve Öğretim Yönetmeliğinin ilgili hükümleri uyarınca Doktora Tezi olarak sunduğum "**ANLAMSAL VEB UYUMLU ETMENLER İÇİN BİR ONTOLOJİ YÖNETİM SİSTEMİNİN GELİŞTİRİLMESİ**" başlıklı bu tezin kendi çalışmam olduğunu, sunduğum tüm sonuç, doküman, bilgi ve belgeleri bizzat ve bu tez çalışması kapsamında elde ettiğimi, bu tez çalışmasıyla elde edilmeyen bütün bilgi ve yorumlara atıf yaptığımı ve bunları kaynaklar listesinde usulüne uygun olarak verdiğimi, tez çalışması ve yazımı sırasında patent ve telif haklarını ihlal edici bir davranışımın olmadığını, bu tezin herhangi bir bölümünü bu üniversite veya diğer bir üniversitede başka bir tez çalışması içinde sunmadığımı, bu tezin planlanmasından yazımına kadar bütün safhalarda bilimsel etik kurallarına uygun olarak davrandığımı ve aksinin ortaya çıkması durumunda her türlü yasal sonucu kabul edeceğimi beyan ederim.

14.10.2015

. . . . . .

Oylum ALATLI

ÖZET

ANLAMSAL VEB UYUMLU ETMENLER İÇİN BİR ONTOLOJİ YÖNETİM SİSTEMİNİN GELİŞTİRİLMESİ

ALATLI, Oylum

Doktora Tezi, Bilgisayar Mühendisliği Anabilim Dalı
Tez Danışmanı: Doç. Dr. Rıza Cenk ERDUR
14.10.2015, 93 sayfa

Anlamsal Veb ve pratikteki uygulaması durumundaki Bağlı Veri Ağı sürekli genişlemekte, içerdikleri ontolojiler ile veri kümeleri kontrolsüz olarak değişmektedir. Dolayısıyla, Anlamsal Veb ve Bağlı Veri Ağı üzerinde çalışan etmenlerin kendilerini ilgilendiren verilerdeki ve ontolojilerdeki değişimleri izleyebilmeleri, değişimlerin gerektirdiği tepkileri zamanında verebilmeleri önem kazanmıştır. Literatürde ontoloji değişimlerini yöneten çalışmalar bulunmasına karşın, bu çalışmalarda gerçekleşen değişimleri yazılımlardan saklamak amaçlandığından etmen sistemlerinin tepkisel olmalarından kaynaklanan değişimlerden haberdar olma gereksinimlerini karşılayamamaktadır. Bağlı Veri kümelerindeki değişimleri izleyen çalışmalar da mevcuttur. Bu çalışmalar ise veri seti düzeyindeki değişimleri izlemeleri, federe sorguları desteklememeleri ya da itme-isteme tabanlı yaklaşımları eş zamanlı olarak desteklememeleri nedeniyle etmen sistemlerinin gereksinimlerini karşılayamamaktadırlar. Bu bağlamda tez kapsamında etmenlerin anlamsal veb üzerindeki veri ve ontolojilerdeki değişimleri izlemek için kullanabilecekleri itme-isteme tabanlı federe sorguları destekleyen bir altyapı geliştirilmiştir. Bu altyapı, "ortam" soyutlaması ile buna ilişkin A&A meta-modeline dayanmakta ve değişimlerin izlenmesi için gerekli artifaktları etmenin ortamı içerisinde çalışma alanlarında barındırmaktadır. Bu tez bilindiği kadarıyla etmen sistemlerinde ortamı Anlamsal Veb ve Bağlı Veri açısından ele alan ilk çalışmadır.

Anahtar Sözcükler: Ontoloji Değişimi, Anlamsal Veb, Bağlı Veri, Bağlı Veri Dinamikleri, Çok Etmenli Sistemler, Ortam Programlama, CArtAgO.

ABSTRACT

DEVELOPMENT OF AN ONTOLOGY MANAGEMENT SYSTEM FOR SEMANTIC WEB ENABLED AGENTS

ALATLI, Oylum

PhD. in Computer Engineering
Supervisor: Doç. Dr. Rıza Cenk ERDUR
14.10.2015, pages

Semantic Web and its practical application Linked Data Web are getting bigger gradually while the ontologies and datasets included change without any control. Therefore, it became important for the agents that work on the Semantic Web and Linked Data Web to monitor the changes in the ontologies and data they use, and react to these changes promptly. Although there are studies for managing ontology changes in the literature, since these studies aim to hide the changes from software, they do not meet the change awareness requirement of agent systems. There are studies for managing the changes in Linked Data datasets too. Since they monitor only dataset level changes, and do not support federated queries or push-pull based approaches, these studies can not meet the requirements of agent systems either. In this context, a pull-push based framework with support for federated queries that can be used by agents for monitoring data and ontology changes on the Linked Data Web and Semantic Web is developed within the scope of this thesis. This framework is built upon the environment abstraction and the related A&A metamodel, and hosts the artifacts necessary for monitoring changes in the agent environment workspaces. To our knowledge, this study is the first one that approaches agent environment from the Semantic Web and Linked Data perpectives.

Keywords: Ontology Change, Semantic Web, Linked Data, Linked Data Dynamics, Multi Agent Systems, Environment Programming, CArtAgO.

TEŞEKKÜR

Doktora sürecinde deneyimleri, bilgileri ve yönlendirmeleriyle beni destekleyen danışmanım Doç Dr. Rıza Cenk Erdur'a ve Prof. Dr. Oğuz Dikenelli'ye çok teşekkür ederim. Değerli katkıları için jüri üyeleri Doç. Dr. Geylani Kardaş'a, Yrd.Doç.Dr. Korhan Karabulut'a ve Yrd.Doç.Dr. Selma Tekir'e teşekkür ederim.

Tez süresince teknik bilgi ve deneyimleri ile bana destek olan Burak Yönyül'e, Tayfun Gökmen Halaç'a, Ziya Akar'a ve Erdem Eser Ekinci'ye teşekkür ederim.

Doktora çalışmalarım boyunca verdikleri manevi destek için başta Birol Çiloğlugil, Özlem Özgöbek ve Ali Murat Tiryaki olmak üzere tüm arkadaşlarıma teşekkür ederim.

Tüm hayatım boyunca beni her konuda destekleyen aileme teşekkür ederim.

Son olarak, 111E027 nolu proje kapsamında verdiği destek için Türkiye Bilimsel ve Teknik Araştırma Kurumu'na teşekkür ederim.

İÇİNDEKİLER

Sayfa

İÇİNDEKİLER (Devam)

ŞEKİLLER DİZİNİ

ŞEKİLLER DİZİNİ (Devam)

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltma	**Açılım**
A&A	Agents and Artifacts
API	Application Programming Interface
CartAgO	Common Artifact Infrastructure for Agents Open Environments
DaDy	Dataset Dynamics Vocabulary
FIPA	Foundation for Intelligent Phsical Agents
OMV	Ontology Metadata Vocabulary
OWL	Web Ontology Language
OWL-ODM	OWL Object Definition Metamodel
RDF	Resource Description Framework
RDFS	RDF Schema
REST	Representational State Transfer
SPARQL	Simple Protocol and RDF Query Language
UGA	Uygulama Geliştirme Arayüzü
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
VOID	Vocabulary of Interlinked Datasets
WWW	World Wide Web

1 GİRİŞ

Anlamsal Veb makinaların veb üzerindeki bilgiyi otonom şekilde tüketebilmesini ve bunun sonucunda özerk çalışmasını amaçlayan Dünya çapında bir bilgi ağıdır. Bu ağdaki temel yapılar bilginin tanımlandığı ontolojiler, bu ontolojiler kullanılarak tanımlanmış olan Anlamsal Veb servisleri ve ontolojilerdeki bilgi ile Anlamsal Veb servislerinin kullanıcısı durumundaki etmenlerdir.

Anlamsal Veb araştırılmaya başlandığı ilk dönemlerden itibaren birbirinden ayrık ontolojiler şeklinde gelişim göstermiş, dolayısıyla hedeflenen Dünya çapındaki bilgi ağına ulaşılamamıştır. Bunun üzerine ortaya atılan Bağlı Veri prensipleri (Lee, 2006) ile amaçlanan Dünya çapındaki Anlamsal Veb'e ulaşılabilmiştir. Anlamsal Veb fikrinin pratikteki uygulaması haline gelen Bağlı Veri prensipleri ile oluşturulan Veri Ağı her geçen gün büyümektedir. Gittikçe artan sayıda kuruluş ve devlet mevcut ontolojileri ve diğer Bağlı Veri setlerini kullanan veri setleri yayınlamaktadır(Schmachtenberg et al., 2014). Ontolojiler ve Bağlı Veri setleri Anlamsal Veb servislerinin tanımlanmasında da kullanılmakta, etmenlerin çıkarsama yapabilmelerini, birlikte çalışabilmelerini, iletişim kurabilmelerini, heterojen bilgi kaynaklarını ve Anlamsal Veb servislerini kullanabilmelerini olanaklı kılmaktadır. Dolayısıyla Bağlı Veri setlerinin ve ontolojilerin etmen sistemleri açısından önemi her geçen gün artmaktadır.

Etmenlerin ve Anlamsal Veb servislerinin işleyişlerinin ontolojilerin ve Bağlı Veri setlerinin sağladığı anlamsal birlikte işlerliğe dayalı olması ontolojileri ve Bağlı Veri setlerini etmenler açısından Anlamsal Veb'in en kırılgan noktası haline getirmektedir. Zira ontolojlerin ve Bağlı Veri setlerinin yayınlanmaları, silinmeleri ve değiştirilmeleri üzerinde herhangi bir kısıt bulunmadığından etmen sistemleri bunların üzerinde yapılan değişikliklerden etkilenmektedirler.

Etmenlerin, temel özelliklerinden olan karşıt eylemliliğin gereği olarak, bu değişimleri algılamaları ve tepki vermeleri gerekmektedir. Bu tepkiselliğin sağlanması da Anlamsal Veb ve Bağlı Veri Ağı üzerinde etmenleri ilgilendiren çok sayıdaki ontolojinin ve veri setinin değişim olasılığına karşı sürekli izlenmesini gerektirmektedir.

Anlamsal Veb ve Veri Ağı'nı izleme işlevine sahip olan bir etmen sisteminin geliştirilmesi etmen geliştiricisinin Anlamsal Veb ile Bağlı Veri teknolojileri ve araçları hakkında ayrı bir bilgi birikimine sahip olmasını ve ek servislerin ya da bileşenlerin gerçekleştirilmesini gerektirdiğinden bu hem etmen

geliştiricilerine hem de etmen uygulamasına ek yük getirecektir. Dolayısıyla etmen sistemlerinin ve geliştiricilerinin Bağlı Veri Ağı ile Anlamsal Veb'in karmaşıklığı ve dinamikliğinden soyutlanmasını sağlayacak bir altyapı gerekmektedir.

Etmen sistemleri literatüründe bu tür akıllı davranış ve otonomluk gerektirmeyen veri erişimine dayalı işlemler etmen ortamının görevleri arasında sayılmaktadır (Weyns et al., 2007). Bu tez kapsamında gerçekleştirilen çalışmadaki amaç da Bağlı Veri Ağı ile Anlamsal Veb'in izlenmesini ve oluşan değişimlerin ilgili etmenlere iletilmesi, dolayısıyla da etmen sistemlerinin ve geliştiricilerin bu detaylardan soyutlanması olduğundan bu servisleri sağlayacak etmen ortamlarının geliştirilmesi amaçlanan çalışma için en uygun yöntem olarak öne çıkmıştır.

Anlamsal Veb ve Bağlı Veri Ağı'ndaki değişimlerin izlenmesi amacıyla geliştirilen sistemlerde iki temel yaklaşım bulunmaktadır: itme ve isteme. Bir diğer yaklaşım ise itme ve isteme tabanlı sistemlerin olumlu yönlerini bir araya getiren isteme ve itme tabanlı sistemlerdir.

Bu sistemleri ayrıştıran bir başka nokta ise takip ettikleri verinin ayrıntı düzeyidir. Temelde dört ayrıntı düzeyi bulunmaktadır: veriseti, kaynak, deyim ve sorgu. Anlamsal Veb'i ve Bağlı Veri Ağı'nı kullanan uygulamalar genellikle sorgulamalarla elde ettikleri verileri tükettikleri için ağ üzerindeki verinin uygulamalar tarafından kullanılan sorgular cinsinden izlenmesi daha etkin bir yöntemdir. Literatürde sorgu düzeyinde veri izleyen uygulamalar (Passant and Mendes, 2010; Popitsch and Haslhofer, 2011) tek bir veri kaynağı üzerinde çalışabilen sorguları izlemektedirler. Oysa ki Bağlı Veri üzerinde birden çok kaynaktan gelen verinin birleştirilmesini gerektiren federe sorgular çalıştırılabilmektedir. Dolayısıyla verinin federe sorgu düzeyinde de izlenmesi gerekmektedir.

Tez çalışmasında yukarıda yapılmış olan problem tanımlaması ışığında geliştirilecek olan altyapı ile literatüre şu katkıların yapılması amaçlanmıştır:

- geliştirilecek altyapının sadece kendisini kullanan etmen sisteminin kullandığı Anlamsal Veb ve Bağlı Veri Ağı kaynaklarına odaklanması,
- odaklanılan veri kaynakları üzerinde etmenlerin gereksinim duydukları basit ve federe sorguların çalıştırılabilmesi,
- etmenlerin sonuçlarını kullandıkları sorguların değişim ihtimaline karşı periyodik olarak izlenmesi ve etmenlerin değişimlerden isteme ve itme tabanlı bir yaklaşımla haberdar edilmesi,

- sorgulama ve değişimlerin izlenmesi işlevlerinin etmen geliştiricileri ile etmen sistemlerinden soyutlanması, böylelikle yazılım geliştirme sürecinin ve etmen yazılımlarının karmaşıklığının azaltılması.

Tezin bundan sonraki bölümleri şu şekildedir:

- 2. bölümde Anlamsal Veb ve Bağlı Veri teknolojileri, sonrasında ise ontolojilerinin ve Bağlı Veri kümelerinin dinamikleri ile bu dinamiklerin yönetimi incelenecektir.
- 3. bölümde etmen ortamları, içkökenli ortamlar, içkökenli ortam modelleri ve bu modellerin gerçekleştirimi açıklanacaktır.
- 4. bölümde çok etmenli sistemlerde ontoloji ve bağlı veri değişimlerinin yönetilmesi amacıyla tez kapsamında geliştirilen sistemin soyut ve gerçekleştirilen mimarileri tartışılmış, sonrasında ise geliştirilen sistemin performans ölçümüne yer verilmiştir.
- 5. bölümde ise çalışmanın katkıları ve planlanan çalışmalar tartışılmıştır.

2 ANLAMSAL VEB VE BAĞLI VERİ TEKNOLOJİLERİ

2.1 Anlamsal Veb ve Bağlı Veri Teknolojileri

Anlamsal Veb, WWW üzerindeki bilginin makineler tarafından da anlaşılır hale getirilmesini hedefleyen dünya çapında bir bilgi ağı olarak tanımlanabilir (Berners-Lee et al., 2001). Ağ üzerindeki bilginin makinalar tarafından anlaşılması yazılımların eriştikleri bilgiyi insan müdahalesi olmadan kullanarak bu bilgi üzerinde çıkarsama yapabilmesini, diğer yazılımlarla ve kullanıcıları ile işbirliği yapabilmesini olanaklı kılması açısından önemlidir. Anlamsal Veb üzerinde çalışan iki temel yazılım türü bulunmaktadır: akıllı etmenler ve Anlamsal Veb servisleri (Berners-Lee et al., 2001).

Bu iki temel yazılım türü işlevlerini yerine getirebilmek için bilginin Anlamsal Veb üzerinde ifade edildiği üçüncü temel bileşen olan ontolojilere ihtiyaç duymaktadırlar. Anlamsal Veb'i şekillendiren temel bileşen ontolojilerdir. Ontolojiler Veb üzerindeki bilginin anlamının makineler tarafından anlaşılmasını ve işlenmesini olanaklı kılan ortak terminolojiler oluştururlar. Bu terminolojiler sayesinde etmenler çıkarsama yapabilmekte, birlikte çalışabilmekte, iletişim kurabilmekte, heterojen bilgi kaynaklarını ve anlamsal veb servislerini kullanabilmektedirler.

Ontolojilerin Anlamsal Veb'deki önemi, onları aynı zamanda Anlamsal Veb'in en kırılgan noktası haline getirmektedir. Anlamsal Veb'deki tüm işleyiş ontolojilerin sağladığı anlamsal birlikte işleyebilirliğe ("interoperability") dayalı olduğu için ontolojilerdeki değişimler tüm işleyişi bozabilmektedir. Ontolojilerin yayınlanması ve değiştirilmesi üzerinde hiçbir kısıt bulunmadığından her an bir ontoloji değişimi gerçekleşebilmekte ya da bir ontoloji habersizce silinebilmekte ve Anlamsal Veb bileşenlerinin bu değişimlere uyum sağlamaları gerekmektedir. Anlamsal Veb'in etmenler tarafından kullanılan diğer bir bileşeni olan anlamsal veb servislerinin tanımlanmaları, bulunmaları, çalıştırılmaları ve birlikte çalışmaları da ontolojilerin sağladığı anlamsal bilgi ile olasıdır. Dolayısıyla ontolojilerin değişimi etmenlerin anlamsal veb servisleri hakkındaki bilgilerini de geçersiz kılmakta, uyum sağlamaları gereken bir başka değişiklik oluşturmaktadır. Dahası, etmenlerin kullanmakta oldukları anlamsal veb servislerinin yapılarının (girdi çıktı tipleri, vs.) değişimi de etmenlerin uyum sağlamaları gereken bir değişim türüdür. Bu temel işlevlerinden ötürü ontolojiler Anlamsal Veb'in belkemiği olarak nitelendirilebilirler.

Ontolojilerin içerdiği bilgiyi kullanan, dolayısıyla algıları ile çalışma ve diğer etmenlerle iletişimlerini ontolojiler üzerine kuran etmenler için bu büyük bir sorun oluşturmaktadır. Bu nedenle ontoloji değişimlerinin yönetiminin etmen sistemleri açısından ele alınması gerekmektedir. Bu tez kapsamında yapılan çalışmada ontoloji değişimlerinin etmen sistemlerinde yönetilmesini de hedefleyen bir mimari geliştirilmiştir. Tezin bu bölümünde ontoloji ve ontoloji değişimi kavramları ile ontoloji değişikliklerinin izlenmesinde kullanılan teknikler incelenecektir.

2.1.1 Ontolojiler

Ontoloji kavramı değişik araştırmacılar tarafından farklı şekillerde tanımlanmıştır.

Neches ve arkadaşlarına (Neches et al., 1991) göre bir ontoloji bir konu alanının sözcük dağarcığını oluşturan temel terimler ve kavramlar ile terimleri birleştirmek için gerekli kuralları ve sözcük dağarcığını genişletmek amacıyla tanımlanacak ilişkileri içerir.

Gruber'e (Gruber, 1993) göre bir ontoloji bir kavramsallaştırmanın açık tanımlamasıdır.

Borst'a (Borst, 1997) göre ontolojiler paylaşılan bir kavramsallaştırmanın ortak tanımlamasıdır.

Tüm bu tanımlamalarda üzerinde anlaşılan nokta ontolojilerin özel bir konu alanındaki kavramları ve bunların arasındaki ilişkileri tanımlamak için kullanılabileceğidir (Ma and Wang, 2009).

Huhns ve Singh (Huhns and Singh, 1997) ise ontoloji kavramını değişik bir bakış açısından tanımlamıştır. Bu yaklaşımda bir ontoloji dünyanın bir kısmını temsil eden bilişimsel ("computational") bir model olarak tanımlanmaktadır. Bu model her etmenin inanç ve eylemlerini temellendirebileceği paylaşılan bir sanal dünya olarak da görülmektedir. Diğer tanımlardan farklı olarak Huhns ve Singh'in tanımı ontolojilerin etmen ortamlarının bir parçası olduğunu vurgulamaktadır. Bu nedenle ontolojilerin makineler tarafından işlenebilecek şekilde ifade edilmesi önemlidir.

Ontolojiler belirli bir alandaki kavramlar için ortak bir anlayış ve tanım oluşturmaları, ilgili oldukları alandaki varsayımları açıkça ifade etmeleri, alan

bilgisinin tekrar kullanılmasını olanaklı kılmaları ve belli bir alanla ilgili bilgiyi makinelerin anlamalarına ve otomatik olarak işleyebilmelerine olanak sağlamaları nedeniyle yazılım geliştirme açısından önem kazanmışlardır (Yu, 2011).

Ontolojiler genellikle anlamsal ağlarla ("semantic network") ifade edilir. Bu ağlarda düğümler kavramlar ya da kavramların örneklerini, kenarlar ise bu kavram ve örnekler arasındaki genelleme ve kalıtım, örnekleme ve kümelenme ("aggregation") ilişkilerini ifade etmektedir. Ontolojilerin diğer temel bileşenleri ise fonksiyonlar, kurallar ve formal aksiyomlardır (Gruber, 1993). Ontolojilerin Anlamsal Veb üzerinde ifade edilmesi için pratikte kullanılan W3C tarafından standartlaştırılmış teknolojiler ise gelişim sıralarına göre RDF, RDFS ve OWL'dur. Tüm bu teknolojilerin işleyişi Anlamsal Veb terminolojisinde kaynak ("resource") olarak isimlendirilen Dünya üzerindeki varlık ya da kavramların birer URI ("Unified Resource Identifier") ile temsil edilmesine dayanmaktadır.

RDF ("Resource Description Framework") Anlamsal Veb üzerindeki URI'leri kullanarak bilgi ifade edilmesini sağlayan bir W3C standardıdır. Anlamsal Veb üzerindeki tüm bilgi RDF formatında ifade edilmektedir. Bu teknoloji basitçe bilginin ifade ("statement") adı verilen özne-yüklem-nesne ("subject-predicate-object") üçlüleri şeklindeki önermelerle ifade edilmesi esasına dayanır. Özne ve nesneler birer URI ile temsil edilen varlıklara ya da kavramlara, yüklemler ise yine birer URI ile temsil edilen ve özneler ile nesneler arasında kurulan ilişkilere karşılık gelmektedir. Kimi üçlülerde nesneler sade ("literal") değerler de alabilmektedirler. Her üçlü tek bir gerçeği ("fact") ifade etmektedir. Birçok RDF ifadesinin bir araya gelmesi ile ise RDF çizgesi ("graph") oluşmaktadır, zira bir URI yüklemler aracılığı ile bir çok farklı URI ile ilişkilendirilebilir.

RDFS ("RDF Schema") ise RDF gösterimi kullanılarak sınıflar, alt sınıflar ve RDF kaynaklarının özelliklerini içeren sıradüzensel ("hierarchical") söz varlıklarının ("vocabulary") tanımlanmasını olanaklı kılan dilsel yapıları içeren bir W3C standardıdır. Sözü geçen dilsel yapıların kendileri de başka sınıf, alt sınıf, özellik ve alt özelliklerin tanımlanmasını olanaklı kılacak temel sınıflar ve özelliklerdir. RDFS kullanılarak sınıfların örnekleri de oluşturulabilir. Ayrıca, özelliklerin tanım ve değer kümeleri tanımlanabilir. Tanım ve değer kümeleri belli bir sınıftan olacak şekilde tanımlanabilmektedir. Değer kümelerini özel tanımlanmış XSD ("XML Schema Definition") veri tipleri ya da RDFS'in içerdiği "Literal" sınıfı cinsinden de tanımlamak mümkündür. RDFS'in bir başka önemli özelliği ise tanımlanan söz varlıklarının içerdiği sınıfların arasındaki kapsama ilişkilerinin basit düzeyde otonom çıkarsamayı da olanaklı kılmasıdır. RDF'i ifade edicilik açısından oldukça genişletmesine karşın RDFS'te hariç tutma

(“exclusion”), olumsuzlama (“negation”), aynılık ilişkisi, kardinalite vb. ifade edilememektedir (Yu, 2011).

RDFS üzerine inşa edilen diğer W3C standardı ise OWL’dur (“Web Ontology Language”). OWL Tanımlama Mantığı (“Description Logic”) temelli çıkarsamayı destekleyen bir ontoloji tanımlama dilidir. Sözdizimsel olarak RDF’i kullanmasına ve her OWL dokümanı sözdizimsel olarak geçerli bir RDF dokümanı olmasına karşın karşın RDF ile ifade edilen her bilgi OWL için anlambilimsel olarak geçerli bir ifade oluşturmaz (Obitko). OWL dili ile tanımlanan söz varlıkları ilgili oldukları alandaki RDF dokümanlarının yaratılmasında kullanılabilmektedir. OWL RDFS’ten gelen içerme (“subsumption”) ve üyelik (“membership”) ilişkilerinin yanısıra mantıksal ifadeler kullanılarak karmaşık sınıflar yaratılmasını ve sınıflar üzerinde kısıtlamaların yaratılmasını olanaklı kılar. Dolayısıyla OWL’da RDFS’in aksine tip belirleme, alt sınıf ve alt özellik belirleme ile özelliklerin tanım ve değer kümelerini belirleme haricinde kalan olumsuzlama ve hariç tutma gibi aksiyomlar da tanımlanabilir (Yu, 2011). W3C günümüze kadar OWL 1 (McGuinness et al., 2004), OWL 1.1 (Peter F. Patel-Schneider and Grau, 2006) ve OWL 2 (Group, 2012) olmak üzere üç OWL standardı yayınlamıştır.

OWL 2’nin OWL 1 ve OWL 1.1’e göre farklılıkları şunlardır (Yu, 2011):

- Sık kullanılan ifadelerin daha kolay oluşturulabilmesi için çıkarsama yöntemini değiştirmeyecek söz dizimsel ekler getirilmiştir.
- Yansımalı (“reflexive”) ve asimetrik özellikler, şarta bağlı kardinalite, özellik zincirleri ve anahtarları gibi ifade gücünü (“expressiveness”) arttıracak yeni yapılar.
- Veri tipleri için genişletilmiş destek ve kullanıcı tanımlı veri tipleri.
- Basit üst modelleme yetenekleri ile genişletilmiş dipnot (“annotation”) yetenekleri.

2.1.2 Ontoloji-Etmen ilişkisi

Anlamsal Veb’deki aktif bileşen olan etmenler kullanıcıları adına otonom işlem yapabilmek için ilgili oldukları alanla ilgili kavramlara ve o anda kendilerinin ve içinde bulundukları ortamın şartlarını tanımlamayan bir bilgi tabanına (“knowledgebase”) ihtiyaç duymaktadırlar. Daha önce de ifade edildiği üzere belli bir alandaki kavramlar ontolojilerle ifade edilebilmektedir. Etmenlere

ait bilgi tabanlarındaki durum tanımlamalarında da yine alan ontolojilerine ait kavramlardan yararlanılmaktadır. Dolayısıyla ontolojiler etmenler için oldukça temel bir gereksinimdir. Bu bölümde etmen sistemlerinde ontoloji kullanımının gerekli olmasının nedenleri daha detaylı olarak incelenecektir.

Bilginin Yeniden Kullanımı ve Paylaşımı

Ontolojiler belli bir alanda paylaşılan bir söz varlığı oluşturur (Ma and Wang, 2009). Ayrıca ontolojilerin içinde ifade edilen bilginin içerdiği terimler ve kavramlar başka kullanıcılar ve etmenler tarafından paylaşılıp tekrar kullanılabilir (Sanz and López, 2006).

İletişim

FIPA tarafından tanımlanan etmen iletişim modeli (ACL) iletişim kuran etmenlerin sözeylemlerin ("speech act") ve protokollerin tanımlandığı bir iletişim ontolojisini paylaştıkları varsayımına dayanmaktadır. İşbirliğine temel oluşturan iletişimin başarılı olabilmesi için etmenler bu iletişim ontolojisine ek olarak çalıştıkları alanla ilgili açık olarak tanımlanmış bir mesaj içeriği ontolojisini ("message content ontology") de paylaşmak zorundadırlar (Ma and Wang, 2009; Dolia, 2010). Paylaşılan ontolojiler ilgili oldukları alandaki kavramların, ilişkilerin ve kısıtların etmenler tarafından kullanılabilmesini olanaklı kılar, böylelikle etmenler arasındaki iletişim, pazarlık ve etkileşim sırasında kullanılan bilgiye açıklık kazandırarak (Sanz and López, 2006) ontolojilerin içerdikleri bilginin üzerinde anlaşılmış ortak bir anlayışla işlenmesini sağlarlar.

Birlikte İşleyebilirlik ("Interoperability") ve İşbirliği

Ontolojiler etmenlerin birlikte çalışabilmesi için gerekli temeli oluştururlar. Daha önce de belirtildiği üzere, paylaşılan ontolojiler ilgili oldukları alandaki kavramların, ilişkilerin ve kısıtların etmenler tarafından kullanılabilmesini ve paylaşılabilmesini olanaklı kılar, bu sayede etmenler arasındaki iletişim, pazarlık ve etkileşim sırasında kullanılan bilgiye açıklık kazandır. Böylelikle etmenlerin birlikte çalışabilmeleri olanaklı hale gelmiş olur. Ayrıca etmenlerin özellik ve yeteneklerinin ontolojik tanımlamaları diğer etmenlerin etkileştikleri etmenleri tanımalarını ve işbirliği yapacakları etmenleri seçmelerini olanaklı kılar (Ma and Wang, 2009).

Çıkarsama (“Reasoning”)

Etmenler kendi için durumlarını tanımlamak ve eylemleri ile dış dünya hakkında çıkarsamada bulunmak için ontolojilerden yararlanırlar. Etmenler ontolojilerdeki ilişkileri ve bağlantıları kullanarak çalıştıkları alandaki öngörülmemiş ve yeni olaylar hakkında çıkarsama yapabilirler. Böylelikle etmen sistemlerinin kırılganlığı azalır, çalıştıkları alandaki değişimlere uymaları kolaylaşır (Sanz and López, 2006).

Modelleme ve Ortam Tanımlaması

Ontolojiler etmenlerin ihtiyaç duydukları kavramların haricinde iç operasyonlarının ve görevlerinin tanımlanmasında da kullanılmaktadır. Örneğin, FIPA Talep Etkileşim Protokolü’nde (“FIPA Request Interaction Protocol”) mesaj tipleri, talep nedeni ve ön koşullar ontolojik olararak tanımlanan kavramlardır. Ortam tanımlamasında ortamdaki etmenlerin yetenek tanımlamaları (Ma and Wang, 2009) da yer alabilir. Anlamsal Veb servislerinin etmenler tarafından bulunması, çalıştırılması, birleştirilmesi (“composition”) ve birlikte çalıştırılması da ontolojiler yardımıyla gerçekleştirilebilmektedir.

Bilgi Erişimi

Etmenlerin bilgiye otonom şekilde ulaşmaları bilgi kaynaklarındaki bilginin ontolojilerle açıklanması (“annotate”) ile mümkün olabilir (Kolomvatsos and Hadjiefthymiades, 2008).

2.1.3 Anlamsal Veb dinamikleri ve ontoloji değişimleri

Anlamsal Veb üzerinde herhangi bir kontrol mekanizması bulunmadığından ontolojiler herhangi bir uyarı gereksinimi olmadan değişik sıklıklarda değişebilmekte, silinebilmekte ya da yeni ontolojiler yayınlanabilmektedir. Ontolojilerin değişimlerinin etkileri, değişimlerin nasıl doğrulanabileceği, ontoloji versiyonlarının nasıl takip edilebileceği ve ontoloji kullanıcılarına nasıl en güncel bilginin ulaştırılabileceği Anlamsal Veb Dinamikleri (“dynamics”) (Antoniou et al., 2011) adı altında incelenmektedir.

Ontolojiler ilgili oldukları alandaki (“domain”) değişiklikler, ortak kavramlardaki değişiklikler (örneğin, bir ontoloji yeni bir iş ya da alan için

uyarlandığında), ontolojilerin ait oldukları alandaki kavramların yorumlanışındaki ve ifade edilmesindeki değişiklikler, ontolojinin ifade edilişinde hatalar bulunması ile ilgili alana yeni kavramların eklenmesi ya da çıkarılması nedeni ile değişebilirler (Klein and Fensel, 2001; Flouris and Plexousakis, 2005). Anlamsal Veb üzerindeki ontolojiler için bir başka değişim nedeni ise bu ontolojilerin kullandıkları diğer ontolojilerde oluşan değişimlerden etkilenmeleridir (Flouris and Plexousakis, 2005).

Ontolojilerde oluşabilecek değişiklikler üzerine değişik ayrıntı düzeyinde ve özellikte olan değişimleri inceleyen çok çeşitli çalışmalar olmasına karşın temelde bu çalışmalar Klein ve arkadaşları (Klein and Fensel, 2001; Klein, 2004; Klein et al., 2002; Klein et al., 2003) ile Stojanovic ve arkadaşları (Stojanovic, 2004; Stojanovic et al., 2002) tarafından yapılan çalışmalara dayanmaktadır.

Klein ve arkadaşlarına göre (Klein et al., 2002) ontolojilerde oluşabilecek değişiklik tipleri şunlardır:

- Mantıksal olmayan değişiklikler: Bir kavram ya da özelliğin rdfs:label özelliğindeki değişiklik gibi doğal dilde yapılan değişiklikler. Bu değişikliklerin ontolojinin içeriğinin anlamı üzerine etkisi yoktur.

- Mantıksal değişiklikler: Bir kavram ya da özelliğin tanımında anlamını etkileyecek şekilde yapılan değişikliklerdir. Bu değişimlere örnek olarak subClassOf (“alt sınıf”), “domain” (tanım kümesi) ya da “range” (değer kümesi) ifadelerinin, bir sınıfın özellik kısıtlarının silinmesi vb. verilebilir.

- Tanımlayıcı değişiklikleri: Bir kavram ya da özelliğe yeni bir tanımlayıcı verildiğinde, örneğin yeniden isimlendirme yapıldığında oluşan değişikliklerdir.

- Yeni tanımlamaların eklenmesi

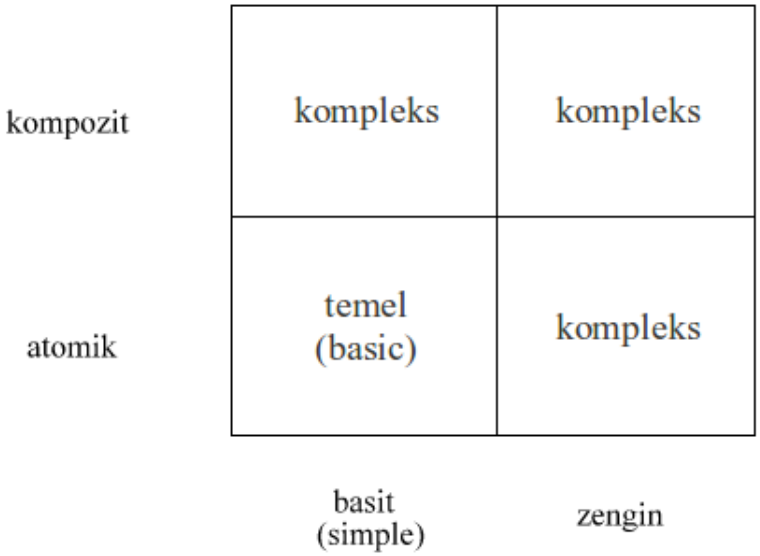
- Mevcut tanımlamaların silinmesi: Stojanovic (Stojanovic, 2004) ise ontoloji değişiklikleri ekleyici ve çıkarıcı değişiklikler olmak üzere kabaca ikiye ayırmaktadır. Ekleyici değişiklikler sırasında ontolojide mevcut olan kavram, özellik ve örneklere müdahalede bulunulmadan sadece eklemeler yapılır. Çıkarıcı değişikler sırasında ise ontolojideki mevcut bileşenler çıkarılır. Ekleyici ve çıkarıcı değişikler minimal ve tam bir değişiklik

kümesi oluştururlar, yani bir ontoloji üzerinde gerçekleştirilebilecek tüm değişiklikler ekleyici ve çıkarıcı değişikliklerin bir arada kullanılması ile oluşturulabilir. Örneğin bir kavramın yeniden isimlendirilmesi aslında kavramın tüm örneklerinin ve kavramın silinmesi, yeni isimle bir kavramın ve bu kavrama bağlı olarak silinen tüm örneklerin yaratılması ile eşdeğerdir.

Stajonovic (Stojanovic, 2004) çalışmasında ontolojilerdeki değişiklik operasyonlarını ekleyici ve çıkarıcı değişiklik operasyonlarını temel alarak basit, kompozit ve kompleks olmak üzere üçe ayırmaktadır:

- Basit değişiklikler ontoloji modelindeki tek bir kavramı ekleyen ya da çıkaran değişikliklerdir. Örneğin öğrenci sınıfının altına lisansüstü öğrenci sınıfının eklenmesi basit bir değişikliktir.
- Kompozit değişiklikler bir ontolojideki bir kavramın ya da özelliğin çevresini değiştiren (bileşen yaratılması, çıkarılması ya da değiştirilmesi) değişikliklerdir. Bir kavramın çevresi altkavramlar, üstkavramlar, kavramın tanım ya da değer kümesinde olduğu özellikler ve kavramın örneklerinden oluşur. Bir özelliğin çevresi ise tanım kümesi kavramları, değer kümesi kavramları, altözellikler, üstözellikler ve özelliğin örneklerini içerir. Kompozit değişikliklere örnek olarak bir kavramın bölünmesi, birden fazla kavramın birleşmesi, kavramların gruplanması, kavramların özelleşmesi ve genelleşmesi, taksonomide bir kavram ve alt kavramı arasına yeni bir kavramın yerleştirilmesi, kavramların taksonomide aşağıya ya da yukarıya çekilmesi verilebilir. Kompozit değişiklikler basit değişiklikler cinsinden ifade edilebilir.
- Kompleks değişiklikler ise en azından iki adet kompozit ya da basit değişikliğe bölünebilen ontoloji değişiklikleridir. Kompleks değişiklikler basit değişiklikler cinsinden ifade edilebilir.

Klein da Stajonovic'e benzer bir değişiklik operasyonu sınıflandırması yapmıştır. Bu sınıflandırmaya (Klein, 2004) göre ontolojilerde gerçekleştirilen değişiklik operasyonları (Y. et al., 2005) iki boyutta sınıflandırılabilir. İlk boyutta değişiklik operasyonları basit ve zengin değişiklikler olmak üzere ikiye ayrılırlar. Basit değişiklikler ontolojinin sadece yapısının incelenmesi ile bulunabilir. Buna karşın zengin değişiklikler gerçekleştirilen operasyonun ontolojinin mantıksal yapısı üzerindeki etkisiyle ilgili bilgi içerirler, dolayısıyla bulunmaları için ontolojinin mantıksal teorisinin sorgulanması gerekir. İkinci boyutta ise değişiklikler atomik ve kompozit değişiklikler olmak üzere ikiye ayrılırlar. Atomik operasyonlar



Şekil 2.1: Ontolojiler üzerinde gerçekleşen değişiklik operasyonlarının sınıflandırılması. (Klein, 2004'ten uyarlanmıştır.)

daha küçük operasyonlara bölünemeyen değişiklik operasyonlarıdır. Kompozit operasyonlar ise mantıksal bir yapı oluşturan operasyonların gruplanması ile oluşturulurlar. Ayrıca, bazı sezgisel yöntemlerin ve kuralların uygulanması ile basit değişikliklerden kompozit değişikliklerin üretilmesi de mümkündür.

Klein ve Stojanovic'in terminolojileri farklı görünmesine karşın aslında Stojanovic'in (Stojanovic, 2004) basit ve kompleks değişiklikleri, Klein'ın (Klein, 2004) atomik ve kompozit değişikliklerine karşılık gelmektedir (Haase, 2004).

Literatürde kompozit değişikliklerin tip ve sayısı üzerine bir mutabakata varılmamıştır (Flouris et al., 2008). Örneğin Stojanovic'in yapmış olduğu bir çalışmada (Stojanovic et al., 2002) tanımlanan kompozit değişiklik tiplerinin sayısı oniki iken, Klein ve arkadaşları tarafından yapılan bir çalışmada (Noy and Klein, 2004) yirmiiki kompozit değişiklik tipi tanımlanmış ve değişikliklerin ayrıntı düzeyine göre tanımlanabilen kompozit değişiklik tiplerinin değişim gösterebileceği ifade edilmiştir. Ontolojilerde yapılabilecek basit değişiklikler ontolojilerin ifade edildiği dille doğrudan bağlantılıdır (Klein, 2004). Örneğin, RDFS'te ifade edilmiş bir ontolojide owl:inverseOf (birbirinin tersi olan özellikleri belirtmek için kullanılır) ifade edilemez. Buna karşın, birçok basit operasyondan (Y. et al., 2005) meydana gelen kompleks operasyonlar tanımlanabilir.

Kompleks değişiklikler içerdikleri basit değişiklikler haricinde değişimle ilgili ek bilgiler de içerirler. Örneğin bir sınıfın altsınıflarının başka bir sınıfın altına taşınması basit bir sil ve yarat operasyonunun ötesinde sınıfların

örneklerinin de taşınmasını gerektirir. Dolayısıyla kompleks bir değişikliğin basit değişikliklerle ifade edilmesi değişiklikteki anlamın kaybolmasına neden olabilir. Dahası, kompleks değişiklikler değişiklik operasyonlarının etkisinin daha net belirlenmesine ve değişikliğin amacının daha net anlaşılmasına olanak tanır (Zablith et al., 2015). Örneğin, bir özelliğin değer kümesinin değiştiğini bilmek eldeki mevcut verinin üzerinde ne gibi bir değişim oluştuğu hakkında fikir vermezken, değer kümesinin genişlediğini bilmek mevcut verilerin hala geçerli olduğunun çıkarsanabilmesine olanak tanır.

Ontoloji Değişikliklerinin İfade Edilmesi

İki ontoloji versiyonu arasındaki değişiklikler değişik şekillerde ifade edilebilir. Her yöntem farklı seviyelerde olan farklı bilgiler verir. Bu yöntemler şöyle özetlenebilir (Klein, 2004):

- Değişikliklerin kaydının tutulması: Bu yöntem genellikle ontolojilerin geliştirildiği ortamlar için uygundur. Bir ontoloji geliştirilirken üzerinde yapılan tüm değişikliklerin kaydı yapıldıkları sırayla tutulur. Kütük ("log") denilen bu kayıtlardaki değişiklikler genellikle basit değişikler silsilesi şeklinde ifade edilir. Anlamsal Veb gibi dinamik ve merkezi olmayan ortamlarda kütükler bulunamayabilir.

- Yapısal fark: Ontolojinin eski versiyonundaki kavramlardan yeni versiyonundaki eşdeğerlerine bir eşleme ile birlikte silinen ve eklenen kavramların bir listesi tutulur. Bu amaçla bazı kurallar ve sezgisel yöntemler kullanılır. PROMPTDIFF (Noy and Musen, 2002; Noy and Musen, 2004; Noy et al., 2006) yapısal fark çıkaran araçlara örnek olarak verilebilir. PROMPTDIFF fark çıkarırken ontolojilerdeki kavramlardan, eğer aranan kavramlar mevcut değilse (Y. et al., 2005) ontoloji bileşenleri arasındaki yapısal ilişkilerden faydalanır. Yapısal fark iki ontoloji versiyonu arasındaki farkları içerse de bu farkları oluşturan operasyonların neler olduklarını içermez.

- Kavramsal ilişkiler: Ontolojinin eski versiyonundaki ve yeni versiyonundaki eşdeğer kavramlar arasındaki kavramsal ilişkilerin (kapsama ya da eşdeğer olma gibi) açık fakat muhtemelen eksik ifadesidir.

- Dönüşüm kümesi: Bir dönüşüm kümesi bir ontolojinin yeni versiyonuna dönüşebilmesi için gerekli olan değişim operasyonlarını içerir. Dönüşüm kümeleri daha önceden tanımlanmış olan bir değişim ontolojisindeki değişim

tiplerini kullanırlar. Bir değişim kümesi eşsiz değildir. Genellikle herhangi bir dönüşümü gerçekleştirebilecek birçok dönüşüm kümesi vardır. Bir dönüşüm kümesindeki değişimler sınıf, özellik ya da örnek ("instance") yaratan değişimler hariç istenen sırada uygulanabilir. Sınıf, özellik ya da örnek yaratan dönüşümlerin en önce uygulanması gerekir. Dönüşüm kümeleri kompleks değişimler de içerebilir.

Dönüşüm kümeleri aşağıdaki özellikleri ile kütüklerden ayrılırlar:

- Kütükler gerçekleşen tüm operasyonları gerçekleştikleri sıra ile tutar. Dönüşüm kümeleri ise sadece hedef değişimi gerçekleştirmek için gerekli olan değişimleri içerir.
- Kütüklerin içeriğinin sıralı olmasına karşın dönüşüm kümelerinin içeriği sıralı değildir.
- Kütükler gerçek değişimi ifade ettikleri için eşsizdir. Bir değişim kümesinin eşsiz olması gerekmez.
- Kütükler genellikle düşük seviyeli değişimleri içerir.

Minimal bir dönüşüm kümesi ise sadece bir ontolojiyi yeni versiyonuna dönüştürmek için yeterli ve gerekli olan operasyonları içerir.

Klein'a (Klein, 2004) göre anlaşılabilir bir değişiklik ifadesi aşağıdaki bileşenleri içermelidir:

- Tanımlayıcı üst veri: Versiyonun tarihi ve numarası, değişiklikleri yapan kişi gibi bilgileri içerir. Daha çok ontolojilerin işbirlikli geliştirildikleri ortamlarda gereklidir.
- Minimal dönüşüm kümesi: Değişimin tam operasyonel ifadesidir. Değişikliğin tekrar oluşturulması, tercüme edilmesi, veri kümelerinin yeniden yorumlanması ve değişiklik hakkında ek bilgilerin edinilebilmesi için kullanılabilir.
- Kavramsal ilişkiler: Ontoloji versiyonlarındaki kavramların birbirleriyle ilişkilerini içerir. Kavramsal ilişkiler ontolojilerin değişik versiyonları kullanılarak tanımlanan veri kaynaklarının sorgulanmasını ve yorumlanmasını destekleyerek veri erişimini kolaylaştırır.

- Kompleks değişiklikler: Bazı değişikliklerin yüksek seviyede ifade edilmesidir. Kompleks değişiklikler minimal dönüşüm kümesi ile birlikte veri dönüşüm betiklerinin (“script”) oluşturulmasında kullanılabilir. Kompleks değişiklikler, değişikliklerin özel mantıksal sorgular ve verinin erişilebilirliği üzerindeki etkisinin anlaşılmasını olanaklı kılar.

- Değişikliğin gerekçesi: Değişikliğin ne amaçla yapıldığını belirtir.

Ontolojiler üzerinde yapılan değişikliklerin daha sonra tekrarlanabilmesi, geri çevrilebilmesi, eski verilere yeni ontolojiler üzerinden ulaşılabilmesi ve değişimin gereksinim duyan uygulamalara iletilebilmesi için, bu değişikliklerin standart bir dil kullanılarak kaydedilmeleri gerekmektedir.

Ontoloji değişimlerinin kullanılan ontoloji diline bağımlı olması nedeni ile değişimlerin ifade edildiği kayıtlar da aynı ontoloji dilinde gerçekleştirilebilecek değişiklikleri betimleyecek şekilde geliştirilmelidir. Bu tez kapsamında gerçekleştirilen sistem OWL dilindeki ontolojileri hedef almaktadır, dolayısıyla değişiklik gösteriminin de OWL ontolojilerindeki değişimleri hedef alması gerekmektedir.

Literatürde OWL dilinde ifade edilmiş ontolojilerdeki değişimleri ifade etme üzere geliştirilmiş çeşitli değişim ontolojileri vardır. Bu ontolojilerde ontoloji değişimi ile ilgili temel kavramlar ve bunların birbirleri ile ilişkileri tanımlanarak ontoloji değişimlerinin tarif edilebileceği bir dil oluşturulmaktadır. Değişim ontolojilerinde gerçek değişimler örnek verisi olarak tanımlanırlar. Burada bahsi geçen değişim ontolojileri sırasıyla Klein (Klein, 2004), Noy ve arkadaşları (Noy et al., 2006), Plessers ve De Troyer (Plessers and Troyer, 2005; Peter Plessers and Casteleyn, 2007) ve Palma ve arkadaşları (Palma et al., 2009; Palma et al., 2008) tarafından geliştirilen ontolojilerdir.

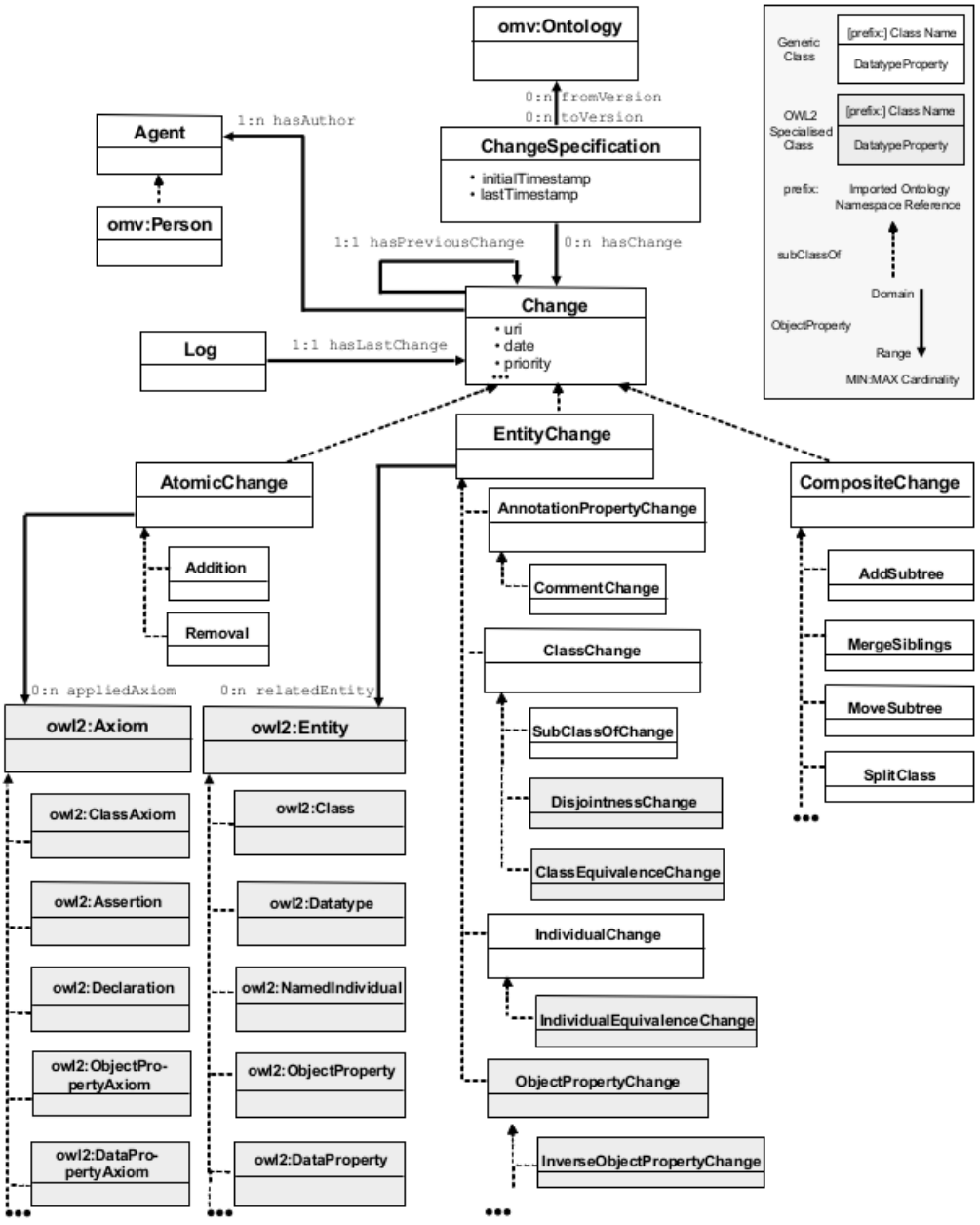
Klein (Klein, 2004) tarafından geliştirilen değişim ontolojisindeki en genel operasyon değişim operasyonudur (“change operation”). Bu kavramın iki altsınıfı vardır. Biri kompozit operasyon (“composite operation”), diğeri ise atomik operasyondur (“atomic operation”). Kompozit operasyonlar önceden tanımlanmış operasyonlar olabilecekleri gibi istenen sayıda atomik operasyonlardan da oluşabilirler. Bu amaçla içerir (“consists of”) ilişkisi ile atomik operasyonlar bir kompozit operasyona bağlanabilir. Atomik operasyonların bu şekilde bir kompozit operasyon altında bağlanmaları birbiriyle ilişkili operasyonların gruplanmasını ve paylaştıkları özelliklerin ifade edilebilmesini olanaklı kılar. Bir kompozit operasyonun tüm özelliklerinin kendisinin bileşenleri için de geçerli olduğu

kabul edilmektedir. Değişim operasyonu kavramı, değişim geçiren ontolojinin içerdiği kavramların tanımlarına "from" (eski ontoloji versiyonundaki kavramlar için kullanılır) ve "to" (yeni ontoloji versiyonundaki kavramlar için kullanılır) özellikleri ile bağlıdır. Tanımlama (“definition”) sınıfı, sınıfları, özellikleri veya örnekleri tanımlayan kaynakların altkümesidir. Bir değişim operasyonları kümesi değişim tanımlaması (“change specification”) olarak adlandırılır. Bir değişim tanımlamasının kavramların ve özelliklerin tanımlandığı kaynak ve hedef ontolojileri vardır. Bir ontoloji ve tanımlamalar arasındaki tanımlar (“defines”) ilişkisi ontolojinin içeriğinden çıkarsanmaktadır. Eğer bir ontoloji bir tanımlamayı içeriyorsa, ontolojinin o kaynağı tanımladığı söylenebilir.

Noy ve arkadaşları tarafından geliştirilen CHAO “Collaborative Protege“ (Tudorache et al., 2008; Tudorache, 2015) projesi kapsamında kullanılmıştır. CHAO ontolojilerin üzerindeki değişiklikleri ve bunlarla ilgili meta-bilgileri (yazar, tarih, vs.) tutan örnekler içermektedir. CHAO’da her değişim için bir değişim tipi, değişen sınıf, özellik ya da örnek, değişimin tarihi gibi bilgiler tutulabilir. Bu ontolojide iki temel sınıf vardır: “Change” ve “Annotation” sınıfları. Bu iki sınıf birbirlerine bir çift ters (“inverse”) özellikle bağlıdır. “Change” sınıfının değişimleri daha özel türlere ayrıştıran özellik ekleme, kısıt silme, sınıf ekleme, vb. alt sınıfları bulunmaktadır. Bir grup değişimi tek bir değişim altında toplamaya yarayan “Composite_Change” sınıfı da bulunmaktadır.

Plessers ve arkadaşları (Plessers and Troyer, 2005; Peter Plessers and Casteleyn, 2007) tarafından geliştirilen ontoloji ise versiyon kütüğü olarak adlandırılmaktadır. Versiyon kütüğü ontolojideki her kavramın ontoloji değişirken oluşan tüm versiyonlarını tutar. Ontolojide yaratılan her sınıf, özellik ve örnek için versiyon kütüğünde bir örnek oluşturulur. Bu örnekler aslında EvrimKavramı (“EvolutionConcept”) sınıfının örnekleridir. Böyle bir EvrimKavramı sınıfı ontolojide ilişkili olduğu kavram ile bu kavramın eski ve yeni versiyonlarına atıfta bulunur. Ontolojideki bir kavram için değişiklik istemi olduğunda yeni bir Versiyon (“Version”) sınıfı örneği oluşturulan EvrimKavramı örneğine eklenir. Böyle bir Versiyon örneği ontolojide ilgili değişikliğin olduğu zamanı, değişimin nedenlerini, değişimin onaylanıp onaylanmadığını ve kavramın kimliğini tutar. Versiyon sınıfının ÖrnekVersiyonu (“InstanceVersion”), ÖzellikVersiyonu (“PropertyVersion”) ve SınıfVersiyonu (“ClassVersion”) altsınıfları vardır.

Palma ve arkadaşları (Palma et al., 2009; Palma et al., 2008) tarafından tanımlanan OWL için Değişim Ontolojisi (ODO-“Change Ontology for OWL”) ise NeOn projesi kapsamında geliştirilmiş olup daha sonra incelenecek olan OMV (“Ontology Metadata Vocabulary”) ontolojisini ve OWL-ODM (“OWL Object



Şekil 2.2: ODO temel sınıfları ve özellikleri (Palma et al.'dan, 2008a)

Definition Meta-model") ontolojisini kullanmaktadır. ODO ontolojisinin yapısı şekil 2.2'de görülebilir. Bu değişim ontolojisinin OWL 2 için tanımlanmış yeni bir sürümü de mevcuttur (Palma et al., 2009). OWL için Değişim Ontolojisi'nde tüm olası ontoloji değişimleri Değişim ("Change") sınıfından türetilmiştir. Değişim sınıfının AtomikDeğişim ("AtomicChange"), VarlıkDeğişimi ("EntityChange") ve KompozitDeğişim ("CompositeChange") olmak üzere üç altsınıfı vardır.

Atomik değişimler ekleyici ve çıkarıcı değişiklikleri temsil eden Ekleme ("Addition") ve Çıkarma ("Removal") altsınıflarına ayrılmaktadır. Sık rastlanan bazı kompozit değişimler ise KompozitDeğişim sınıfının alt sınıfları olarak tanımlanmıştır. Atomik değişimleri ve varlık değişimlerini uygun atomik elemanlarla/operasyonlarla (örneğin, aksiyomlara) ve varlıklarla ilişkilendirmek için sırasıyla uygulananAksiyom ("appliedAxiom") ve ilişkiliOlanVarlık ("hasRelatedEntity") nesne özellikleri ("object properties") kullanılmaktadır. Bir VarlıkDeğişimi'nin birden fazla AtomikOperasyon'dan oluştuğunu belirtebilmek için atomikOperasyonlardanOluşur ("consistsOfAtomicOperations"), bir kompozit değişikliğin ise başka değişikliklerden meydana geldiğini belirtebilmek için ise içerir ("consistsOf") özellikleri tanımlanmıştır. Ayrıca, belli bir ontoloji versiyonuna uygulanan tüm değişikliklerin gruplanabilmesi için, değişen ontolojinin önceki ve mevcut versiyonlarını belirtmek amacıyla OMV ontolojisinden gelen Ontoloji ("Ontology") örnekleriyle ilişkilendirilebilen bir DeğişimTanımı ("ChangeSpecification") sınıfı tanımlanmıştır. Değişimlerin sırasının tutulabilmesi için ise Log sınıfı ve öncekiDeğişimiVardır ("hasPreviousChange") özelliği tanımlanmıştır.

Ontoloji Değişikliklerinin Bulunması

Ontoloji versiyonlamak için gerekli olan ontolojilerin karşılaştırılması işlemi için kullanılabilecek yöntemler şunlardır (KWTR, 2011; Kremen et al., 2011; Konev et al., 2008a):

- Tamamen sözdizimsel: Yazılım versiyon yönetimine benzer şekilde sadece söz dizimsel farkların çıkarılmasıdır. Değişimin anlamı yakalanmaz.
- İşlem tabanlı: Veritabanı teorisinden esinlenilmiştir. Değişiklikler mevcut veri kaybedilmeyecek şekilde işlem ("transaction") mekanizması kullanılarak ele alınır.
- Birleştirme ("merge") tabanlı: Mevcut yarı otomatik ontoloji birleştirme teknolojisi bir ontolojinin iki versiyonu arasındaki farkı bulmak için

kullanılır.

- Söz dizimsel tekniğin anlamsal uzantıları: Anlamsal ve söz dizimsel karşılaştırmanın birarada gerçekleştirilmesini gerektirir. İki ontoloji arasındaki fark, her iki ontolojiye de kendilerinden mantıksal olarak çıkarsanabilen ifadelerin eklenmesinden sonra söz dizimsel karşılaştırma tekniğinin uygulanması ile elde edilir. Gerçekleştirim kullanılan ontoloji dilinin anlamına dolayısıyla da dilin kendisine bağımlıdır.

- Mantıksal: Bu yaklaşımda çıkarsama metotları yardımıyla iki ontoloji versiyonun anlamları arasındaki farklar bulunur.

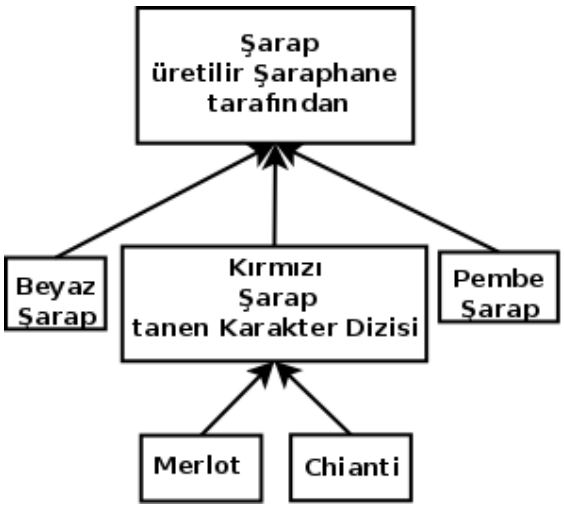
Ontolojilerin karşılaştırılması şu şekilde de sınıflandırılabilir (Volkel, 2006):

- Küme tabanlı karşılaştırma: İki RDF üçlü kümesinin farkının alınması ile elde edilir. Jena gibi kütüphanelerde bu farkları almak için gerekli olan yerleşik ("built-in") fonksiyonlar vardır. Küme tabanlı karşılaştırmada hiç çıkarsama yapılmaz.

- Yapısal karşılaştırma: Küme tabanlı karşılaştırmadan farkı boş düğümlerin de dikkate alınmasıdır. Küme tabanlı karşılaştırma bir ontolojideki tüm boş düğümleri farklı düğümler olarak gördüğünden bu düğümleri içeren ifadeleri hem eklenmiş hem de çıkarılmış olarak rapor eder.

- Anlamsal karşılaştırma: Anlamsal karşılaştırma kullanılan ontoloji dilini de dikkate almak zorundadır. Dolayısıyla, yazılacak programlar da kullanılan ontoloji diline bağımlı olacaklardır. İki ontoloji arasındaki fark, en basit haliyle, her iki ontolojiye de kendilerinden mantıksal olarak çıkarsanabilen ifadelerin eklenmesinden sonra söz dizimsel karşılaştırma tekniğinin uygulanması ile elde edilir. Aralarında yapısal olarak fark olmayan ontolojilerin anlamsal olarak da farklı olmadıkları söylenebilir (Volkel, 2006).

Ontolojilerin karşılaştırılmasında sözdizimsel karşılaştırmanın yetersiz olduğunu ve sezgisel yöntemlerle desteklenmesi gerektiğini savunan araştırmacılar da vardır (Noy et al., 2006; Noy and Musen, 2002). Bu araştırmacılara göre ontolojilerin serileştirilmeleri için farklı yollar olduğu, hatta ontoloji içindeki kavramların tanım sıraları değişebileceği için metin tabanlı karşılaştırma yanlış sonuçlar verecektir. Dolayısıyla sezgisel yöntemleri savunan araştırmacılar (Noy and Musen, 2002; Noy and Musen, 2004) yapısal karşılaştırmayı daha önce verilen tanımlamadan (Volkel, 2006) farklı olarak, ontolojinin yapısının sezgisel yöntemler ve kurallar kullanarak



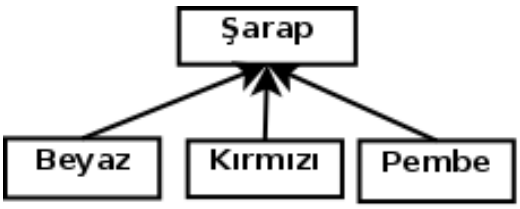
Şekil 2.3: Şarap Ontolojisinden Bir Parça (Noy et al.'dan, 2002)

karşılaştırılması ve böylelikle hem basit hem de karmaşık değişikliklerin bulunması şeklinde tanımlamaktadır.

Ontolojilerdeki Değişiklikleri Bulmak İçin Geliştirilmiş Araçlar

PROMPTDIFF: Noy ve Musen (Noy and Musen, 2002; Noy and Musen, 2004) tarafından geliştirilmiş sezgisel yöntemleri destekleyen bir yapısal karşılaştırma aracıdır. PROMPTDIFF algoritması iki parçadan oluşur: (1) genişletilebilir bir sezgisel eşleyiciler kümesi ve (2) iki versiyon arasındaki yapısal farkı oluşturmak üzere bu eşleyicilerin sonuçlarını birleştiren sabit nokta ("fixed-point") algoritması. Her eşleyici ontolojilerin az sayıda yapısal özelliğini inceleyerek eşlemeler oluşturur. Sabit nokta algoritması ise eşleyicileri birinin sonucunu diğerine vererek yeni bir fark belirlenemeyinceye kadar defalarca çalıştırır. Algoritma bir yapının ("frame") eski ve yeni versiyonlarda eşlenemediği durumları ikiye ayırmaktadır: eşşekilli ("isomorphic"): yapıların içerdiği değerler birbirinin yansıması ("image") olmasına karşın aynı değildir, değişmiş: yapılar birbirinin yansıması olmayan değerler içermektedir.

Sezgisel eşleyicilerin işleyişine örnek verebilmek için Şekil 2.3'de verilen ontolojinin bir parçasının Şekil 2.4'te gösterilen şekle dönüştürüldüğünü varsayalım (Noy and Musen, 2002). Bu değişikliğin bulunması için aynı öneki ya da soneki alan kardeş yapıları değerlendiren sezgisel eşleyici kullanılabilir. Bu örnekte ontolojinin bir önceki versiyonunda isminin sonunda şarap olan tüm sınıfların isimleri bu son ek silinerek değiştirilmiştir. Bu özellik kullanılarak eşleme yapılabilir. Genellikle, eğer bir ontolojinin iki versiyonunda iki sınıf eşleşiyorsa



Şekil 2.4: Şarap Ontolojisinin Değişimden Sonraki Hali (Noy et al.'dan, 2002)

ve bu sınıfların alt sınıflarının isimleri bir ön ya da son ek haricinde aynıysa bu alt sınıfların eşleştiği söylenebilir.

ONTOVIEW: Klein (Klein, 2004) tarafından geliştirilen OntoView sistemi de ontolojileri yapısal düzeyde karşılaştırmaktadır. Bu sistemdeki karşılaştırma fonksiyonu ekleme, çıkarma ve değişiklik işlemlerini tanımlayabilmektedir.

Klein'a göre (Klein, 2004) ontolojilerdeki değişikliklerin belirlenmesinde iki temel problem vardır. İlk problem değişikliklerin saptanacağı soyutlama düzeyidir. Gösterimdeki değişimlerin anlamı etkileyip etkilemediklerinin belirlenebilmesi için soyutlama şarttır. Genellikle aynı ontolojik tanımlama değişik şekillerde yapılabilir. Örneğin, sınıflar RDFS dilinde rdf:type ifadesi ile ya da <rdf:Class > ifadesi ile tanımlanabilir. Gösterimin bu iki tanımlama kullanılarak değiştirilmesi anlamı etkilemez. Dolayısıyla, değişikliklerin belirlenmesinde gösterim farkları tek başına yeterli değildir.

Buna karşın çok yüksek düzeyli soyutlama da problemlere neden olabilir. Sadece mantıksal anlamın dikkate alınması da yeterli değildir. Değişik ontolojik tanımlamalardan oluşan kümelerin aynı aksiyom kümesini üretebileceği ispatlanmıştır. Bu durumlarda mantıksal anlam değişmemiş olsa da ontoloji değişmektedir.

İkinci problem ise doğru soyutlama düzeyi bulunduğunda bile değişimin kavramsal sonuçları açık değildir. Kavramsal ve gösterimsel değişimler arasındaki farklar nedeni ile sadece görünen değişimden mantıksal sonuçların çıkarılması mümkün değildir. Bu amaçla sezgisel yöntemler kullanılabilir.

OntoView aracının çalışması ise şöyle özetlenebilir :

1. Ontoloji dokümanı bir XML dokümanı olarak ele alınır ve parçalara ayrılır.
2. Daha sonra elde edilen tanımlamalar RDF üçlüleri şeklinde ayrıştırılır. Böylelikle küçük çizgelerden ("graph") oluşan bir küme meydana getirilmiş

olur. Bu çizgelerden herbiri bir kavram ya da özelliği tanımlar.

3. İlk iki basamakta verilen işlemler ontolojinin yeni versiyonu için de gerçekleştirilir.

4. Eski versiyondaki her kavrama ya da özelliğe ait olan çizge, yeni ontolojide kendisine karşılık gelen çizge ile eşlenir. Daha sonra bu çizgeler belli sezgisel kurallara göre incelenir.

Kuralların formatı şöyledir:

Eğer eski versiyonda <A, Y, Z> varsa, yeni versiyonda <X, Y, Z> varsa ve yeni versiyonda <A, Y, Z> yoksa, değişim tipi A'dır. Bu formata göre özellik tipinde bir değişiklik yapılması ile ilgili bir kural örneği şöyle olabilir:

```
IF exist:old
    <X, rdf:type, rdf:#Property>
    <X, rdf:type, daml:#TransitiveProperty>
exist:new
    <X, rdf:type, rdf:#Property>
not-exist:new
    <X, rdf:type, daml:#TransitiveProperty>
THEN Unset_Transitivity
```

Klein'ın yaklaşımında (Klein, 2004), anlamsal karşılaştırmaya benzer şekilde, öncelikle ontolojiden çıkarsanabilecek tüm ifadelerin çıkarsanması gerekmektedir.

ONTODIFF: Tury ve Bielikova (Tury and Bielikova, 2006) tarafından geliştirilen ONTODIFF sistemi de sezgisel yöntemleri kullanarak yapısal karşılaştırma yapan bir sistemdir. ONTODIFF sisteminde bir ontolojinin eski ve yeni versiyonunda farklı olan yapılar iki değişik türde sınıflandırılmaktadır: izomorfik değişim ve anlamsal değişim. İzomorfik değişimde ontoloji bileşeninin içeriği değişmez ama ismi değişmiştir. Anlamsal değişimde ise ontoloji bileşeninin iç yapısı değişmiştir.

SemVersion: Völkel ve arkadaşları (Volkel, 2006; Volkel and Groza, 2006) tarafından geliştirilmiş bir versiyonlama aracı olup, amacı gereği ontolojilerin

karşılaştırılması işlevini de taşımaktadır. SemVersion hem yapısal hem de anlamsal karşılaştırmayı destekler. Ayrıca, değişik amaçlarla kullanılabilmesi ve genişletilebilmesi için bir Uygulama Geliştirme Arayüzü (UGA-"Application Programming Interface") de vardır.

OWLDIFF: İki ontolojinin arasındaki mantıksal, anlamsal ve sözdizimsel farkı çıkarabilen açık kaynak kodlu bir araçtır (Kremen et al., 2011; OWLdiff, 2015). OWLDIFF OWL 1 dili ile yazılmış ontolojilerin yanı sıra CEX fark algoritmasını (Konev et al., 2008a; Konev et al., 2008b) kullanarak OWL 2 dili ile yazılmış ontolojilerin de mantıksal farklarını çıkarabilmektedir.

Ontolojilerde Oluşan Değişikliklerin Etkileri

Ontolojiler Anlamsal Veb üzerinde kontrolsüz bir şekilde değiştirilebilirler. Bir ontolojinin değişmesi kendisini kullanan ya da kendisine atıfta bulunan diğer ontolojileri, servisleri, veriyi, uygulamaları, değişen ontoloji ile diğer ontolojiler arasında eşlemeleri ("mappings"), ve hatta kendisi hakkında bilgi tutan üst veri kaynaklarını dahi etkileyebilir (Palma et al., 2009; Flouris and Plexousakis, 2005). Dolayısıyla, bir ontoloji değiştiğinde bu değişim, ontoloji ile bağlantılı olan tüm üst veriye, ontoloji örneklerine, değişen ontolojiyi kullanan diğer ontoloji ve uygulamalara aksettirilmelidir (değişiklik yayılımı-"change propagation") (Klein and Fensel, 2001; Haase, 2004; Palma et al., 2009; Stojanovic, 2004). Kimi uygulamalar kullandıkları ontolojilere bağımlı olarak katı şekilde kodlanmış olduklarından ("hard coded") bu uygulamaların sorgularının bir ara bileşen tarafından ontolojilerin yeni versiyonlarında çalışabilecek şekilde tercüme edilmesi de gerekebilir (Liang et al., 2006b; Liang, 2006).

2.1.4 Ontoloji değişimlerinin yönetilmesi

Ontoloji değişimlerinin etkileri göz önüne alındığında ontoloji değişimlerinin yönetilmesinin gerekliliği ortaya çıkmaktadır. Bu amaçla sürdürülen araştırmalar ontoloji versiyonlama ve ontoloji evrimi olarak ikiye ayrılmaktadır.

Ontoloji versiyonlama değişmekte olan bir ontolojinin değişik versiyonlarını, değişimin değişen ontolojiye bağlı olan ontolojiler, bağlı veri kaynakları, etmenler, uygulamalar ve servisler üzerinde yaratacağı ters etkileri minimize edecek şekilde yönetir. Bu amaçla ontolojiye erişmeye çalışan elemana bağımlı olarak (veri kaynakları vs. ile ontoloji versiyonlarının otomatik olarak ilişkilendirilmesidir

(Klein and Fensel, 2001)) ontolojinin güncel ya da daha eski sürümlerine şeffaf erişimin sağlanması gerekir. Böylelikle ontolojileri kullanan elemanların yeni versiyona göre güncellenmesi işleminin zamanı konusunda esneklik sağlanmış olur (Flouris et al., 2008). Ontoloji versiyonlama tipik olarak ontolojinin eski ve yeni tüm versiyonlarının saklanmasını içerir. Bu amaçla tanımlama ("identification") konusunun (örneğin, bir ontolojinin değişik versiyonlarına nasıl kimlik verileceği), değişik versiyonlar arasındaki ilişkilerin ve uyumlulukla ilgili bilginin edinilip saklanmasının dikkate alınması gerekmektedir (Flouris et al., 2008).

Ontoloji evrimi mevcut bir ontolojinin içine yeni bilginin nasıl yerleştirileceği ile, dolayısıyla da değişimlerin kendileriyle uğraşmaktadır. Yeni bilgilerin mevcut bilgilerle çelişkiye düşmesi olasılığı olduğundan, problemin bir kısmı da değişimin ontolojide çelişkilere yol açmasını önlemektir (Flouris et al., 2008).

Ontoloji evrimi değişik şekillerde tanımlanmıştır:

- Ontoloji evrimi ontolojinin ait olduğu alandaki değişikliklere hem kendinin hem de kendisine bağlı olan bileşenlerin tutarlılığını koruyarak adapte olması sürecidir (Peter Plessers and Casteleyn, 2007).

- Ontoloji evrimi bir ontolojinin oluşan değişikliklere vaktinde adapte olması ve bu değişiklikleri tutarlı bir şekilde kendisine bağımlı olan bileşenlere iletmesidir (Stojanovic, 2004).

- Ontoloji evrimi bir kaynak ontolojinin üzerinde bir değişim operasyonları kümesi uygulanarak ontolojinin ait olduğu alandaki ya da ontolojinin kavramsallaştırılmasındaki değişimlere tepki vermeyi amaçlayan bir süreçtir (Flouris et al., 2008).

Bazı çalışmalarla ontoloji versiyonlamanın tanımı hem değişimlerin ontolojilerin içine yerleştirilmesi hem de her değişim kümesi sonucunda yeni bir ontoloji versiyonu tanımlayarak bu versiyonlara şeffaf erişimi içerecek şekilde yapılmakta, dolayısıyla da ontoloji versiyonlamanın ontoloji evrimini de içeren daha güçlü bir kavram olarak tanımlanmaktadır (Haase, 2004).Ontolojilerin eski versiyonlarının saklanmadığı durumlarda saf ontoloji evrimi gerçekleştirildiği söylenebilir (Flouris et al., 2008).

2.2 Bağlı Veri

Anlamsal Veb fikri Tim Berners Lee ve arkadaşları tarafından ortaya atıldığında (Berners-Lee et al., 2001) Anlamsal Veb'in birbirine bağlı ontolojilerden oluşan evrensel bir bilgi ağı ve bu ağ üzerinde çalışan akıllı etmenlerden oluşacağı öngörülmüştür. Ancak beş yıl sonrasında, 2006'da, Anlamsal Veb birbiriyle bağlantısı olmayan ayrık sistemlerde ayrık ontolojilerle çalışan ayrık etmen sistemleri görünümündeydi (Dikenelli, 2014). Bu nedenle Tim Berners Lee ve arkadaşları 2006 yılında Anlamsal Veb'in hedefini daha açık şekilde tanımlayan Bağlı Veri (Shadbolt et al., 2006) fikrini ortaya attı.

Bağlı Veri'de temel amaç Veb üzerinde farklı kaynaklardan gelen, anlamı açıkça tanımlanmış dolayısıyla da makinalar tarafından da anlaşılabilen verileri içeren veri kümelerini tip tanımlaması bulunan anlamsal bağlarla birbirine bağlayarak Dünya çapında bir Veri Ağı oluşturmaktır (Bizer, 2009; Bizer et al., 2009). Bu yönüyle Bağlı Veri Anlamsal Veb fikrinin pratiğe dökülmüş halidir de denebilir (Hausenblas, 2009). Bağlı Veri Anlamsal Veb teknolojileri ve standartları kullanılarak yayınlanmakta ve bağlanmaktadır, dolayısıyla da Anlamsal Veb'in Tim Berners Lee tarafından ortaya atılan ilk amacı olan Dünya çapında makinelerin de işleyebileceği veri ağının oluşumunu sağlamaktadır (Yu, 2011).

Bağlı Veri RDF verisi içeren dokümanların tipi belli olan bağlantılarla birbirine bağlanması ve bu dokümanlara HTTP ("Hypertext Transfer Protocol") üzerinden ulaşılması esasına dayanır. Dokümanlarda tanımlanan varlıkları birbirine bağlayarak önermeler oluşturmak için RDF kullanılır. Bu dokümanlarda ontolojilerde OWL dili ile tanımlanan kavramlar da kullanılabilir. Böylelikle Veri Ağı ("Web of Data") oluşur. Veri Ağı veri ile tanımlanan varlıkların oluşturduğu bir ağ olarak tanımlanabilir (Bizer, 2009).

Bağlı Veri ağ üzerinde bilgi yayınlamanın esaslarını oluşturan ve yayınlanan verinin Dünya çapında genel bir veri ağının parçası haline gelmesini amaçlayan ve Bağlı Veri Prensipleri ("Linked Data Principles") olarak da isimlendirilen dört temel kurala dayanmaktadır (Lee, 2006) :

1. Varlık isimleri olarak URI'ler kullanılmalıdır.
2. İnsanların da varlık isimlerini kullanarak tanımlamalara erişebilmeleri için HTTP URI'leri kullanılmalıdır.
3. Bir kişi bir URI'ye eriştiğinde, standartlar kullanılarak (RDF, RDFS)

kendisine fayda sağlayacak bilgi edinebilmelidir.

4. Başka URI'lere bağlantılar da sağlanarak insanların başka bilgilere de erişebilmeleri olanaklı kılınmalıdır.

2.2.1 Veri Ağı ("Web of Data")

Bağlı veri fikrinin ortaya çıkmasından sonra bir çok kuruluş sahip olduğu veriyi Veb üzerinde yayınlamaya başlamış ve Veri Ağı ("Web of Data") olarak isimlendirilen küresel bir veri uzayı ("dataspace") meydana gelmiştir (Schmachtenberg et al., 2014; Heath and Bizer, 2011). Dolayısıyla Veri Ağı'nın dünyadaki varlıkları ağ üzerindeki veri ile tanımladığı söylenebilir (Bizer et al., 2009). Veri ağının özellikleri şöyle özetlenebilir (Heath and Bizer, 2011; Bizer et al., 2009):

- Veri Ağı geneldir ve her tür veriyi içerebilir.
- Veri Ağı'nda herkes veri yayınlayabilir.
- Veri Ağı bir varlıkla ilgili birbiri ile uyuşmayan bilgiler içerebilir.
- Varlıklar RDF bağları ile bağlanmıştır. Böylelikle yeni veri kaynaklarının keşfini olanaklı kılan ve bir çok veri kaynağını içeren küresel bir ağ oluşmaktadır. Uygulamalar çalışma esnasında yeni veri kaynakları bulabilirler.
- Veri yayıncıları yayınladıkları veride kullanıdıkları söz varlığı ("vocabulary") açısından kısıtlanmış değillerdir.
- Veri kendi kendini tanımlar niteliktedir. Eğer Bağlı Veri kullanan bir uygulama tanıdık olmayan bir sözcük dağarcığı ile karşılaşırsa ilgili sözcükleri tanımlayan URI'leri çözümleyerek tanımlarına ulaşabilir.
- HTTP standart veri erişimi, RDF ise standart veri modelidir.
- Veri formatlama ve sunum ile ilgili detaylardan katı şekilde soyutlanmıştır.

Günümüzde Veri Ağı (Schmachtenberg et al., 2014) spor, insanlar, şirketler, devlet verileri, istatistiksel bilgiler gibi birçok farklı alandan milyarlarca RDF üçlüsünü içeren devasa küresel bir çizgedir (Heath and Bizer, 2011). Veri Ağı'nın içerdiği bilginin artışı ile birlikte akıllı etmenlerin Anlamsal Veb üzerinde işbirlikli çalışabilmelerini sağlayacak ortam şekillenmeye başlamıştır.

2.2.2 Bağlı Veri'nin tüketilmesi

Uygulamaların kullanacakları Bağlı Veri tüketim yaklaşımı kullanmayı hedefledikleri veri kaynağı sayısına, ihtiyaç duydukları veri güncelliğine, kullanıcı etkileşimi ve sorgu işlemede hedefledikleri tepki süresi ile çalışma zamanında yeni veri bulmayı amaçlayıp amaçlamadıkları dikkate alınarak belirlenmelidir (Heath and Bizer, 2011). Seçilen yaklaşım aşağıda incelenen üç yaklaşımdan biri olabileceği gibi bu yaklaşımların ikisinin ya da hepsinin birleştirildiği hibrid bir yaklaşım da olabilir:

- Emekleme ("crawling") modeli ("pattern"): Bu modeli gerçekleyen uygulamalar RDF bağlantıların takip ederek buldukları veriyi depolar ve sonrasında yerel önbelleğe ("cache") yüklenen veriyi temizleyip entegre ederler. Bu yaklaşım Google gibi Veb Arama motorlarının mimarisini andırmaktadır.
- Anında çözümleme ("On-The-Fly Dereferencing") modeli: Bu tüketim tipinde uygulamalar veriye ihtiyaç duydukları anda URI'leri çözümlemekte ve bağlantıları takip etmektedirler. Bu yaklaşım Bağlı Veri tarayıcılarının mimarisini andırmaktadır.
- Sorgu birleştirme ("Query Federation") modeli: Uygulamaların ihtiyaç duydukları verinin elde edilmesini sağlayacak karmaşık sorguların alt sorgularının doğrudan ilgili oldukları veri kaynaklarına gönderilmesi ve sonuçların birleştirilmesine dayanan bu yaklaşım tipi SPARQL ("Simple Protocol And RDF Query Language") teknolojisini kullanmaktadır.

SPARQL Bağlı Veri'nin sorgulanması için kullanılan standartlaşmış bir dildir. RDF üçlülerinin sorgulanması amaçlandığından SPARQL sorguları da üçlüler şeklinde ifade edilmekte ancak verilen üçlülerdeki öğelerden bir ya da daha fazlası değişken olabilmektedir. Sorgular çalıştırılırken sorgulanan RDF çizgesindeki üçlülerden sorguda verilen üçlü tanımları ile eşleşenler çekilerek sorgu yanıtı döndürülmektedir.

Dört temel SPARQL sorgu tipi vardır:

- SELECT sorguları: SELECT sorguları bir veri setinde verilen üçlü tanımlamalarına uyan RDF üçlülerini çekerek bunları bir sonuç kümesi ("resultset") olarak döndürür.

- CONSTRUCT sorguları: Bu sorgu tipi bir veri setinde verilen üçlü tanımlamalarına uyan RDF üçlülerinden oluşan bir RDF çizgesi döndürür.
- ASK sorguları: ASK sorguları verilen üçlü tanımlamaları sorgulanan veri setinde mevcut ise mantıksal doğru aksi halde mantıksal yanlış döndürür.
- DESCRIBE sorguları: Belli bir URI ile ilgili tüm bilgi çekilmeye çalışıldığında DESCRIBE sorguları kullanılabilir. Bu sorgu tipi bir veri setinde verilen URI ile ilgili tüm üçlüleri bir RDF çizgesi halinde döndürür.

SPARQL sorguları yerel veri setleri üzerinde çalıştırılabileceği gibi Anlamsal Veb üzerindeki SPARQL uç noktaları üzerinde de çalıştırılabilir. Birden fazla SPARQL uç noktasından ya da veri setinden gelen verinin birleştirilmesi ile yanıtlanabilecek sorgular federe SPARQL sorguları olarak tanımlanmaktadır. Federe SPARQL sorguları her biri farklı bir veri kaynağını hedefleyen birden fazla temel SPARQL sorgusundan meydana gelmektedir. Bu sorgu türünün çalıştırılabilmesi için sorguları uygun veri kaynaklarına ileten aracılara ihtiyaç vardır. Bu aracılara örnek olarak Jena ARQ (ARQ), FedX (Schwarte et al., 2011a; Schwarte et al., 2011b), SPLENDID (Görlitz and Staab, 2011), DARQ (Quilitz and Leser, 2008), WoDQA (Akar et al., 2012) federe sorgu motorları verilebilir. Jena ARQ en basit federe sorgu motoru gerçekleştirimlerinden biridir. İşlenen federe sorguların her birinin hangi SPARQL uç noktasına yöneltileceğinin sorgu içerisinde belirtilmesini zorunlu tutar.

FedX ise çalıştıracağı sorguların hedefleyebilecekleri veri setlerinin SPARQL uç noktalarının adreslerini alır ve bir sorgu çalıştıracağı zaman ilgili veri setinin kullanılıp kullanılmayacağına bu uçnoktalara ASK SPARQL sorguları atarak karar verir.

DARQ sorgu geliştiricilerinin manüel olarak hazırladıkları Servis Tanımlaması (“Service Description”) olarak adlandırılan veri seti üstverisini kullanmaktadır. Sorgu planının optimizasyonunda ise varlık ve üçlü sayısından yararlanılmaktadır.

SPLENDID ve WODQA ise federe sorguların işletilmesinde Bağlı Veri’yi tanımlayan VoID (Vocabulary of Interlinked Datasets) üstverisinden yararlanır. SPLENDID de sorgu optimizasyonunda ASK sorgularından destek almaktadır. Buna karşın WODQA federe sorgu çalıştırmak için sadece VoID üstverisini kullanmaktadır.

VoID olarak adlandırılan ve RDF veri kümeleri hakkında üst bilgilerin ifade edilmesinde kullanılan RDF şema sözlüğüdür (Alexender vd., 2009).

VoID, veri kümelerinin keşfinde, kataloglanmasında veya arşivlenmesinde kullanılabilmektedir. VoID ile tanımlanabilecek üst veri tipleri şunlardır:

- Genel üst veri: Bir veri setinin ismi, bu seti oluşturan kuruluş, yayıncı ve oluşturulduğu tarih gibi genel üst veriler "Dublin Core" sözlüğünden alınan terimler ile tanımlanmaktadır.

- Erişim üst verisi: Veri setlerine erişilirken kullanılacak SPARQL uç noktalarının adresleri VoiD sözlüğünün tanımları ile ifade edilmektedir.

- Yapısal üst veri: Yapısal üst bilgi bir veri setinin oluşturulmasında kullanılan kavramların ait oldukları ontolojileri voiD sözlüğünü kullanarak tanımlar.Bu üst bilgi türü de voiD sözlüğü ile tanımlanmaktadır. Bu grup altındaki en önemli üst bilgi ilgili veri kümesinde kullanılan sözlüklerin (ontolojilerin) neler olduğuna ilişkin bilgilerdir. VoID ile tanımlanan dördüncü üst bilgi türü ise, iki veri kümesi arasında ilişkilendirilmiş olan üçlülerden oluşan ve bağlantı kümesi ("linkset") olarak adlandırılan yapılar hakkındaki bilgilerdir. Hangi veri kümesinin özneleri, hangisinin nesneleri sağladığı ve bağlantıya ilişkin üçlülerin hangi veri kümesinde bulunduğuna dair bilgiler örnek olarak verilebilecektir. Bu projede ortaya konan anlamsal ortamdaki çalışma alanları da VoID depoları içermektedir. Bu VoID depoları, anlamsal ortamın bir parçası olarak kullanılan bağlı veri sorgu motorunun verilen bir SPARQL sorgusunu işletmek için hangi veri kümelerini sorgulaması gerektiğini belirlemektedir.

2.2.3 Bağlı Veri dinamikleri

Anlamsal Veb'in pratik uygulaması olarak da tanımlanan Bağlı Veri Uzayı da sürekli değişen, dinamik bir yapıya sahiptir. Bağlı Veri Uzayı'nda yer alan kaynaklar silinebilir, taşınabilir ya da değiştirilebilir, kaynakların arasındaki bağlantılar silinebilir, yeni bağlantılar yaratılabilir, birçok üçlüyü içeren çizgeler ve verisetleri tamamen değiştirilebilir ya da silinebilir. Bağlı veri uzayındaki bu değişimlerin izlenmesi ve yönetilmesine odaklanan araştırma alanı bağlı veri dinamikleri olarak adlandırılmaktadır (Käfer et al., 2013; Umbrich et al., 2010).

Tezin sonraki bölümünde Bağlı Veri dinamiklerinin yönetilmesi ile ilgili çalışmalar incelenecektir.

2.2.4 Bağlı Veri dinamiklerinin yönetilmesi

Bağlı Veri üzerindeki değişimler hakkındaki çalışmalar değişiklikleri üç ayrıntı düzeyinde incelemektedir (Umbrich et al., 2010):

- veriseti düzeyi: verisetinde meydana gelen her türlü değişikliği kapsar.
- kaynak düzeyi: bir kaynağın URI'sinin değiştiği durumları kapsar.
- deyim düzeyi: bir üçlünün silindiği ya da eklendiği durumları kapsar.

Bağlı Veri dinamikleri çalışmaları ise dört ana başlık altında toplanmaktadır (Umbrich et al., 2010):

1. değişik ayrıntı düzeyinde ("granularity") ifade edilen veriseti değişiklikleri için veri tüketicileri ve üreticileri arasındaki etkileşimi sağlayacak veb ölçeğinde çalışabilecek iletişim metotları
2. veriseti özelliklerinin ve değişim tanımlamalarının kendiliğinden keşfi ("auto discovery")
3. verisetlerinin dinamik özelliklerinin ve üzerlerinde meydana gelen değişikliklerin ifade edilmesinde kullanılan sözlükler ("vocabulary")
4. bir verisetinin iki değişik sürümü arasındaki farkı verimli şekilde hesaplayabilecek algoritmalar.

Veriseti Özellikleri ve Değişikliklerinin İletilmesi için Geliştirilen Sistemler

Veriseti özelliklerinin ve değişikliklerinin iletilmesi için geliştirilen sistemler kullandıkları yaklaşıma göre isteme ("pull") ve itme ("push") tabanlı olmak üzere iki kategoride incelenmektedir. İsteme tabanlı yaklaşımda, bağlı veriyi kullanan uygulamalar periyodik olarak veri kaynağında değişiklik olup olmadığını kontrol etmektedirler. İtme tabanlı yaklaşımda ise veri kaynakları içerdikleri ve izlenen öğelerinde bir değişiklik olduğu anda ilgili öğeleri izleyen uygulamaları bilgilendirmektedir. Bunun için, uygulamaların ilgili izleme seviyesinde veri kaynağı tarafına üye olmaları gerekmektedir. İtme tabanlı mekanizmalar, periyodik olarak kontrol gerektirmediği için sıklıkla değişen verilerin izlenmesi için daha etkin bir yöntem olarak görülmelerine karşın üye listelerinin yönetimi, bildirimde

kullanılacak ağ iletişim altyapısı performansı ve ölçeklenebilirlik gibi dezavantajları da bulunmaktadır. İsteme tabanlı yaklaşım ise istemcilerin sürekli değişiklik olup olmadığını kontrol etmelerini gerektirdiğinden istemciler için yük oluşturmaktadır. Eğer bir sistem istemcilerinin hakkında haberdar edilmek için kayıtlandıkları değişimler gerçekleştiğinde uyarırsa bu sistemlere isteme ve itme tabanlı sistem adı verilmektedir. Tezin bu bölümünde veriseti dinamiklerinin iletilmesi için geliştirilmiş sistemler incelenecektir.

sparqlPUSH: "sparqlPuSH" sistemi (Passant and Mendes, 2010), gerçek zamanlı veri akışı ("stream") için geliştirilmiş olan PubSubHubbub (Fitzpatrick et al., 2010) isimli merkezi olmayan ("decentralized") gerçek zamanlı ("real time") isteme ve itme tabanlı ağ protokolünün kullanımına dayanan sorgu düzeyinde çalışan itme tabanlı bir sistemdir.

SparqlPush sisteminde kullanıcılar bir RDF deposu üzerinde takip etmek istedikleri SPARQL sorgusunu sistem arayüzü üzerinden kayıt ettirmektedirler. Daha sonra, sistem arayüzü bir besleme ("feed") oluşturarak bu beslemeyi bir PuSH dağıtım merkezine ("hub") kayıt ettirmektedir. Dağıtım merkezine örnek olarak Google'ın herkese açık PuSH dağıtım merkezi verilebilir. Bu işlemden sonra, dağıtım merkezi adresine ilişkin bağlantıyı içeren besleme kullanıcıya geri döndürülmektedir. Kullanıcı da gelen beslemeyi çözümleyerek bu dağıtım merkezindeki ilgili beslemeye olan ilgisini dağıtım merkezine kayıt ettirmektedir. Veri kaynağı tarafında, RDF deposunda bir değişiklik olduğu zaman kayıtlı SPARQL sorguları işletilmekte, sonuçları değişen sorgulara ait beslemeler güncellenmekte ve dağıtım merkezine bir bildirim gönderilmektedir. Dağıtım merkezi de değişen bilgileri ilgili beslemeler kapsamında üye olan tüm kullanıcılara dağıtmaktadır.

sparqlPUSH sisteminin veri yayınlayan RDF kaynaklarının dağıtım merkezine yeni veri hakkında bildirim göndermesini zorunlu tutması ve takip ettiği sorguların sadece tek bir RDF deposunu hedeflemesi dezavantaj yaratmaktadır.

DSNotify: "DSNotify" sistemi (Popitsch and Haslhofer, 2011) hem kaynak hem de sorgu düzeyinde çalışabilen hem itme hem de isteme tabanlı bir sistemdir. Sistem bağlı veri uzayı üzerinde iki farklı yöntemle gözleme alanları oluşturabilmektedir. İlk yöntem, başlangıç URI'si ile bir düzenli ifade verildikten sonra ilk URI'den başlayarak düzenli ifadenin sağladığı hiçbir URI kalmayana dek bulunan URI'lerden oluşan RDF çizgesini gözleme alanı olarak tanımlamaktadır. Diğer yöntem ise bir SPARQL sorgusunun tanımladığı RDF çizgesinin izleme alanı olarak tanımlanmasıdır. İki yöntemden birisi tercih edilerek tanımlanan gözleme

alanı izlenmekte ve belirlenen değişimler sistem tarafından kayıt edilmektedir. Bu değişimler daha sonra ya kullanıcıların http istekleri ile sistemi sorgulamasına dayanan isteme yaklaşımı ile ya da kullanıcıların üyelik esasına dayalı XML tabanlı uzak metot çağrımı (XML-RPC) kullanarak gerçekleştirilen itme yaklaşımı ile bildirilebilmektedir.

DSNotify sistemi sparqlPUSH sistemindeki gibi veri yayıncılarının işbirliğini gerektirmese de sparqlPUSH sistemine benzer şekilde sadece tek bir SPARQL uçnoktasındaki sorguları takip edebilmekte, federe sorguları işletememektedir.

SDShare: SDShare (Moore and Marius, 2015) bir verisetindeki değişikliklerin kaynak düzeyindeki tanımlamalarının yayınlanması için kullanılabilecek isteme tabanlı bir protokoldür . SDShare protokolünde bir verisetinin anlık görüntülerinin ve kaynak düzeyindeki değişimlerin REST ("Representational State Transfer") servisleri ile nasıl yayınlanabileceğini tanımlar.

SDShare protokolünü kullanan veri sunucuları sahip oldukları sahip oldukları veri setlerini en az bir veri seti içeren kolleksiyonlar halinde tanımlalar. Bir sunucu istemcilerine her kolleksiyonla ilgili olarak dört ATOM beslemesi sağlar: sunucunun hakkında veri sağladığı kaynakların bulunduğu kolleksiyonların listesini içeren tanıtım ("overview") beslemesi, her kolleksiyonla ilgili olarak bir bellek enstantanesi ("snapshot") ve bölümleme ("fragment") beslemesi bağlantılarını içeren kolleksiyon beslemesi, her kolleksiyon için ilgili kolleksiyonun o anki durumunu ("state") ifade eden bellek kopyası beslemesi, her maddesi ("entry") kendisi ile ilişkili olan bir kaynağın ("resource") değişimi hakkında bilgi içeren bölümleme beslemesi.

SDShare protokolünü arşiv (Garshol and Borge, 2013) ya da konu haritası ("topic maps") yönetimi amacı ile kullanan sistemler (Gießmann et al., 2009) olsa da Bağlı Veri dinamiklerinin yönetimini SDShare protokolü'nü kullanarak gerçekleştirdiği bilinen bir sistem yoktur.

OAI-PMH (Open Archives Protocol for Matadata Harvesting): OAI-PMH (Haslhofer and Schandl, 2010) sayısal kütüphane depolarının aralarında HTTP üzerinden bibliyografik üst veri değiştokuşu yapabilmelerine için geliştirilmiş isteme tabanlı bir sistemdir. Geliştirilen eklentilerle sistemin RDF formatında verileri kabul ederek OAI-PMH sisteminin yerel XML tabanlı üst veri yapısına uygun şekilde kaydetmesi ve yerel üst verinin RDF formatında yayınlaması olanaklı kılınmıştır. Sistem kendisine gelen istemler dahilinde belirli tarihler arasında değişen, yaratılan ya da silinen üst veri kayıtlarına erişim sağlayarak üst veri

kümelerindeki değişikliklerin takibini olanaklı kılsa da genel kullanım amacı taşımadığından kullanımı kısıtlı kalmaktadır.

Ping The Semantic Web: Ping the Semantic Web (PTSW), yeni bir RDF dokümanı yaratıldığında ya da bir RDF dokümanı değiştiğinde PTSW'nin dokümanın yazarı tarafından dokümanın URL adresi ile uyarılması esasına dayanan bir kayıtçı ("registry") servisidir (Heath et al., 2008; Umbrich et al., 2010). PTSW kendisine hakkında uyarı gelen RDF dokümanlarının adreslerini bir liste olarak tutar ve bir web servisi aracılığı ile istemcilerine sunar.

RDFSync: RDFSync iki RDF deposunun farklarını alarak istemcinin olduğu RDF deposuna farkı oluşturan minimal çizgeleri ileten senkronizasyon amaçlı isteme tabanlı bir sistemdir (Tummarello et al., 2007). Bu amaçla RDFSync algoritması bir RDF deposunda bulunan çizgeleri Minimum Bağımsız Çizge ("Minimal Self Contained Graph"-MSG) adı verilen minimal üçlü altkümelerine ayırır.

Bir Minimal Bağımsız Çizge s adı verilen bir üçlüyü ve özyinelemeli olarak bu üçlünün nesnesi ya da öznesi olan boş düğümler ve bu düğümler içeren üçlülerin Minimal Bağımsız Çizgeleri'nden oluşur ve bağımlı olduğu ifade ile MSG(s) şeklinde ifade edilir. RDFSync kendisine girdi olarak verilen her RDF çizgesini Minimal Bağımsız Çizge'lere ayırır ve bu çizgelerin eşleme değerlerini ("hash") hesaplar. Bu değerler kullanılarak iki RDF deposundaki çizgelerin farkları alarak istemci tarafa kendisinde bulunmayan hash değerine sahip çizgeleri aktarır.

WebHooks: WebHooks HTTP geri arama ("callback") olaylarını kullanarak kendisinde kayıtlı URL'lere veri gönderen genel amaçlı itme tabanlı bir sistemdir. Kayıtlı olan URL'lerde gelen veriyi kabul edecek bir sunucunun çalışması gerekmektedir (Umbrich et al., 2010; Trifa et al., 2010).

Semantic Pingback: Semantic Pingback (Sebastian Tramp et al., 2010) Sosyal Veb üzerindeki Pingback (Langridge, 2002) sisteminin Bağlı Veri Ağı'na uyarlanması esasına dayanan itme tabanlı bir sistemdir. Semantic Pingback sisteminde, üzerinde Semantic Pingback sunucusu çalışan bir bağlı veri kaynağına bağ tanımlandığında bağı tanımlayan Semantic Pingback istemcisinin bulunduğu bağlı veri kaynağı sunucuya oluşturulan yeni bağı bildirir. Her iki tarafta da Semantic Pingback sistemi bulunduğundan uyarılar karşılıklı yapılabilmektedir.

Veriseti Özellikleri ve Değişikliklerinin İfade Edilmesinde Kullanılan Sözlükler

DaDy (Dataset Dynamics Vocabulary): voiD için geliştirilmiş bir eklentidir(Hausenblas, 2010). DaDy ile verisetlerinin değişimlerinin düzeni (düzenli ya da düzensiz şeklinde ifade edilmektedir) ve sıklığı (sabit, düşük sıklıkta, orta sıklıkta ya da yüksek sıklıkta şeklinde ifade edilmektedir) tanımlanabilmekte ve değişiklikler konusunda uyarıları yayınlayan kaynağın URI'si tanımlanabilmektedir.

DSNotify Eventset Sözlüğü: DSNotify Eventset Vocabulary (Popitsch and Haslhofer, 2011; Umbrich et al., 2010) DSNotify sistemi tarafından kullanılan kaynak düzeyinde değişim tanımlama sözlüğüdür. DSNotify EventSet sözlüğü kullanılarak bir verisetindeki kaynaklar üzerinde yapılan değişiklikler zamana göre sıralanarak bir EventSet içinde ifade edilir. Her EventSet verisetinin ilk ve sonraki durumunu temsil eden kaynak ve hedef verisetleri ile ilişkilidir. Bir EventSet içinde ifade edilebilen değişiklikler ise kaynak verisetindeki bir kaynağın silinmesi, yaratılması, değiştirilmesi ya da farklı bir URI ile ifade edilmesi ile sonuçlanan taşıma ("move") olaylarıdır. Her olay tanımına etkilediği üçlülerin listesi de eklenebilmektedir.

Talis Changeset Sözlüğü: Kaynak düzeyindeki RDF çizgesi değişimlerini tanımlayan bir sözlüktür (Shinavier, 2010; Tunnicliffe and Davis, 2015). Bu sözlük de kaynak düzeyindeki değişim olaylarını DSNotify Eventset sözlüğüne benzer şekilde bir ChangeSet içinde ifade etmektedir. ChangeSet sözlüğünde bir RDF çizgesinin iki versiyonunda farklı tanımlanmış olan bir kaynağın tanımlamaları arasındaki fark silinen ve eklenen üçlüler şeklinde tanımlanır. Her ChangeSet bir kaynağın iki tanımlaması arasındaki farkı içerir.

The Graph Update Ontology (GUO): RDF çizgelerinde yapılan güncellemelerin ifade edilmesinde kullanılan üçlü düzeyinde bir ontolojidir (Shinavier, 2010; Dabrowski et al., 2011). Güncellemeler ekleme ve çıkarma şeklinde sınıflandırılmış bir RDF kaynağı olarak ifade edilmektedir. Değişikliği temsil eden RDF kaynağına eklenen ya da çıkarılan üçlünün özne, yüklem ve nesnesi "rdf:subject", "rdf:predicate" ve "rdf:object" yüklemleri kullanılarak bağlanmakta böylelikle değişikliğe neden olan üçlü tanımlanmaktadır.

SPARQL 1.1 Update: RDF çizgelerinin güncellenmesinde kullanılan üçlü düzeyinde bir dildir (Gearon et al., 2013) . W3C tarafından verilen bir öneri olup standartlaşması hedeflenmektedir.

Delta: İki RDF çizgesi arasındaki farkın eklenen ve çıkarılan üçlüler cinsinden ifade edilmesini olanaklı kılan bir ontolojidir (Berners-Lee and Connolly, 2004).

CHAO: CHAO (Change and Annotation Ontology) (Noy et al., 2006) Protege ontoloji geliştirme ortamının işbirlikli ontoloji geliştirilmesini olanaklı kılan "Collaborative Protege" sürümünde (Tudorache et al., 2008; Tudorache, 2015) değişimlerin takip edilmesinde kullanılan PROMPTDiff aracı ile birlikte kullanılmak üzere geliştirilmiştir. İki ontoloji versiyonunun yapısal farkının alındığı durumlarda farkın ifade edilmesinde de kullanılabilmektedir. CHAO ontolojisinde iki ana sınıf bulunmaktadır: Change ve Annotation. Bu iki sınıf birbirine iki ters özellik ile bağlıdır. Bir ontoloji üzerinde yapılan değişiklikler Change sınıfının alt sınıfları ile daha detaylı şekilde ifade edilebilmektedir. Bu altsınıflara örnek olarak sınıf eklenmesi, çıkarılması gibi değişiklikler için yaratılmış olan "ClassChange", birden fazla atomik değişiklikten oluşan değişiklikler için yaratılmış "CompositeChange", ontolojiye bilgi eklenmesi ile ilgili değişikliklerin ifadesi için yaratılmış "Created_Change", ontolojiden bilgi çıkarılması ile ilgili değişikliklerin ifade edilmesi için yaratılmış "Deleted_Change" ve özelliklerin eklenmesi ve çıkarılması ile ilgili değişikliklerin ifade edilmesi için yaratılmış olan "Property_Change" verilebilir. Bu sınıfların da daha detaylı tanımlamaları olanaklı kılan alt sınıfları bulunmaktadır.

Sesame RDF Transactions: Sesame 2.0 RDF çatı ("framework") kullanılan üçlü düzeyinde XML tabanlı bir RDF güncelleme formatıdır (Shinavier, 2010) .

OWL 2 Change Ontology (OWL için Değişim Ontolojisi -ODO): Ontoloji değişimlerinin ifade edilmesi için kullanılan ODO bağlı veri değişimlerinin ifade edilmesi için de kullanılabilmektedir (Umbrich et al., 2010). Daha detaylı bilgi için bu ontolojinin incelendiği "Ontoloji Değişikliklerinin İfade Edilmesi İçin Geliştirilmiş Ontolojiler" başlıklı bölüme bakılabilir.

3 ETMENLER VE ETMEN ORTAMLARI

Anlamsal Veb teknolojisi üç temel bileşen üzerine kurulmuştur: etmenler, ontolojiler ve Veb servisleri (Berners-Lee et al., 2001). Bu bileşenlerden etmenler ontolojilerde mevcut olan anlamsal bilginin kullanıcılarıdır. Dolayısıyla etmenler ve ontolojiler arasında sıkı bir bağ bulunmaktadır.

Etmen kavramı literatürde değişik şekillerde tanımlanmaktadır. Russel ve Norvig'e göre (Russell and Norvig, 2003) algılayıcıları yardımıyla ortamı algılayan ve etkileyicileri yardımıyla bu ortamı etkileyen her sistem etmen olarak tanımlanabilir. Hayes-Roth'a göre (Hayes-Roth, 1995) etmenler ortamdaki dinamik değişimleri algılayan, ortamı etkileyen eylemlerde bulunan ve algılarını yorumlayarak amaçları doğrultusunda yapılması gereken eylemleri belirlemek için akıl yürüten sistemlerdir. Wooldridge ve Jennings'e göre (Wooldridge and Jennings, 1995) etmenler kullanıcıların doğrudan katılımı olmadan belli bir derecede otonomluk çerçevesinde çalışan, kullanıcılarla ve diğer etmenlerle iletişimde bulunan, ortamı algılayıp ortamdaki değişimlere karşı eylemde bulunan ve eylemlerini belli amaçlara ulaşabilmek için gerçekleştiren donanım veya genellikle yazılım tabanlı sistemlerdir. Genesereth ve Ketchpel'e göre (Genesereth and Ketchpel, 1994) bir sistem ancak ve ancak bir etmen iletişim dili ile iletişimde bulunabiliyorsa etmen olarak sınıflandırılabilir. Verilen tanımlardan da anlaşılacağı üzere üzerinde anlaşılmış bir tanım olmamasına karşın her etmende bulunması gereken bazı temel özellikler vardır. Bu özellikler birincil özellikler olarak adlandırılmakta ve etmenleri klasik donanım ve yazılım sistemlerinden ayırmaktadır. Etmenlerin birincil özellikler şunlardır (Wooldridge and Jennings, 1995):

- Otonomluk ("autonomy"): Otonomluk bir etmenin insan kullanıcıların, diğer etmenlerin ya da sistemlerin doğrudan katılımı olmadan görev başlatabilme ve çalıştırabilirle yeteneği olarak tanımlanmaktadır. Otonom bir etmen kendi eylemleri ve içsel durumunu kontrol edebilmelidir.

- Karşıt-eylemlilik (tepkisellik-"reactivity"): Bir etmen sürekli olarak bulunduğu ortamı algılamalı, ortamdaki değişimlere göre bilgisini, amaçlarım, eylemlerin değiştirebilmelidir.

- Sosyal yetenek ("social ability"): Bir etmen, planlarına ilişkin görevlerini tamamlayabilmek için genelde diğer etmenler ya da insan kullanıcılarla etkileşmek durumunda kalmaktadır. Bu etkileşim ancak bir etmenler arası iletişim dili ile sağlanabilir.

- Kalıcı süreklilik (“temporal continuity”): Bir etmen belli bir görevi yapıp durmamalıdır. Etmenler bir ortamda sürekli çalışan birimlerdir. Etmen, başlattığı görevi tamamladıktan sonra da etkin olarak kalmalı ve çevresini algılayıp gereken eylemleri gerçekleştirmek üzere hazır olarak beklemelidir. Etmenlerin temel özelliklerine bakarak bir etmenin ortamla sürekli etkileşim halinde olacağı ve her seyi bilmese (“omniscient”) de sağlıklı işleyişini devam ettirebilmek adına ortam hakkında yeterli bilgiye sahip olması gerektiği söylenebilir.

Literatürde ortam kavramı da değişik şekillerde tanımlanmıştır.

Russell ve Norvig’e göre ortam genel bir ortam programı ile modellenebilir (Russell and Norvig, 2003). Bu basit program etmenlere algı sağlamakta ve karşılığında onların tepkilerini toplamaktadır. Program daha sonra etmenlerin tepkilerine ve ortamda bulunan etmenlerin dışında kalan diğer dinamik süreçlere göre ortamın durumunu güncellemektedir.

Rao ve arkadaşlarına göre (Rao and Georgeff, 1992) ortam etmen sisteminin kullanılacağı alana göre şu özelliklere sahip olacaktır:

1. herhangi bir anda ortam bir çok farklı şekilde değişebilir,
2. herhangi bir anda olası bir çok eylem vardır,
3. farklı hedeflere aynı anda ulaşılamayabilir,
4. değişik hedeflere varılmasını sağlayacak eylemler ortamın durumuna bağlıdır,
5. ortam yerel olarak hissedilebilir,
6. eylemlerin ve hesaplamaların gerçekleştirilme süresi ortamın değişim süresine göre makul uzunlukta olmalıdır.

Parunak (Parunak, 1997) ortamı <Durum, Süreç> şeklinde tanımlanmış bir değişkenler uzayı (“tuple space”) olarak tanımlar. Durum, ortamı etmenler ve ortamdaki nesneler de dahil olacak şekilde tamamen tanımlayan bir değerler kümesidir. Süreç ortamın kendisinin dinamik bir varlık olduğunu ifade etmektedir. Ortam içindeki etmenlerin eylemlerinden bağımsız olarak kendi durumunu değiştirebilecek bir sürece sahiptir. Sürecin temel amacı ortama dinamizm katabilmektir.

Demazeau (Demazeau, 1999)ise etmen sistemleri için dört temel inşa bloku belirlemiştir: etmenler, etkileşimler, organizasyonlar ve etmenler arasındaki dış etkileşimleri yapılandırmak için gerekli olan alan bağımlı (“domain dependent”) bileşenler olarak tanımlanmış olan ortam.

Odell ve arkadaşları (Odell et al., 2003) ise ortamın bir varlığın (etmen ya da nesne) varolduğu şartları sağladığını belirtmektedir. Yazarlar fiziksel ortamı ve iletişim ortamını ayırmaktadır. Fiziksel ortam etmenlerin ve nesnelerin fiziksel varoluşlarını destekleyen ve yöneten kanunları, kuralları, kısıtları ve politikaları sağlamaktadır. İletişim ortamı ise fikirler ve bilginin değiştokuşunu destekleyen ve yöneten süreç ve prensipler ile roller, gruplar ve etkileşim protokolleri gibi iletişim için sıklıkla kullanılan fonksiyon ve yapılan sağlar.

FIPA’ya göre (J. et al., 2003) ise ortam olmadan etmenler yararsızdır. Dünyanın geri kalanından ayrılan bir etmen algılayamaz ve eyleme geçemez. Ortam, etmen ya da nesne, tüm varlıkların var olabilmeleri için gerekli olan koşullan sağlar, etmenin çalışacağı dünyanın özelliklerini tanımlar. Etkili etmenler için ortamın hem fiziksel hem de iletişimsel yönlerinin dikkatle düşünülmesi gerekir, Zira etmenler ortam aracılığıyla etkileşebilirler. Ortam sadece etmen dışı varlıklardan değil, etmenlerin var olabilmelerini ve iletişim kurabilmelerini sağlayan süreç ve prensipleri de içerir.

Weyns ve arkadaşlarına (Weyns et al., 2007) göre ise ortam etmenlerin varoluşları için gerekli koşulları sağlayan, etmenler arası etkileşime ve kaynaklara erişime aracılık eder.

Yukarıda verilen tanımlamalar ortamı farklı açılardan ele almaktadır. Ortamın tüm bakış açılarından öne çıkan özellikleri derlendiğinde karşımıza daha net bir ortam kavramı çıkmaktadır. Buna göre:

- Ortam etmenlerin var olmaları için gerekli olan fiziksel koşulları ve etkileşimsel aracılığı sağlar.
- Ortam etmenlerin arasında bilgi kaynaklarının da olduğu kaynaklara erişimine aracılık eder.
- Ortam etmenlerin karşıt eylemliliğini destekleyecek şekilde etmenlere algı sağlar, barındırdığı etmen dışı aktif süreçler ve etmenlerin tepkileri ile sürekli değişir.
- Ortam etmenlerin alanına göre alan bağımlı bileşenler içerir.

3.1 İçkökenli ve Dışkökenli Ortamlar

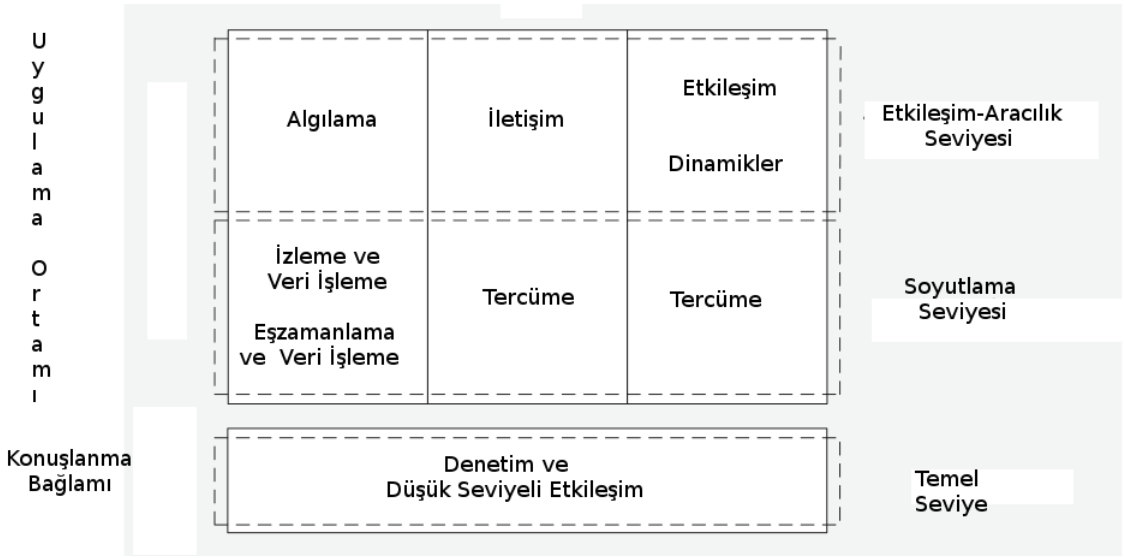
Çok-etmenli sistemlerde ortam bileşeninin gerçekleştirimine dışkökenli (“exogeneous”) (Ricci et al., 2011)ve içkökenli (“endogeneous”) olmak üzere iki farklı yaklaşım vardır. Dışkökenli yaklaşımın kökeni yapay zeka araştırmalarına dayanmaktadır. Bu yaklaşıma göre ortam sistemin dışında kalan ve etmenin görevlerini yerine getirmek için algıladığı ve etkileştiği dünyadır (Russell and Norvig, 2003) Çok-etmenli sistemlerin mühendisliğinden gelen içkökenli (Ricci et al., 2011) ortam yaklaşımında ise ortam etmenlerle aynı düzeyde olan birinci sınıf bir sistem bileşeni olarak tanımlanmaktadır (Weyns et al., 2007). Birinci sınıf bileşenler sağladıkları soyutlama ve bilgi saklama mekanizmaları nedeni ile gerçekleştirimi diğer modüllerde herhangi bir değişiklik gerektirmeden değiştirilebilecek bağımsız yazılım parçaları ya da yazılım inşa bloklarıdır (Odell et al., 2003). İçkökenli yaklaşımda ortam, etmenin aktivitelerini desteklemek amacıyla gerekli fonksiyon, servis ve veri gibi kaynakları bünyesinde barındıran, çok-etmenli sistemin bir parçası olarak görülen, etmenler tarafından doğrudan erişilebilen ve çok-etmenli sistem ile bütünleşik bir şekilde tasarlanıp, programlanabilen bir bileşen olarak tanımlanmaktadır.

3.1.1 İçkökenli ortamlar

Ortamın içkökenli bakış açısı ile modellenmesi sağladığı soyutlama nedeniyle karmaşık ÇES uygulamalarının tasarlanmasını ve gerçeklenmesini kolaylaştırmaktadır ve etmenler gibi otonom varlıklarla modellenmemesi gereken fonksiyonel bileşenlerin (iletişim altyapısı gibi) modellenmesi için kavramsal olarak uygundur (Odell et al., 2003). Dolayısıyla içkökenli ortamlar dışkökenli ortamlara göre gerçekleştirim ve ölçeklenebilirlik açısından bazı avantajlar sağlamaktadır.

İçkökenli ortamlar etmen sistemlerine Şekil 3.1’de de görülebilen üç seviyede destek verir (Weyns et al., 2007):

- **Temel seviye:** Ortamın temel görevi etmenlerin içinde bulundukları konuşlanma bağlamına (“deployment context”) yani ÇES’nin etkileştiği donanım, yazılım ve veritabanları, Veb servisleri, vb. gibi dış kaynaklara erişmelerini sağlamaktır.
- **Soyutlama seviyesi:** Soyutlama seviyesi etmenler ile konuşlanma bağlamının düşük seviyeli detaylarını ve diğer olası kaynakları arasında köprü



Şekil 3.1: İçkökenli ortam mimarisi referans modeli (Weyns'den, 2007)

görevi gören bir soyutlama sağlar.

- **Etkileşim-aracılık seviyesi:** Bu seviye paylaşılan kaynaklara erişimi düzenler ve etmenler arası iletişime aracılık eder. Etkileşim-aracılık seviyesi sayesinde ortam ÇES içinde aktif bir varlık haline gelir. Etkileşim aracılığı sayesinde etmenler ortamı kullanarak davranışlarını koordine edebilirler.

İçkökenli ortamların temel sorumlulukları ise şunlardır (Weyns et al., 2007; Odell et al., 2003; Weyns and Holvoet, 2005; Weyns et al., 2005) :

- **Ortam ÇES'ni yapılandırır:** Ortam etmenler, kaynaklar ve servislerin yer aldığı ortak uzaydır ve sistemi yapılandırır. Burada kaynakların etmenler tarafından da gözlenip kullanılabilen özel durumları vardır ve servisler de etmenlere fonksiyonellik sağlayan tepkisel varlıklar olarak düşünülürler. Etmenler, kaynaklar ve servisler birbirleriyle dinamik olarak ilişkilidirler. Bu ilişkilerin uyması gereken kuralları tanımlamak ortamın sorumluluğudur. Aslında, ortam ÇES'ni değişik yönlerden şekillendirir. Bunlara örnek olarak sistemin topolojisini ve dağıtıklığını da içeren fiziksel ve uzamsal yapı, örtülü ("implicit") iletişime ve mesaj iletimi yolu ile açık iletişime destek olan iletişimsel yapılar ile roller, gruplar, topluluklar vb. birimlerin kullanılması ile oluşturulmuş organizasyonel yapı verilebilir.

- **Ortam servisleri ve kaynakları içerir:** Kaynaklar ve servisler tipik olarak fiziksel bir yapı içinde yerleşmişlerdir. Ortam kaynakların ve servislerin düşük düzeyli detaylarını soyutlama düzeyinde destek vererek etmenlerden

saklamalıdır. Ortamın etmenlerin eylemleri haricinde, etmenlerden bağımsız olarak kendisine ait süreçleri diğer bir deyişle dinamikleri olabilir.

- **Ortam etmenler tarafından yerel olarak gözlenebilir:** Etmenlerin tersine, ortamın mutlaka gözlenebilir olması gerekmektedir. Etmenler ortamın değişik yapılarını, kaynaklarını, servislerini ve muhtemelen diğer etmenlerin dış durumlarını gözleyebilmelidir. Bir yapının gözlenmesi tipik olarak etmenin içinde bulunduğu bağlamla sınırlıdır. Genel olarak, etmenler ortamı mevcut görevlerine göre denetlemelidirler.

  Ortamın bir ortam ontolojisi ile tanımlanabilecek olan anlamsal tanımı da gözlenebilirlikle bağlantılıdır. Ontoloji ortamın değişik yapılarını, kaynakların, servislerin ve etmenlerin gözlenebilir özelliklerini, aralarındaki ilişkileri ve muhtemel olarak düzenleyici kuralları içermelidir. Sembolik temelli etmenler için ortamı yorumlayabilmeleri ve ortam hakkında çıkarsama yapabilmeleri için açık ("explicit") bir ontoloji tanımlanmalıdır.

- **Ortam etmenler tarafından yerel olarak erişilebilir olmalıdır:** Etmenler ortamın değişik yapılarına, kaynaklarına, servislerine erişebilmelidir. İletişim altyapısına erişim doğrudan iletişim (mesaj transferi), doğrudan olmayan ve örtük iletişim için destek sağlanmasını ifade eder. Sosyal yapılara erişim organizasyonlar, grup üyelikleri ve diğer normatif sosyal yapılar gibi sosyal birimlerle etkileşebilme yeteneğini ifade etmektedir.

- **Ortam ÇES'ndeki tüm varlıklar için değişik tipte kurallar tanımlayabilir:** Kurallar eldeki problem alanı ile ilgili kısıtlar ya da tasarımcı tarafından konulan kuralları ifade edebilir. Kurallar belli tipteki etmenlerin bazı kaynaklara ya da servislere erişimini kısıtlayabilir veya etmen etkileşimlerinin sonuçlarını belirleyebilir. Ortam etmenlerin aktiviteleri hakkında kurallar koyarak ÇES'ini uygulama alanının özellik ve gereksinimleriyle uyumlu olacak şekilde istikrarlı tutan bir hakem olarak görev yapar.

- **Ortam iletişimi olanaklı kılar:** Ortam etmenlerin iletişim için somut yollar tanımlar. İletişim değişik şekiller alabilir. En çok kullanılan yöntem bir etmenden diğerine mesaj gönderilmesidir ("message passing"). Üretken ("generative") ya da doğrudan olmayan ("indirect") iletişimde ise etmenler ortamda iletişim nesneleri üretirler ve bunları okuma amacıyla tüketirler.

Ortam genellikle ÇES'nin üzerinde çalıştığı konuşlanma bağlamı ile karıştırılmaktadır. Dolayısıyla ortamın fonksiyonları ÇES'nin içine uygulamaya özel çözümlerle ya da örtük şekilde entegre edilmektedir. Bu karışıklığın

giderilmesi adına içkökenli ÇES ortamları için üç katmanlı bir mimari önerilmiştir (Weyns et al., 2005):

- **ÇES uygulama katmanı:** Bu katman ÇES çatısını, ÇES uygulamasını ve uygulamanın içinde çalıştığı ortamı içermektedir. Bu bileşenler eldeki problem için getirilmiş olan çözümü temsil eder. Uygulama ortamı, etmenler için gerekli olan uygulamaya özel alan gösterimini sağlar. Uygulama ortamı, uygulama etmenlerinin uygulama alanı kaynakları ve diğer etmenlerle etkileşimini sağlamak amacıyla etmenlere uygulamaya özgü kaynakların karmaşıklığını, etkileşim ve tutarlılık yönetimini gizleyen bir soyutlama sunar.
- **Çalıştırma platformu katmanı:** Bu katman "middleware" ve işletim sistemi alt-katmanlarından oluşur. "Middleware" katmanı dağıtık bileşenleri bir arada tutan bir yapıştırıcı görevi yapar. Uzak prosedür çağırmaları, iş parçacığı çalıştırma, hareketler ("transaction"), kalıcılık ("persistence"), yük dengeleme, vs. ile ilgili destek verir. İşletim sistemi katmanı ise uygulama yazılımlarının donanım üzerinde çalışabilmesini olanaklı kılar.
- **Fiziksel altyapı:** Bilgisayar donanımı ve ÇES'nin fiziksel parçalarına karşılık gelen fiziksel dünya olmak üzere ikiye ayrılır.

İçkökenli ortamlar için önerilen mimari katmanlar ve sorumluluklar dikkate alındığında bir uygulama ortamının modeli şu bileşenleri içermelidir (Weyns et al., 2007):

- **Durum:** Durum modülü uygulama ortamının güncel durumunu temsil eder. Ortamın durumu tipik olarak konuşlanma bağlamının bir soyutlamasını (örneğin ağ topolojileri, fiziksel ortam haritası, vb.) ve muhtemelen ÇES ortamıyla ilişkili diğer durum tanımlamalarını içerir. Ortam durumu etmenlere özel bilgiler de içerebilir.
- **Senkronizasyon ve veri işleme:** Senkronizasyon ve veri işleme modülü konuşlanma ortamının alana özel parçalarını izler ve durum modülündeki ilgili gösterimi güncel tutar. Bu modül sadece elde ettiği bilgiyi yansıtmaz, aynı zamanda kullanılan sensörlerin ve bu sensörlerdeki verilerin sisteme uygun hale getirilmesi (örneğin, veri sıralama, sensör kalibrasyonu, veri düzeltme, vs.) ile de ilgilenir.
- **Dinamikler:** Bu modül etmenlerden bağımsız olarak gerçekleşen ortam dinamiklerinin devamlılığından sorumludur.

- **Kurallar:** Kurallar modülü etmenlerin ortam içindeki etkileşimleri üzerinde uygulamaya özel olarak konmuş olan kısıtları temsil eder. Kurallar etmen algısı, etkileşimi ve iletişimi üzerinde kısıtlamalar getirebilir.

- **Algılama:** Bu modül etmenlerin ortamı algılamalan için gerekli fonksiyonlan sağlar. Bir etmen ortamı algıladığında algılama modülü uygulama ortamının güncel durumuna ve muhtemelen konuşlanma bağlamından elde edilen verilere göre algılar üretir. Etmen algısı kurallar tabidir. Algılama kuralları algılamayı kısıtlamak için yollar sağlar.

- **Gözlemleme ve veri işleme:** Bu modül algılama modülüne konuşlanma bağlamını gözlemleyebilmesi için için gerekli fonksiyonları sağlar. Konuşlanma bağlamından elde edilen veriler muhtemelen işlendikten sonra algılama modülüne geçirilirler.

- **Etkileşim:** Etkileşim modülü etmenlerin ortamdaki eylemleriyle ilgilenir. Bu modül bazı eylem komutlarının nasıl çalıştırıldığını tanımlayan bir eylem modeli içerir. Örneğin, etmen ortama bir etkide bulunur ve ortam da ortamın durumunu değiştirerek etkiye tepkiyle karşılık verir. Etmenlerin eylemleri uygulama ortamının durumunu değiştirme amaçlı eylemler ve konuşlanma bağlamının elemanlarını değiştirme amaçlı eylemler olmak üzere ikiye ayrılır ve etkileşim kurallarına tabidirler.

- **İletişim:** İletişim modülü mesajları toplar ve uygun etmenlere iletir. Etkileşim modülüne benzer şekilde, iletişim modülü de etmenler arasında gönderilen mesajları uygulama ortamının mevcut durumuna ve kurallara dayanarak düzenleyebilir.

Platon ve arkadaşları (Platon et al., 2007) ise içkökenli ÇES ortamlarının geliştirilmesi sırasında destek sağlaması amacıyla ortamda bulunması gereken temel mekanizmaları belirlemişlerdir. Bu mekanizmalar ortam tarafından aracılık edilen etkileşim kanalları, senkronizasyon mekanizmaları, bindirmeli ("overlay") ağlar, kaynak ve bağlam yöneticisi, bağlamsal olayların bildirilmesi ve yardımcı veri yapılarıdır.

Önerilen mekanizmalardan kaynak ve bağlam yöneticileri etmenlerin ÇES içindeki kaynaklara ve bağlam bilgilerine erişimini kontrol eder. Bu mekanizma etmenlerin kaynaklara erişimini sağlayan arayüzleri ve bilgi saklamak için gerekli olan depoları içerir. Kaynaklara erişim, veri depolarında veri tutmak ve depoların tutarlılığını korumak için bir protokoller kümesi kullanılır. Bağlam olaylarını bildiren mekanizma ise olay dağıtıcı ("event dispatcher") depolarına gereksinim

duymaktadır. Etmenler protokoller aracılığıyla yeni olay kaynaklarına abone olabilir ve bir olay gerçekleştiğinde bundan haberdar edilebilirler.

3.1.2 İçkökenli ortam modelleri

Etmen ortamları literatüründe içkökenli ortamların modellendiği tek çalışma A&A ("Agents&Artifacts", Etmenler ve Artifaktlar) metamodelidir. A&A metamodeli Aktivite ve Dağıtık Bilişsellik teorilerinden ilham alarak oluşturulmuştur.

A&A metamodelinin temel bileşenlerinden biri artifaktlardır. Artifakt kavramı Aktivite ve Dağıtık Bilişsellik teorilerinden gelmektedir. Dolayısıyla, A&A metamodelinin detaylı şekilde incelenmesinden önce bu teorilerin kısaca incelenmesi faydalı olacaktır.

Aktivite Teorisi

Aktivite Teorisi, psikoloji ve diğer sosyal bilimlerde insanların ve oluşturdukları sosyal varlıkların aktivitelerinin oluşum, yapı ve süreçlerinin doğal günlük yaşam koşulları içinde analiz edilmesi yoluyla anlaşılmasını amaçlayan bir yaklaşımdır (Kaptelinin and Nardi, 2006). Aktivite Teorisi'nde tek bir insanın zihinsel yeterliliklerinin anlaşılması hedeflense de insan izole edilmiş tek bir birey olarak incelenmez. Bireylerin diğer bireylerle, iş ortamıyla ve toplumla olan etkileşimleri de Aktivite Teorisi'nin ilgi alanına girer. Dolayısıyla insan aktivitelerinin sosyal, kültürel ve teknik detayları da bu teorinin içinde incelenir (Wikipedia, a).

Aktivite genelde, sadece insan aktivitesi olarak değil, herhangi bir öznenin dünya ile bir amaç dahilinde etkileştiği, nesne ve özne kutuplarının ayrı ayrı değişime uğradığı bir süreç olarak da tanımlanmaktadır (Kaptelinin and Nardi, 2006). Bir aktivitenin öznesi tek bir birey olabilceği gibi bir çok bireyden oluşan sosyal varlıklar da olabilir (Kaptelinin and Nardi, 2012).

Özneler Dünya'da yaşayan ve sadece Dünya'da var olmaları ve eyleme geçmeleriyle karşılayabilecekleri ihtiyaçları olan varlıklardır. Eylemler özne tarafından ihtiyaçlarını karşılamak amacıyla başlatılan etkileşimlerdir. Özne ve nesne arasındaki etkileşimde nesne öznenin belirli bir ihtiyacını karşıladığından özneye eyleme geçmesi için motivasyon sağlar. Dolayısıyla, aktiviteler nesneler olmadan var olamazlar zira bir öznenin her aktivitesi bir nesneye yöneliktir.

Aktiviteler belli nesneleri motivasyon olarak almalarına karşın tek parçalı değillerdir. Her aktivite üç seviyeden oluşan hiyerarşik bir yapı ile temsil edilebilir. En üst seviyede aktivitenin kendisi vardır. Bir aktivite birçok basamaktan oluşabilir. Bu basamakların doğrudan motivasyona yönelik olmaları gerekmez; ancak basamaklar dizisi sonunda aktivitenin motivasyonuna ulaşılmasını sağlar. Bu basamakların her birine eylem ("action") adı verilir. Eylemlerin itici gücü olan nesneler ise hedeflerdir ("goals"). Hedefler bilinçli olarak seçilir, ancak motivasyonların hemen farkına varılmayabilir (Kaptelinin and Nardi, 2006). Eylemler de daha alt seviye birimler olan operasyonlara bölünebilirler. Operasyonlar sürmekte olan bir eylemin mevcut şartlara uyumunu sağlayan otomatikleşmiş rutin süreçlerdir. Operasyonlar öznenin bir hedefe varmaya çalıştığı sırada altında olduğu koşullara yöneliktir. Genelde özneler operasyonların farkında değillerdir (Kaptelinin and Nardi, 2006).

Aktivite Teorisi'nde vurgulanan bir diğer kavram ise artifaktlardır. Özneler nesnelerle olan etkileşimlerinde artifaktların aracılığından yararlanırlar. Artifaktlar belli bir ihtiyacını gidermeye çalışan öznenin bu amaçla kullandığı araçlar, dokümanlar, yönergeler vb. olabilir.

Aktivite Teorisi'nde önemli olan bir diğer temel kavram ise fonksiyonel organlardır ("functional organs"). Fonksiyonel organlar, iç ve dış kaynakların bireyler tarafından birleştirilmesiyle yaratılır. Fonksiyonel organlar öznelerin doğal yeteneklerini yapay artifaktlarla birleştirir, ve öznelerin başka şekilde erişemeyecekleri hedeflere ulaşmalarını sağlarlar (Kaptelinin and Nardi, 2006), Fonksiyonel organların yaratılması ve kullanılması için bazı özel yetenekler gereklidir. Araç ilişkili yeterlilikler ("tool-related competencies"), aracın fonksiyonları hakkında bilgi ve aracı kullanabilmek için gerekli yeteneklere sahip olmayı gerektirir. Görev ilişkili yeterlilikler bir aracın kullanılmasıyla erişilebilecek yüksek seviyeli hedefler hakkında bilgi sahibi olmayı, ve bu hedeflere ulaşabilmek için aracın hangi fonksiyonlarının nasıl kullanılması gerektiğini planlayabilmeyi gerektirir (Kaptelinin and Nardi, 2006).

Özneler bir fonksiyonel organı ne zaman kullanacaklarına, veya bir fonksiyonel organın ne zaman değiştirilmesi ya da güncellenmesi gerektiğine karar vermelidirler. Dolayısıyla, öznelerin fonksiyonel organları etkin bir şekilde yaratıp kullanmalarını sağlayacak özel bir tür yeterlilikleri olmalıdır. Fonksiyonel organları öznelerin aktivitelerinin içine entegre eden bu yeterlilikler metafonksiyonel olarak adlandırılabilirler (Kaptelinin and Nardi, 2006).

Aktivite teorisi aktivite kavramını iç aktiviteler ve dış aktiviteler olmak

üzere ikiye ayırır. Zihinsel süreçler iç aktivitelere karşılık gelir. İçselleştirme ("internalization") dış aktivitelerin iç aktivitelere dönüştürülmesidir. Örneğin, bir kişi ilk kez klavye kullanmaya başladığında tuşlara bakar, ancak bu zamanla azalır. İç aktivite dış aktiviteden ortaya çıkar, fakat dış aktivitenin bir kopyası değildir. Örneğin, daktilocu zihninde bir klavye görmez. Dış aktivite içselleşirken şekli de değişir. İçselleştirme, öznelerin gerçekle zihinsel benzetimler ve tasarımlar yoluyla, fiziksel nesneler üzerinde manipülasyonlar gerçekleştirmeden etkileşmeleri için bir yol sunar. İçselleştirme aktivitenin iç ve dış bileşenleri arasında tekrar dağıtılmasını sağlar. Dışsallaştırma ("externalization") ise iç aktiviteleri dış aktiviteler haline getirir. Dışsallaştırma genellikle, bir iç eylemin tamir edilmesi ya da ölçeklenmesi gerektiğinde gerekli olur. Zihinsel dört işlem için çok büyük sayılarla işlem yaparken kağıt kalemin kullanılması buna örnek olarak verilebilir. Bazı zurumlarda ise, örneğin birden fazla öznenin işbirliği yapması gerektiğinde, aktiviteler koordine edilebilmeleri için dışsallaştırılırlar (Kaptelinin and Nardi, 2006).

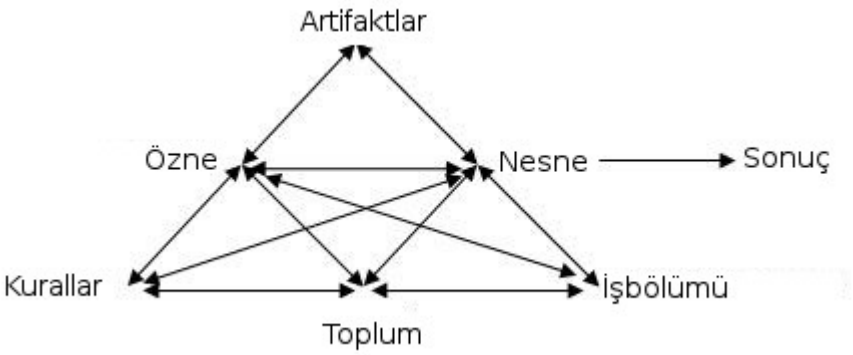
Öznelerin gerçekle olan etkileşimini şekillendiren artifaktlar da içselleştirme prensibi nedeniyle de, dış aktivitelerin şekillenmesi, iç aktivitelerin şekillenmesine neden olur. Dahası, artifaktlar genellikle benzer problemleri daha önce çözmeye çalışmış olan ve bu amaçla bir araç yaratmış, ya da mevcut bir aracı değiştirmiş olan öznelerin deneyimlerini yansıtır. Bu deneyimler artifaktların şekil, materyal gibi yapısal özelliklerinde, ve artifaktın nasıl kullanılmasını gerektiği hakkındaki bilgide birikir. Dolayısıyla artifaktlar dış davranışın doğasını ve öznelerin zihinsel fonksiyonlarını etkiler (Kaptelinin and Nardi, 2006).

Engeström Aktivite teorisinin yukarıda tanımlanan bileşenleri arasındaki ilişkileri Şekil 3.2'de gösterilen modelle ifade etmiştir. İki yönlü oklar, Aktivite teorisinin temel fikri olan etkileşimi temsil eder. Aktivite sisteminin bileşenleri sürekli olarak birbirlerini etkiler ve değiştirirler. Dahası, her sistem bağlantılı aktivite sistemlerindeki bir düğümdür. Örneğin, bir aktivite sistemindeki artifakt bir diğer aktivite sisteminin sonucu olabilir. Dolayısıyla, aktivite sistemlerinin tarihsel gelişim katmanlarından etkilendikleri söylenebilir (Gilbert, 1999).

Dağıtık Bilişsellik Teorisi

Dağıtık Bilişsellik ise 1980'lerin ortasında geliştirilmiş, bilişselliğin sosyal yönünü vurgulayan bir psikoloji teorisidir. İki temel bileşeni vardır:

- bilginin içinde tutulduğu ve dönüştürüldüğü gösterimler ve



Şekil 3.2: Engeström'ün Aktivite Sistemi Modeli

- gösterimlerin birbirleriyle koordine edilmelerini sağlayan süreç.

Dağıtık bilişselliğe göre, bilişsellik ve insan bilgisi bireyle sınırlanmamıştır. Bunun yerine, bilgi ve bilişsellik anıların, gerçeklerin ya da bilginin çevremizdeki nesnelerin, bireylerin ve araçların üzerine yerleştirilmesi ile dağıtılmıştır. Dağıtık bilişsellik bir sistemi, bir gösterimler kümesi olarak görür ve bilginin bu gösterimler arasındaki değiş tokuşunu modeller. Bu gösterimler katılımcıların zihinsel uzayında olabileceği gibi, çevrede bulunan dış gösterimler de olabilir (Wikipedia, b).

Bu teoride bilişsel süreçler üç şekilde dağıtıklaştırılabilir: bir sosyal grubun üyeleri arasında, bilişsel sistemin çalışmasının iç ve dış (maddesel ya da çevresel) bileşenler arasındaki koordinasyona bağlı olacağı şekilde ve önceki olayların sonraki olayların doğasını etkileyeceği şekilde zamana dayalı olarak (Wikipedia, b; Gilbert, 1999). Dağıtık bilişselliğe en basit örnek insanların karmaşık bir aritmetik problemini çözerken kağıt ve kalem kullanmalarıdır. Problemi çözen kişi, problemi netleştirmek için arkadaşı ile konuşabilir, daha sonra problemin çözümünde nerede olduğunu kaybetmemek için kağıdın üzerine kısmi yanıtlar yazabilir. Burada Dağıtık bilişsellik problemin bir başka kişi ile işbirliği ile kurulmasında, hem zihinsel olarak hem de kağıt üzerinde kısmi yanıtlar yazılmasında görülebilir. Yanıtların çıkarılması işlemi sadece iki kişinin algı ve düşüncelerini değil, bir bireyin belleğinin genişletilmesi amacıyla araç olarak bir kağıt parçasının da kullanımını gerektirir. Dolayısıyla, zeka kişiler arasında, ve bir kişi ile bir nesne arasında dağıtılmıştır .

Dağıtık bilişselliğin bilgisayar teknolojisindeki uygulamalarından biri de Anlamsal Web için gerekli üst düzey ontolojilerdir. Ontolojilerin Anlamsal Web üzerindeki kullanımı, bilginin dağıtık gösterimini gerektirmektedir. Bu gösterimler gelecekte Anlamsal Web üzerindeki servislerin otomatik koordinasyonunu sağlayacaktır (Heylighen et al., 2004).

A&A Metamodeli

Aktivite Teorisi'nden ilham alan A&A metamodelinde etmenler öznelere, ÇES'ler ise insan toplumlarına karşılık gelmektedir. Yine Aktivite Teorisi'ne benzer şekilde, A&A metamodelinde de etmenler belli bir amaca ulaşmak için artifaktların aracılık ettikleri ve destek verdikleri eylemleri diğer etmenlerle paylaştıkları çalışma ortamlarında gerçekleştirmektedirler (Ricci et al., 2008b). Bir çalışma ortamında yer alan etmenler diğer etmenlerle iş bölümü ve kurallar dahilinde etkileşebilmekte, artifakt oluşturabilmekte, mevcut artifaktlardan yeni artifaktlar türetebilmekte, mevcut artifaktları paylaşabilmekte ya da birlikte kullanabilmektedirler.

A&A meta-modelinde bir çalışma ortamı etmenlerin aktivitelerinin desteklenmesi amacı ile önceden tasarlanır, çalışma zamanında dinamik olarak inşa edilir ve etmenler tarafından kullanılır. Çalışma ortamlarının içerecekleri artifakt tiplerinin ÇES programcısı tarafından önceden belirlenmesi ve tasarlanması gerekir. Çalışma ortamları mantıksal çalışma alanlarına ("workspace") bölünmüştür. Bir çalışma alanı önceden belirlenmiş organizasyonel kuralları ve artifakt tipleri içeren mantıksal bir mekan oluşturur (Ricci et al., 2009; Ricci et al., 2008a). Etmenler katıldıkları çalışma alanlarında ihtiyaç duydukları artifaktları arayıp bulabilir, fonksiyonlarını kullanabilir ve gözlemleyebilirler. Bir etmen birden fazla çalışma alanına da katılabilir.

Ortamı A&A metamodeline göre tasarlanan bir ÇES'nde otonom, hedef ve görev odaklı kısmın temel inşa bileşenleri etmenler, otonom olmayan, fonksiyon odaklı kısmın temel bileşenleri ise artifaktlardır. Etmenler artifaktları örnekleyebilir, keşfedebilir, paylaşabilir, kullanıp gözlemleyebilir. Artifaktlar ise etmenlerin ortamında bulunan, etmenlerin hedeflerini gerçekleştirmek ve görevlerini desteklemek için bireysel ya da toplu olarak kullandıkları, bir çeşit fonksiyon ya da servis sunmak üzere tasarlanmış olan yazılımsal aygıtlardır (Ricci et al., 2009).

Bir artifaktın tanımlanması için dört temel bileşen kullanılır: kullanım arayüzü ("Usage Interface"), çalıştırma talimatları ("Operating Instructions"), fonksiyon, yapı ve davranış (Ricci et al., 2006).

Gerçek dünyadaki artifaktlara benzer şekilde, yazılımsal artifaktların kullanım arayüzleri de, etmenlerin bir operasyonun çalışmasını başlatabilecekleri ve kontrol edebilecekleri bir kullanım arayüzü kontrolleri kümesinden oluşur. Her kontrol bir etiket (operasyon ismi) ve girdi parametreleri listesiyle tanımlanır. Operasyonlar artifakt fonksiyonelliğini oluşturmakta kullanılan temel birimler olup,

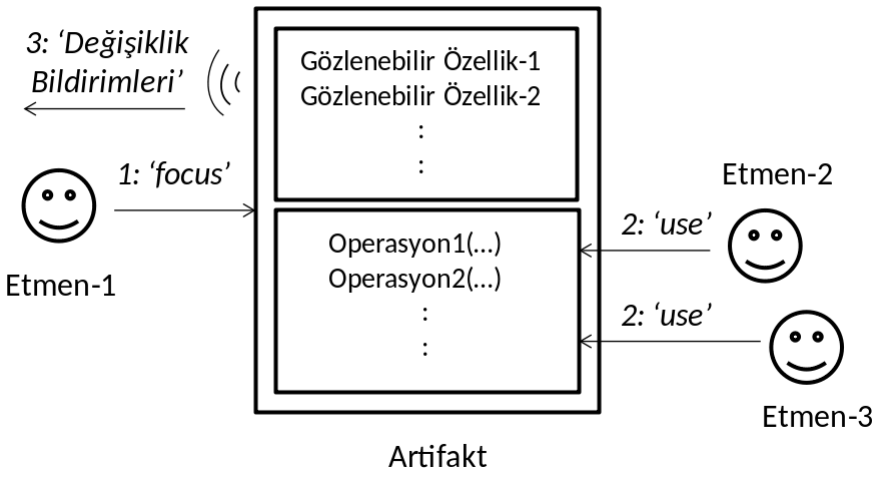
atomik ya da atomik basamakların sıralanmasından oluşan süreçler olabilirler. Bir operasyonun çalıştırılması hem artifaktların iç durumlarında gözlenemeyen değişikliklere neden olabilir, hem de ya artifaktı kullanan ya da gözleyen etmenler tarafından algılanabilen gözlenebilir dış olaylar olarak algılanırlar (“perceptions”) (Ricci et al., 2009; Ricci et al., 2006) .

Bir etmen-artifakt etkileşimi, etmenin kullanım arayüzündeki bir kontrolü seçmek ve gerekli parametreleri vermek için bir kullan (“use”) eylemi gerçekleştirmesiyle başlar. Eğer kullan eylemi başarılı olursa, artifaktın ilgili kullanım arayüzü kontrolüne bağlı olan operasyonun yeni bir örneği artifaktın içinde çalışmaya başlar. Operasyonun çalıştırılması, hem artifaktı gözlemleyen hem de kullanan etmenler tarafından gözlemlenebilen bir olaylar dizisinin oluşmasına neden olur. Artifaktlar, bir operasyonun çalışmasının bitmesi, gözlemlenebilir bir özelliğin değişmesi ya da operasyonun başarısız olması gibi durumlarda, tiplerinden bağımsız olarak, bazı basit olaylar oluştururlar (Ricci et al., 2009).

Artifaktlar tarafından üretilen olaylar etmenler tarafından aktif ya da pasif olarak algılanabilir. Aktif durumda, kullan eylemini gerçekleştiren etmen dışsal olarak bir almaç (“sensor”) tanımlar. Gözlenebilir olaylar üretilir üretilmez algılanarak almaçların içinde biriktirilirler. Daha sonra, etmen bu algılara ihtiyaç duyduğunda, bir algıla (“sense”) eylemi ile almaçtan bu algıları çekebilir. Etmen algıla eylemi sırasında aktiviteyi kendisine uyan bir algı bulunana kadar durduran filtreler kullanabilir. Bu durumda almaçlar, etmen tarafından dışsal olarak yönetilebilen algısal bir bellek görevi görürler. Pasif durumda ise, artifaktlar tarafından üretilen olaylar etmene ya bir iç olay ya da bir inanç olarak iletilir; almaçlar aracı olarak kullanılmaz (Ricci et al., 2009).

Artifaktlar gözlenebilir olayların yanı sıra, değişimleri kendisiyle etkileşimde olmayan etmenler tarafından da algılanabilecek, dinamik değerli gözlenebilir özelliklere de sahip olabilirler.

Etmenler artifaktları kullanabilecekleri gibi sadece gözlemleyebilirler de. Etmenler kullanmadıkları artifaktları gözleyerek, bu artifaktların gözlenebilir özelliklerini ve ürettikleri gözlenebilir olayları takip edebilirler. Gözlemlenebilir özellikler doğrudan etmenlerin algılarına, dolayısıyla inançlarına yansıtılır. Gözlemlenebilir olaylar ise almaçlar aracılığıyla odaklan (“focus”) eylemi ile takip edilir (Ricci et al., 2009). Şekil 3.3’te Etmen-1 diğer etmenlerin use eylemi ile çalıştırdıkları operasyonlar sonucu oluşan gözlenebilir özellik değişimlerini focus eylemi ile odaklanarak izlemektedir.



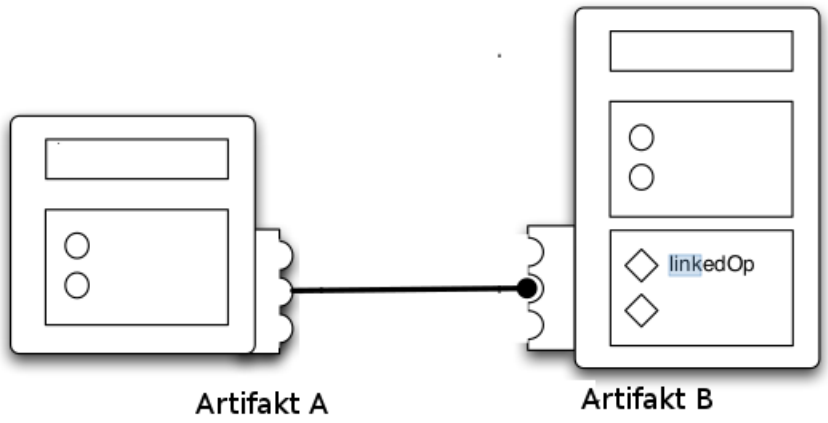
Şekil 3.3: Temel artifakt yapısı ve artifaktın 'focus' eylemi ile gözlenmesi.

İnsanların kullandıkları artifaktlara benzer şekilde, yazılımsal artifaktların de kılavuzları olabilir. Bu kılavuzlarda artifakt fonksiyonu, kullanım arayüzü ve çalıştırma talimatları bulunur. Yazılımsal artifaktların kılavuzları, özellikle akıllı etmenlerin alet seçimi ve kullanımını doğru gerçekleştirerek amaçlarına varabilmeleri için, çalışma zamanında denetlenmek ve kullanılmak üzere oluşturulurlar (Ricci et al., 2009). Çalıştırma talimatları ise bir artifaktın fonksiyonlarından yararlanmak için nasıl kullanılması gerektiğini tarif eder. Çalıştırma talimatları artifaktın üstünde başlatılabilecek operasyonlar gibi olası kullanım protokollerini tanımlar. Çalışma talimatları, artifaktların bilişsel kullanımını olası kılmak ve teşvik etmek için söz dizimsel bilginin yanı sıra, etmenlerin çıkarsamalarında kullanabilecekleri anlamsal bilgiler de içerebilir (Ricci et al., 2006).

Bir artifaktın yapısı ve davranışı ise iç yapısı ile ilişkili olup, artifaktın hedeflenen fonksiyonları sunmak için nasıl gerçeklendiğini tanımlar. Artifaktların bu yönü, tipik olarak, kullanıcılardan saklanır (Ricci et al., 2006).

Yazılımsal artifaktların sahip oldukları bağlantı arayüzleri ("link interface") vasıtasıyla birbirlerine bağlanmaları da mümkündür. Artifaktların biribirlerine değişik şekillerde bağlanmaları ile farklı işlevleri olan bileşik artifaktlar meydana getirilebilir. Şekil 3.4'te Artifakt A'ın Artifakt B'nin bağlantı arayüzündeki linkedOp operasyonunu çalıştırması sağlanarak iki artifaktın birlikte çalıştığı bileşik bir artifakt meydana getirilmektedir.

Etmenlerin aksine, artifaktlar otonom, proaktif ve sosyal olacak şekilde tasarlanmazlar. Artifaktların doğaları gereği sahip oldukları diğer özellikler



Şekil 3.4: Artifakt A Artifakt B'nin bağlantı arayüzünde yer alan linkedOp operasyonunu çalıştırıyor.

şunlardır (Viroli et al., 2005; Ricci et al., 2006):

- Denetlenebilirlik: Etmenlerin denetle ("inspect") operasyonu aracılığıyla, artifaktların durum, yapı ve davranışlarını çalışma zamanında kontrol edilebilmesi, ayrıca artifakt yönetiminin hata nedeni bulma, hata ayıklama ve test açısından desteklenmesidir.
- Şekillendirilebilirlik: Açık ÇES'de, artifaktların fonksiyonlarının sistemin gereksinimleri doğrultusunda çalışma zamanında değiştirilebilmesi ya da adapte edilebilmesidir.
- Tahmin edilebilirlik: Artifaktların servis tanımlamaları ve çalıştırma talimatları, etmenlerin artifaktların davranışlarını tahminleyebilmelerini olanaklı kılar.
- Bağlanabilirlik: Dinamik yeniden kullanılabilirliği ve ölçeklenebilirliği sağlamak için artifaktların çalışma zamanında birbirine bağlanabilmesidir.

Artifaktlar A&A metamodelinde tek etmen tarafından kullanılan bireysel artifaktlar, birden fazla etmen tarafından kullanılan sosyal artifaktlar ve kaynak artifaktları olarak üçe ayrılmaktadır (Omicini et al., 2006).

Kaynak artifaktları dış kaynakları kavramsal olarak sararlar, yani ÇES ve dış kaynaklar arasında aracılık görevi yaparlar. Dış kaynaklar eski sistemler ve araçlar, etmen olmayan teknolojiler için yazılmış uygulamalar, ya da etmenlerin algılayıp tepki göstermeleri gereken fiziksel kaynaklar olabilir. Prensip olarak, kaynak artifaktları dış ÇES kaynaklarını etmen bilişsel seviyesine çıkarmak için

kullanılırlar. Dış kaynaklara bir kullanım arayüzü ve işletim talimatları vasıtasıyla erişim sağlar , yüksek düzeyli etmen-kaynak etkileşimlerini kaynakların düşük düzeyine dinamik olarak eşlerler.

Sosyal artifaktlar ise birden fazla etmen tarafından kullanılan ve etmenler arasında aracılık yapan artifaktlardır. Bu artifaktlar genellikle ÇES'ndeki sosyal hedeflere ulaşılması için gerekli olan koordinasyon gibi hizmetleri sağlarlar (Viroli et al., 2005).

CArtAgO Ortam Çatısı

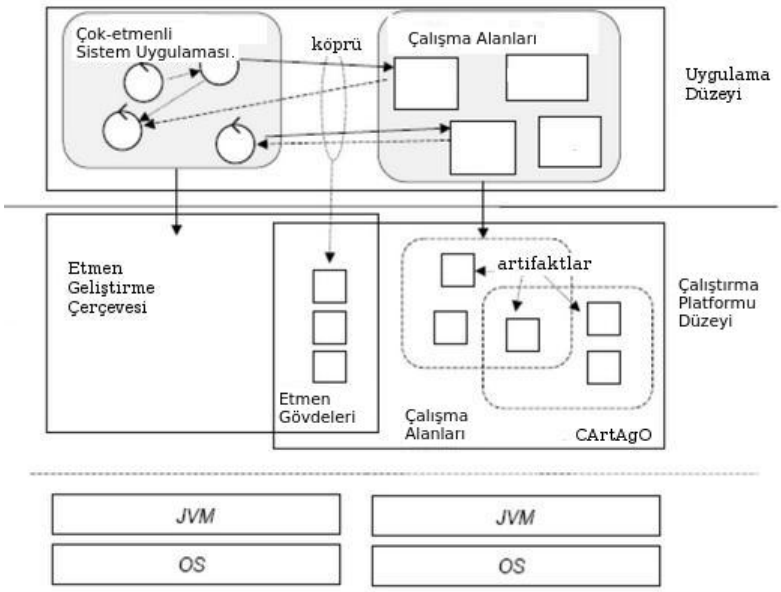
CArtAgO (Common ARTifact infrastructure for AGents Open environments), A&A meta-modelinin Java tabanlı gerçekleştirimidir. Literatürde A&A metamodelini gerçekleyen başka bir sistem bulunmamaktadır.

CArtAgO ortam çatısının da temel yapıları A&A metamodelinde olduğu gibi çalışma alanları ve artifaktlardır. CArtAgO platformunun üç temel arayüzü vardır (Ricci et al., 2009):

- Artifakt programlama arayüzü: ÇES'lerindeki ortamın A&A metamodeline göre geliştirilmesi sırasında ortamın içereceği artifaktlar CArtAgO'nun artifakt programlama arayüzü kullanılarak geliştirilmektedir.
- Etmen uygulama geliştirme arayüzü: Etmenlerin CArtAgO çalışma alanlarında artifakt oluşturmaları ve bunlarla etkileşmeleri, ayrıca çalışma alanlarına katılmaları ve bunlardan ayrılmaları için gerekli olan temel eylemleri içerir.
- Çalışma zamanı ortamı ve ilgili araçlar: Çalışma alanlarının çalışmasını, yönetimini ve artifaktların yaratılmasını, çalıştırılmasını ve yok edilmesini destekleyen arayüzdür.

CArtAgO kullanılarak geliştirilen etmen ortamları farklı etmen platformları ile çalışabilmektedir. Bu amaçla ilgili etmen platformu ile CArtAgO ortamını bağlayan bir köprü yazılımının geliştirilmesi gerekmektedir. Etmen uygulama geliştirme arayüzü köprü yazılımı tarafından desteklenmelidir.

Etmenler bir çalışma alanı ile etmen uygulama geliştirme arayüzü aracılığı ile etkileşirler. Bir etmenin bir çalışma alanı üzerinde gerçekleştirmek istediği bir eylem köprü yazılımı tarafından çalışma ortamına iletilir, ortamdan etmene gelen



Şekil 3.5: CArtAgO Ortam Çatısı'nın mimarisi (Ricci2007a'dan, 2007)

bilidirimler de yine köprü yazılımı aracılığı ile etmene iletilir. Etmenler etmen uygulama arayüzünü kullanarak aşağıdaki eylemleri gerçekleştirebilirler:

- çalışma alanlarına katılıp çalışma alanlarından ayrılabilirler (joinWorkspace ve quitWorkspace metotları),
- artifakt oluşturabilir, arayabilir ve silebilirler (makeArtifact, lookupArtifact ve disposeArtifact metotları),
- artifakt arayüzünde sunulan operasyonları işletebilir ve artifaktlara odaklanarak gözlenebilir özelliklerini izleyebilirler (use, focus ve observeProperty metotları- use metodu, artifakt'a ilişkin sistem tarafından verilen tanımlayıcı değerini, ilgili operasyon adını ve operasyonun ihtiyaç duyduğu parametre listesini parametre olarak almakta ve operasyonu işleterek sonucu döndürmektedir),
- artifaktları birbiri ile bağlayabilirler (linkArtifacts, unLinkArtifacts metotları).

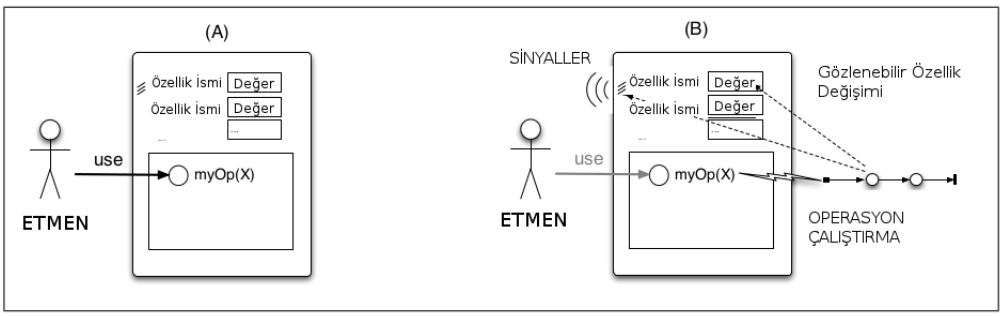
Bir çalışma alanı oluşturulurken önceden belirlenmiş bir artifakt kümesini içerecek şekilde oluşturulmaktadır. Burada amaç çalışma alanı içinde yer alan etmenler ve çalışma alanı yönetimi için gerekli olan temel fonksiyonların ve servislerin sağlanmasıdır. Her CArtAgO çalışma alanınıda artifaktların kaydını tutarak izlenmesini sağlayan bir kayıt artifaktı ile insan kullanıcılar tarafından okunması gereken mesajları basan standart çıktı artifaktı olan bir konsol

artifaktı bulunmaktadır CArtAgO kullanan bir çok-etmenli sistem uygulamasına ilişkin uygulama düzeyi ve çalıştırma platformu düzeyi katmanları Şekil 3.5'te görülmektedir. Uygulama katmanında, birbirileri ile etkileşen etmenlerden oluşan bir çok-etmenli sistem uygulaması ile bu uygulama içersinde yer alan etmenlerin kullandıkları CArtAgO çalışma alanları görülmektedir. Çalıştırma platformu katmanının en önemli bileşeni etmenler ile CArtAgO ortamının arasında köprü görevi yapan etmen gövdeleridir.

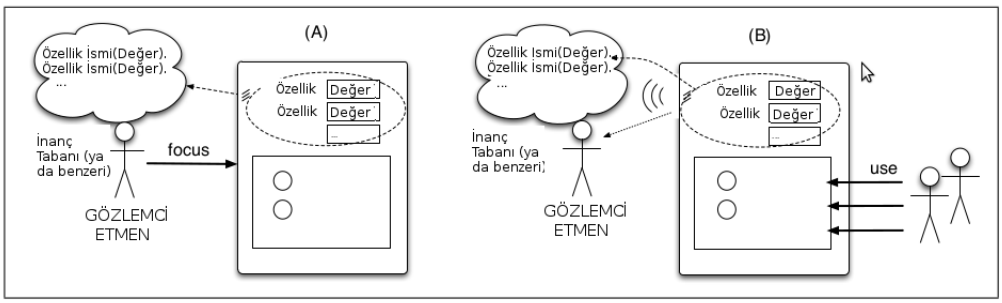
Etmen gövdeleri CArtAgO çalışma ortamı ile etmenler arasındaki bağlantıyı sağlamaktadır. Bir CArtAgO çalışma ortamında çalışacak her etmen için bir gövde oluşturulmaktadır. Bir etmen gövdesi çalışma ortamı üzerinde eylemler gerçekleştirilmesi için efektörler ("effector"), ve çalışma ortamından uyarıcılar toplamak için dinamik bir algılayıcılar ("sensor") kümesi içermektedir. Etmen gövdesi, etmen tarafından kontrol edildiği için etmenin kendisini kontrol etmesini sağlayacak bir arayüze de sahiptir.

Etmenler gövdelerini yöneterek, artifakt oluşturulması, seçimi ve kullanımı eylemlerini gerçekleştirmek ve bu artifaktlar tarafından üretilen gözlemlenebilir olayları algılamak suretiyle çalışma ortamları ile etkileşmektedirler. Şekil 3.6'da bir etmenin bir artifaktın myOp operasyonunu çalıştırması sonucu gözlenebilir özelliklerin değiştirilmesi ve Şekil 3.7'de de bununla bağlantılı olarak ilgili artifakta odaklanmış olan etmenlerin gövdelerindeki algılayıcılara değişen gözlenebilir özellik değerlerinin aktarılması görülebilir. Etmen gövdesinin parçası olan algılayıcılarda biriktirilen algılar etmen istediğinde gerekli filtrelerden geçirilerek kullanılabilmektedir. Bir etmen, tamponlama, filtreleme ya da sıralama gibi değişik fonksiyonları olan birçok algılayıcıyı dinamik ve esnek bir şekilde gövdesine bağlayabilir ya da mevcut algılayıcıları gövdesinden ayırabilir. Bu nedenle, CArtAgO'da algılama algılayıcılardaki uyarıların toplanmasını ve etmenin bu uyaranların farkına varmasını sağlayan bir iç eylem olarak tanımlanmaktadır.

CArtAgO sisteminde etmen gövdeleri çalışma alanında yer almalarına karşın, kendilerini yöneten etmenler etmen platformunda bulunurlar. Mimari açısından bakıldığında, etmenler ve gövdelerini bağlamak için daha önce de söz edilmiş olan köprü yazılımları ("bridge") gereklidir. Bu köprüler, gövdeyi kontrol etmek ve gövde algılayıcılarından gelen uyaranları algılayabilmek için etmen tarafında çalışan ambalajlardır ("wrapper"). JASON, JADEX (Ricci et al., 2008a) etmen platformları ile CArtAgO platformu arasında bağlantıyı sağlayan köprüler bulunmaktadır.



Şekil 3.6: (A) Etmen artifaktın arayüzünde listelenmiş myOp operasyonunu çalıştırıyor. (B) myOp operasyonu çalıştırıldığında artifaktın gözlenebilir özellikleri değişiyor, değişiklikler artifaktı gözleyen etmenlerin algılayacağı sinyaller şeklinde iletiliyor (Ricci'den, 2011).



Şekil 3.7: Etmen bir artifakt üzerinde focus eylemini uyguladığında artifaktın özelliklerinde oluşan değişimleri algılar ve ortam hakkındaki bilgilerini günceller (Ricci'den, 2011).

4 ÇOK ETMENLİ SİSTEMLERDE ONTOLOJİ VE BAĞLI VERİ DEĞİŞİMLERİNİN YÖNETİLMESİ

ÇES'de ontoloji ve Bağlı Veri değişimlerinin izlenmesindeki temel amaç etmenlerin bu değişimlere zamanında tepki verebilmesini sağlamaktır. Ancak literatürde ontoloji değişimlerinin takibi için kullanılan metotlar ontoloji versiyonlama ve ontoloji evrimi metotlarının her ikisinde de amaç uygulamaların ve kullanıcıların değişimlerden mümkün olduğunca az etkilenmesini sağlamaktır. Ontoloji değişimlerinin uygulamalar üzerinde yarattığı etkiyi ve bu etkinin yönetimini inceleyen çalışmalar da vardır (Liang et al., 2006b; Liang, 2006; Liang et al., 2006a). Bu çalışmadaki amaç ise uygulamalardan gönderilen uygulama sorgularının ontolojideki değişikliklerle paralel şekilde güncellenmesi ve yeni sorgu sonucu döndürülen yanıtların uygulamaya iletilmesidir. Dolayısıyla, bu çalışmada etmenlerin tepkiselliğine sahip olmayan uygulamalar hedef alınmaktadır.

Bağlı Veri değişimlerinin izlenmesinde kullanılan isteme ve itme tabanlı yöntemler etmen sistemlerini değişim oluşması durumunda uyardığı ve etmenlerin sürekli sorgulama yapmasını gerektirmediğinden avantajlı gözükmektedir. Ancak, bu sistemlerde de bir takım dezavantajlar bulunmaktadır:

- Etmen programcısı kullanacağı isteme ve itme tabanlı sistemin uygulama geliştirme arayüzünü kullanmak durumundadır. Kullanılacak sistemin değiştirilmesi etmen sistemindeki kodun da değişmesini gerektireceğinden uygulama geliştiricisine hem bilgi dağarcığı hem de kodlama açısından ek yük getirmektedir.
- ÇES'de etmenlerin ihtiyaç duydukları değişiklik bildirimleri genellikle sonuçları sadece kendilerini ilgilendiren bilgiyi içerdiğinden sorgu düzeyinde olmaktadır. Mevcut isteme ve itme tabanlı sistemlerden sorgu düzeyinde destek verenler olmasına karşın (Passant and Mendes, 2010; Popitsch and Haslhofer, 2011) hiçbir sistem federe sorguların ve bu sorguların bağlı oldukları ontolojilerdeki değişimlerin izlenmesini desteklememektedir.
- İç kökenli ortamlar ÇES'nin geliştirilmesini sağladıkları soyutlama sayesinde kolaylaştırmakta ve sistem performansını yükseltmektedirler. Bu yaklaşıma göre ÇES'de kaynaklara erişim ve kaynakların algılanması yerel ortamdan sağlanmalıdır (Weyns et al., 2007). Mevcut hiçbir sistem ÇES'nin Bağlı Veri Ağı ve Anlamsal Veb üzerindeki kaynaklara erişimine aracılık ederek kaynak erişimini ve kaynakların algılanmasını yerel ortam içinden sağlamamaktadır.

Tüm bu veriler ışığında ÇES'nde etmenlerin Anlamsal Veb ve Bağlı Veri kaynaklarına yerel olarak erişmesini ve bu kaynakları yerel olarak gözlemlemesini olanaklı kılacak, etmenleri değişimlerden isteme ve itme tabanlı bir yöntemle haberdar edecek iç kökenli bir ortam soyutlamasına ihtiyaç duyulduğu söylenebilir. Tez kapsamında geliştirilen ve gerçekleştirilen iç kökenli ortam modeli ile bu soruna çözüm getirilmeye çalışılmıştır. Tezin bu bölümünde sırasıyla soyut mimari ve gerçekleştirilen mimari incelenecektir.

4.1 Soyut Mimari

3. bölümde de belirtildiği üzere Anlamsal Veb Dağıtık bilişselliğin bilgisayar teknolojisindeki uygulamalarından biridir. Anlamsal Veb ve Bağlı Veri Ağı'nda bilgi gösterimi ve bilişsellik etmenler, Anlamsal Veb Servisleri, ontolojiler ve verisetleri üzerinde dağıtık haldedir. Etmenler dağıtık durumdaki bu bilgiyi kullanarak aralarındaki koordinasyonu ve birlikte işleyebilirliği sağlamaktadırlar. Bu bilgi gösterimleri Aktivite Teorisi'ndeki artifaktlara karşılık gelmektedir (Kaptelinin and Nardi, 2006). Dolayısıyla tez kapsamında geliştirilen sistemdeki veriseti, ontoloji ve sorgu gösterimlerinin herbirinin bir artifakta karşılık geldiği söylenebilir.

Etmen literatüründe etmen dışında kalan sistem bileşenleri etmen ortamının parçaları olarak görülmektedir. Dolayısıyla Anlamsal Veb ve Bağlı Veri Ağı'nın izlenmesi de ortam içinde hizmet veren bileşenler tarafından sağlanması gereken hizmetlerdir. Tezin 3. bölümünde de belirtildiği üzere ortamın içkökenli bakış açısı ile modellenmesi sağladığı soyutlama nedeniyle karmaşık ÇES uygulamalarının tasarlanmasını ve gerçeklenmesini kolaylaştırmaktadır. Bu yönüyle içkökenli ortam yaklaşımı bu tezin etmen sistemlerinin geliştirilmelerini kolaylaştırma ve yüklerini azaltma amacına uygun düşmektedir. İçkökenli etmen ortamlarının tezin hedeflerine uygun şekilde etmenlerin bulundukları konuşlanma bağlamına erişimlerini düşük seviyeli detaylardan soyutlamaları beklenmektedir.

A&A metamodeli hem tek iç kökenli ortam modeli olması, hem de Aktivite Teorisi'ne dayandığından verisetleri, ontolojiler ve sorguların artifaktlarla gösterimini olanaklı kılması nedeniyle sistemin soyut mimarisinin tanımlanması için uygun bulunmuş ve kullanılmıştır.

Tez kapsamında A&A metamodeli özelleştirilerek itme ve isteme tabanlı bir anlamsal ortam modeli geliştirilmiştir (Erdur et al., 2012; Erdur et al., 2013). Geliştirilen ortam modeli anlamsal çalışma alanlarından oluşmaktadır.

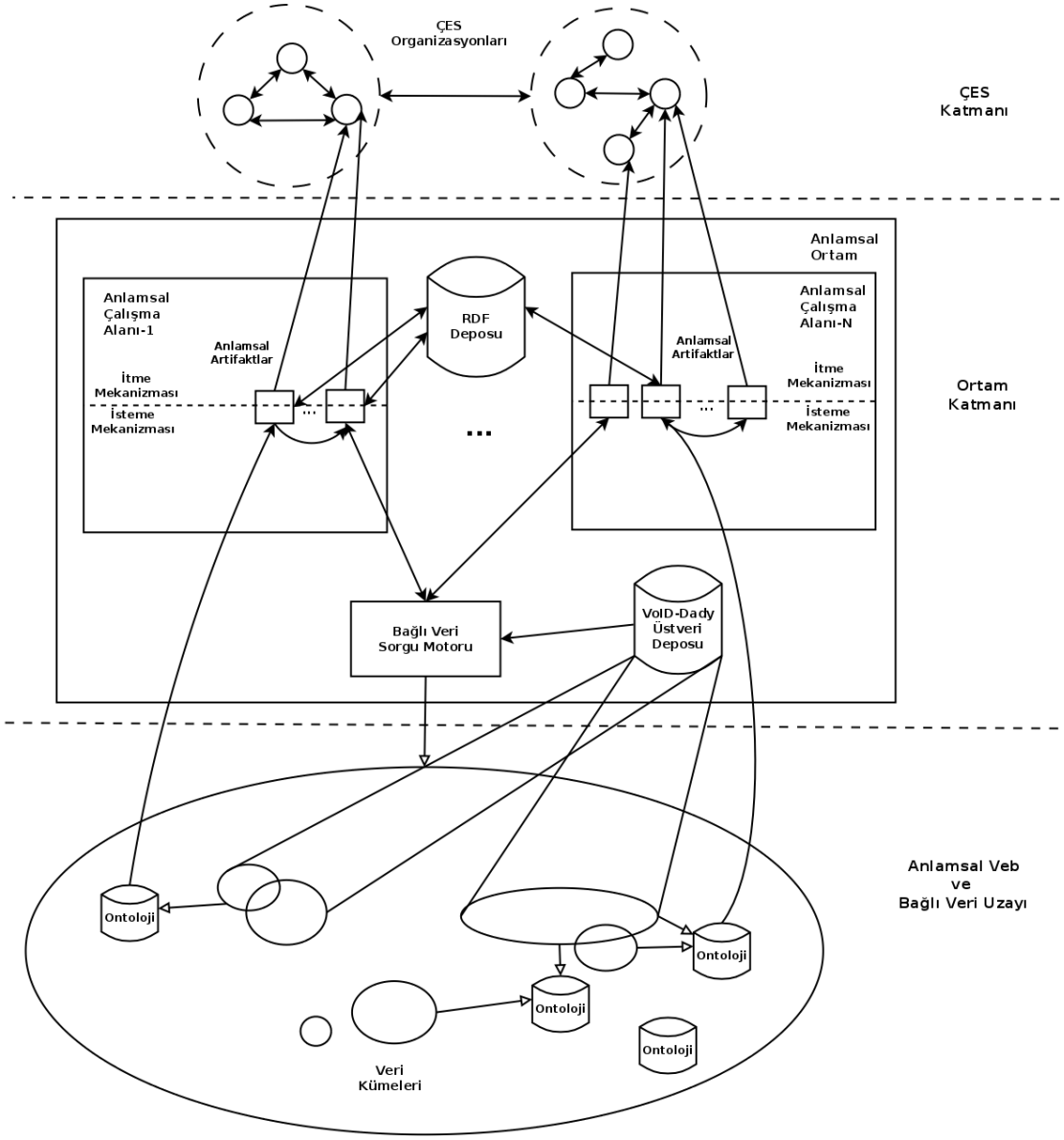
Her anlamsal çalışma ortamı izlenen veri kümelerine ilişkin VoID (Alexander and Hausenblas, 2009) ve Dady (Hausenblas, 2010) üst bilgilerini içeren bir üst veri deposu, izlenen sorguların sonuçlarını ifade eden RDF çizgelerini ve anlamsal ortamın sınırları dahilinde izlenen veri kümelerinin kullandığı ontolojileri tutan bir RDF deposu ve federe sorguları çözümleyip işletebilen bir bağlı veri sorgu motoru içermektedir.

Daha önce 2.2.4 numaralı bölümde de belirtildiği gibi, isteme tabanlı yaklaşımda, bağlı veriyi kullanan uygulamalar periyodik olarak veri kaynağında değişiklik olup olmadığını kontrol etmektedirler. İtme tabanlı yaklaşımda ise veri kaynakları içerdikleri ve izlenen öğelerinde bir değişiklik olduğu anda ilgili öğeleri izleyen uygulamaları bilgilendirmektedir. Bunun için, uygulamaların ilgili izleme seviyesinde veri kaynağı tarafına üye olmaları gerekmektedir.

Benzer şekilde, anlamsal çalışma ortamında çalışmakta olan etmenler de sisteme sonuçlarında değişiklik olduğunda haberdar olmak istedikleri sorguları kaydederler. Böylelikle sistemin izlemek istedikleri veride değişiklik olduğunda ilgili değişiklikleri kendilerine bildirmesini yani bu değişiklikleri kendilerine itmesini sağlarlar. Sistem de sonuçları izlenecek sorguları periyodik olarak orijinal kaynaklarının üzerinde çalıştırarak veriyi kaynağından çeker yani ister. Bu bölümde incelenecek olan soyut mimari ve bileşenleri Şekil 4.1'de görülebilir.

VoID dokümanları anlamsal çalışma ortamlarının sınırlarını çizer. Anlamsal çalışma alanlarının izlediği veri uzayına yeni veri kümelerinin eklenmesi gerektiğinde üst veri deposuna ilgili VoID dokümanlarının eklenmesi, mevcut veri kümelerinin çıkarılması gerektiğinde ise ilgili VoID dokümanlarının silinmesi yeterli olmaktadır. Dolayısıyla, anlamsal çalışma alanının etki alanı çalışma zamanında genişletilip daraltılabilmektedir. Her VoID dokümanında ise o dokümanın ilgili olduğu veri setinin değişimleri ile ilgili istatistiksel üst veri yer almaktadır.

Bir anlamsal çalışma alanı kendisine bağlanan etmenlerin Anlamsal Veb ile Bağlı Veri Uzayı'na erişimini ve buradaki verinin değişimini gerek sorgu gerekse ontoloji düzeyinde izleyebilmesini sağlayacak anlamsal artifaktlardan oluşmaktadır. Anlamsal çalışma alanları bir etmen organizasyonunun bir uygulama kapsamında erişmesi ve izlemesi gereken Anlamsal Veb ve Bağlı Veri parçalarını tanımlayan mantıksal haznelerdir. Anlamsal çalışma alanlarında izlenen sorgulara ait sonuç çizgelerinin ve kullanılan ontolojilerin yerel kopyaları anlamsal çalışma ortamında bulunan RDF deposunda tutulmaktadır.



Şekil 4.1: Geliştirilen sistemin soyut mimarisi

Anlamsal artifaktlar etmenlerin Anlamsal Veb ve Bağlı Veri Ağı ile etkileşim için kullandıkları özelleşmiş artifaktlardır. Bu artifaktların üç temel türü vardır:

- **yönetimsel artifakt:** Anlamsal ortamdan veri izleme hizmeti almak isteyen etmenler değişimlerini izlemek istedikleri sorguları yönetimsel artifakt aracılığı ile sisteme kaydettirirler. Yönetimsel artifakt etmen tarafından izlenmek istenen sorgu önceden kaydedilmiş ise ilgili sorguyu izleyen artifaktın kimliği istemci etmene iletir. Etmen kimliği kendisine iletilen artifakta odaklanarak bu artifakttan gelecek değişiklik sinyallerini izleyebilir duruma gelir. Değişiklik sinyallerinin etmen tarafından izlenebilmesi mimarinin itme fonksiyonuna karşılık gelmektedir. Eğer etmenin izlemek istediği sorgu önceden kaydedilmemiş ise yönetimsel artifakt yeni bir sorgu artifaktı yaratır ve bu artifaktın kimliğini istemci etmene iletir. Etmen daha sonra ilgili artifakta odaklanır.

- **sorgu artifaktları:** Sorgu artifaktları kendilerine verilen SPARQL sorgularının sonuçlarını anlamsal ortamda bulunan bağlı veri sorgu motorunu kullanarak izlerler. Her sorgu hedeflediği veri setlerinin Dady üstverisinde belirtilen değişim sıklıklarına göre bağlı veri sorgu motoru aracılığı ile çalıştırılır. Böylelikle değişiklik uyarı sisteminin isteme mekanizması gerçekleştirilmiş olur. Bağlı veri sorgu motorundan dönen sonuçlardan çizge tipinde olanlar RDF deposuna kaydedilir. Her sorgu artifaktı izlediği sorgu için elde ettiği yeni sonuçla bir önceki sonucu karşılaştırır. Sonuçlarda değişiklik olması durumunda sorgu artifaktı kendisini izleyen etmenlerin algılayabileceği şekilde gözlemlenebilir değişiklik özelliğini günceller. Etmenlerin sorgu sonuçlarındaki değişikliği algılamasının sağlanması ile de uyarı sisteminin itme kısmı gerçekleştirilmiş olur.

  Sorgu artifaktlarının bir diğer görevi de izledikleri SPARQL sorgusunda önek (“prefix”) olarak belirtilmiş olan ontolojilerdeki değişimleri izleyecek ontoloji artifaktlarını yaratmaktır. Bir sorgu artifaktı izlemeye başlayacağı sorguda önek olarak ifade edilmiş bir ontoloji ile karşılaştığında ilgili ontolojiyi takip eden bir ontoloji artifaktının sistemde mevcut olup olmadığına bakar. Eğer böyle bir artifakt mevcut ise sorgu artifaktı ilgili ontoloji artifaktına kaydolur. Eğer böyle bir artifakt mevcut değilse sorgu artifaktı ihtiyaç duyduğu ontoloji artifaktını yaratır ve bu artifakta kaydolur. Kaydolunan ontoloji artifaktı izlediği ontolojide bir değişim meydana geldiğinde kendisinde kayıtlı olan sorgu artifaktlarını uyarır, sorgu artifaktları da birer tekil olay etiketi oluşturarak etmen platformundaki ilgili etmeni bir sinyal göndererek uyarır.

- **ontoloji artifaktları:** Ontoloji artifaktları izledikleri ontolojiyi periyodik

olarak anlamsal veb üzerindeki adresinden indirir ve değişimler için karşılaştırırlar. Ontoloji versiyonlarının karşılaştırılması sırasında iki versiyon arasında yapısal ve mantıksal farklar aranır. Karşılaştırma sonucu elde edilen farklar bir fark çizgesi olarak ifade edilir ve gözlemlenebilir farkÇizgesi özelliğinin değerini oluşturur. Bir ontolojide mantıksal ya da yapısal düzeyde bir fark oluştuğunda bu bir iç ontoloji değişim olayı olarak adlandırılır ve bu olaya tekil bir olay etiketi atanır. Ontoloji artifaktları zincirleme ontoloji değişimlerini de takip ederler. Zincirleme ontoloji değişimi takibinde ise değişen bir ontolojiyi kullanan ("import" eden) diğer ontolojilerde oluşabilecek farklar da takip edilir. Bir ontoloji değiştiğinde o ontolojiyi kullanan diğer ontolojileri ve sorguları izleyen artifaktlar değişimden ontolojiDeğişimiBildirim operasyonlarının çalıştırılması ile haberdar edilirler. Doğrudan değişen ve değişimden etkilenmesi muhtemel olan ontoloji ve sorguları izleyen artifaktlar birer tekil olay etiketi oluşturur ve etmen platformundaki ilgili etmene sinyal göndererek uyarır.

Artifaktların ve görevlerinin incelenmesi sonrasında soyut mimari ile Weyns ve arkadaşları (Weyns et al., 2007) tarafından önerilen içkökenli ortam referans mimarisinin (bkz. Şekil 3.1) örtüştüğü noktalar daha net açığa çıkmaktadır. Bu referans mimarisi temel alındığında önerilen soyut mimarinin verdiği hizmetler etmen ortamlarının temel düzeyi ile soyutlama düzeyinde verilmesi gereken hizmetler olduğu görülmektedir. Sistemdeki artifaktlar

- etmenler tarafından kaydedilen sorguları periyodik olarak çalıştırarak olası değişimler için izlemektedir; bu hizmet konuşlanma bağlamındaki veri kümelerine erişim için temel seviyede ve ilgili sorguların sonuçların izlemek için soyutlama seviyesinde verilen desteklere karşılık gelmektedir,

- gelen sonuçları karşılaştırmakta ve değişimler için izlemektedirler; bu hizmet veri işleme için soyutlama seviyesinde verilen desteğe karşılık gelmektedir,

- sonuçlar arasında fark bulunduğunda yerel sonuç kopyasını güncellemektedirler; bu hizmet veri eşzamanlaması ("synchronization") için soyutlama seviyesinde verilen desteğe karşılık gelmektedir,

- bir sorgunun farklı olan iki sonucu arasındaki farkları ontolojik gösterime tercüme etmektedirler; bu hizmet tercüme etme için soyutlama seviyesinde verilen desteğe karşılık gelmektedir.

4.2 Gerçekleştirilen Mimari

4.2.1 Gerçekleştirilen mimaride kullanılan teknoloji ve araçlar

Gerçekleştirilen mimarinin temelini CArtAgO ortam çatısı oluşturmaktadır. Bu mimarinin üzerinde Bağlı Veri ve ontoloji değişimlerinin takibi amacıyla kullanılan teknolojiler ise şunlardır:

- Bağlı Veri ve ontoloji erişimi olan Java uygulamalarının geliştirilmesinde kullanılan Jena Uygulama Geliştirme Arayüzü,
- federe bağlı veri sorgularının parçalanması, ilgili oldukları veri kümelerinin bulunması ve çalıştırılmasında kullanılan WODQA sorgu motoru,
- ontoloji ve RDF çizgesi versiyonlarının karşılaştırılmasında kullanılan OWLDIFF aracının uygulama geliştirme arayüzü,
- OWLDIFF aracının karşılaştırma yaparken çıkarsama yapmak amacıyla kullandığı Pellet çıkarsama motoru
- ve yakalanan değişimlerin ifade edilmesinde kullanılan OWL için Değişim Ontolojisi'dir.

Kullanılan teknolojilerden CArtAgO çatısı tezin önceki bölümlerinde incelendiğinden bu bölümde tekrar ele alınmayacaktır.

Jena UGA (Uygulama Geliştirme Arayüzü)

Jena (Carroll et al., 2004) Java dili için geliştirilmiş bir Anlamsal Veb ve Bağlı Veri çatısıdır. Jena UGA'sı RDF, RDFS, OWL ve SPARQL teknolojilerini kullanan Java uygulamaları geliştirmeyi olanaklı kılmaktadır.

WODQA Sorgu Motoru

WoDQA (Akar et al., 2012) bağlı veri bulutu üzerinde federe sorgu çalıştırılmasını olanaklı kılan bir sorgu motorudur. Bir sorguyu dağıtmak amacıyla, WoDQA sorguyu yalnızca ilişkili veri kümelerinde çalışan birleştirilmiş bir şekle dönüştürmektedir. Bir sorgunun ilişkili veri kümelerini belirlemek için ise buluttaki

veri kümelerini temsil eden VoID belgeleri kullanılmaktadır. VoID belgeleri buluttaki veri kümeleri ile ilgili bilgileri içeren üst verilerdir ve bağlı veri bulutunun yansıması olarak düşünülebilir. WoDQA, üç ana bileşenden oluşmaktadır. Bu bileşenler, veri kümesi çözümleyicisi (DatasetAnalyzer), sorgu düzenleyici (QueryReorganizer) ve bağlı ARQ (Bound ARQ) olarak adlandırılmıştır.

Veri kümesi çözümleyicisi VoID belgelerini kullanarak ilişkili veri kümelerini belirlemek ve ilişkili olmayanları elemekle sorumlu birimdir. Veri kümesi çözümleyicisi sorgudaki her bir üçlü deseni için veri kümesi ve bağ kümesi tanımlarını çözümlemektedir. Bu çözümleme sorgunun sonucunu etkilemeyen veri kümelerini elemektedir. WoDQA veri kümesi çözümlemesini kural tabanlı olarak gerçekleştirmektedir. İkinci birim olan sorgu düzenleyici, veri kümesi çözümleyicisi'nden gelen sonuçlara göre sorguyu yeniden oluşturmaktadır. Bu yeniden oluşturma süreci iki eniyileştirme adımını içermektedir. Önce aynı veri kümesinden sorgulanacak üçlü desenleri gruplanmakta ve daha sonra her bir gruptaki üçlü deseni de işletim maliyetlerine göre kendi arasında sıralanmaktadır. Üçüncü bileşen ise Jena ARQ'yu (Foundation, 2015) FILTER tabanlı bağlı birleştirme mekanizması ile genişleten bağlı ARQ'dur. Bu bileşen, sorgu düzenleyici ile oluşturulmuş SPARQL sorgularını işletir ve işletim sırasındaki HTTP isteği sayısını azaltarak sonuçları daha hızlı bir şekilde elde etmektedir.

OWLDIFF ve Pellet

OWLDIFF, OWL dilinin son sürümü olan OWL2'yi de desteklemektedir. Bu nedenle, tez kapsamında OWLDIFF aracının kullanılması tercih edilmiştir.

Ancak, mantıksal fark alma için bir ontoloji çıkarsama motoruna da ihtiyaç duyulmuştur. Mantıksal fark, ontolojilerde gerçekleşen yapısal farkların yan etkisi olarak meydana gelen, bir ontolojinin iki sürümünden çıkarsama sonucunda elde edilen ifadelerin karşılaştırılması ile bulunabilen ontoloji farkı olarak tanımlanabilir. Bu gereksinim çerçevesinde mantıksal fark alma konusunda ihtiyaç duyulan çıkarsama desteği, OWLDIFF aracının kullandığı, literatürde çok bilinen ve kullanılan bir OWL2 destekli çıkarsama motoru olan Pellet (Sirin vd., 2007) kullanılarak sağlanmıştır. Pellet Jena ile de sağladığı programlama arayüzü ile kullanılabilmektedir.

ODO

Tezin önceki bölümlerinde tanıtılan ODO, tez kapsamında Anlamsal Veb üzerinde örnek (instance) seviyesindeki değişimlerin yanı sıra ontolojiler ile ilgili

olarak yapısal ve mantıksal değişimlerin de ifade edilmesinde kullanılmaktadır. Bu yapısal ve mantıksal değişimlerin ifade edilebilmesi için, ODO'ya bir takım sınıf ve özellikler eklenmiştir. ODO'ya eklenen sınıflar şunlardır:

- "AddSubClassOf" ve "RemoveSubClassOf": Bu sınıflar, alt sınıflarla ilgili değişikliklerin tanımlanmasında kullanılan "SubClassOfChange" sınıfının alt sınıflarıdır. "AddSubClassOfChange" bir sınıfa alt sınıf eklenmesi durumunda gerçekleşen değişikliği, "RemoveSubClassOfChange" ise bir sınıfın alt sınıflarından birinin silinmesi durumunda gerçekleşen değişikliği ifade etmekte kullanılmaktadır.

- "AddDataProperty" ve "AddObjectProperty": Bu sınıflar bir sınıfa veri özelliği ve nesne özelliği eklenmesi durumunda oluşan değişiklikleri ifade etmekte kullanılmaktadırlar.

- "RemoveDataProperty" ve "RemoveObjectProperty": Bu sınıflar bir sınıfa ait bir veri özelliğinin ya da nesne özelliğinin silinmesi durumunda oluşan değişiklikleri ifade etmekte kullanılmaktadırlar.

- "AddTypeSpecification" ve "RemoveTypeSpecification": Bu sınıflar ontolojide tanımlanan bir örneğin hangi sınıfa ait olduğunu ifade eden bir aksiyomun eklendiği ve silindiği durumlarda oluşan değişiklikleri ifade etmekte kullanılmaktadırlar.

Özellikler (properties) açısından bakılacak olursa, ODO'ya "hasRelatedEntity" isimli bir özellik eklenmiştir. "hasRelatedEntity" özelliği gerçekleşen bir değişiklikle ilgili olan sınıf, örnek ya da özelliklerin ifade edilmesi amacıyla kullanılmaktadır. Bu özelliğin alt sınıfları ise şunlardır: "hasRelatedClass" (değişiklikle ilgili sınıfı ifade etmek için kullanılır), "hasRelatedDataProperty" (değişiklikle ilgili veri özelliğini ifade etmekte kullanılır), "hasRelatedIndividual" (değişiklikle ilgili örneği ifade etmekte kullanılır), "hasRelatedObjectProperty" (değişiklikle ilgili nesne özelliğini ifade etmekte kullanılır).

Sonuç olarak, ODO bu eklentilerle birlikte anlamsal veb üzerinde gerek örnek seviyesindeki veri değişimleri gerekse yapısal ve mantıksal seviyede olan ontoloji değişimlerinin ifade edilmesinde ve etmen platformuna bildiriminde proje kapsamında kullanılan bir şema olarak nitelendirilebilir.

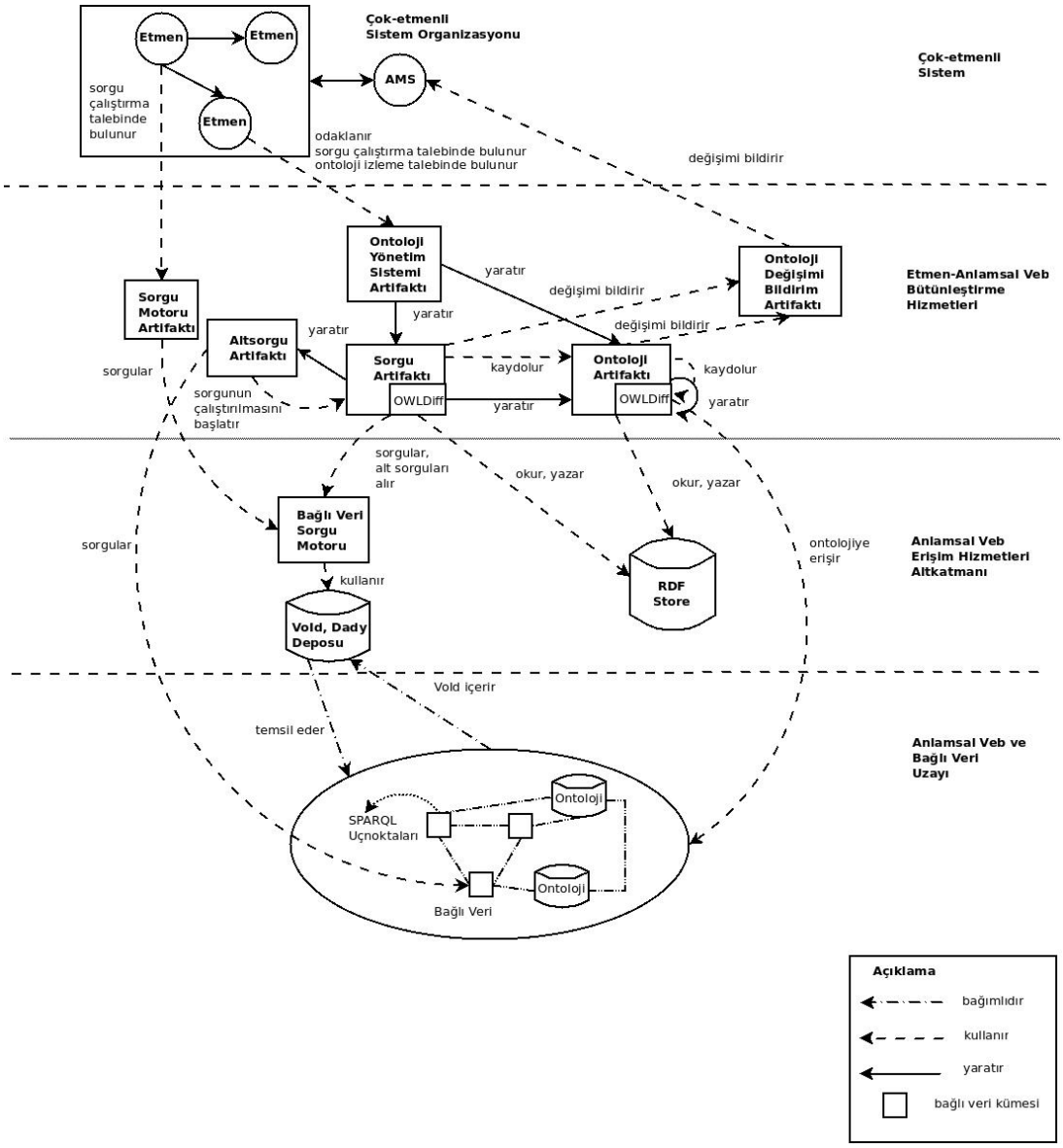
4.2.2 Mimarinin gerçekleştirimi

Anlamsal ortamın gerçekleştiriminde A&A meta-modelinin tek gerçekleştirimi olan CArtAgO çerçevesi kullanılmıştır. CArtAgO çerçevesinde ortam her biri belli bir amaca yönelik olan çalışma alanlarından meydana gelmektedir. Her çalışma alanında alanının amacına yönelik olarak oluşturulmuş artifaktlar bulunmaktadır. Bu bağlamda, anlamsal ortam etmenlerin anlamsal veb ve bağlı veri ile etkileşim için ihtiyaç duydukları hizmetleri etmenlerin etkileştikleri çalışma alanlarının içinde önceden tanımlanmış artifaktlar aracılığı ile sağlayan özelleşmiş bir CArtAgO ortamı olarak tanımlanabilir.

Anlamsal ortamın gerçekleştirim mimarisi Şekil 4.2'de görülmektedir. Üst katmanda yer alan çok-etmenli sistem uygulamaları anlamsal vebe ve bağlı veri uzayına anlamsal çalışma alanı üzerinden erişmektedirler. Anlamsal çalışma alanı etmenlere anlamsal veb üzerindeki verinin sorgulanması ve veriler ile ontolojilerdeki değişimin izlenmesi hizmetlerini vermektedir. Etmenler bir anlamsal çalışma alanına katılabilmekte ve çalışma alanının izlediği veri uzayı alt kümesinde yer alan veri kümelerini hedefleyen sorgular çalıştırabilmektedirler. Ayrıca, bu sorguların sonuçlarındaki değişimleri ve sorguların kullandıkları ontolojiler ile hedef, plan, rol gibi iç yapılarında kullandıkları ontolojilerdeki değişimleri takip edebilmektedirler. Sorgu sonuçlarının takip edilmesindeki amaç etmenlerin bu değişimlere zamanında tepki verebilmelerini sağlamaktır. Ontolojilerdeki değişimlerin takip edilmesindeki amaç ise sorguların ve etmelerin içsel yapılarının güncellenmesini gerektirebileceği için etmen sisteminin yönetim biriminin ("Agent Management System"-AMS) uyarılmasıdır.

Mimarinin ikinci katmanında ise, anlamsal ortam ve içinde yer alan bileşenler görülmektedir. Şekil 4.2'den de görüldüğü gibi anlamsal ortam iki alt katman şeklinde gerçekleştirilmiştir. Etmen-Anlamsal Veb Bütünleştirme Hizmetleri alt katmanında yer alan artifaktlar aşağıda detaylı şekilde açıklanacaktır. Bu artifaktlar,kavramsal mimaride belirtilen itme ve isteme mekanizmalarının gerçekleştiriminde kullanılmaktadırlar. Anlamsal Veb Erişim Hizmetleri alt katmanında ise bağlı veri sorgu motoru, veri kümeleri üst bilgilerini tanımlayan VoID ve Dady tanımları ile RDF deposu yer almaktadır. Bu iki alt katmanın oluşturduğu orta katman kavramsal mimaride gösterilen bir anlamsal çalışma alanına karşılık gelmektedir.

Mimarinin en alt katmanında ise anlamsal veb ve bağlı veri uzayı yer almaktadır. Her anlamsal çalışma alanı anlamsal veri uzayının belirli bir alt kümesi



Şekil 4.2: Anlamsal ortamın gerçekleştirim mimarisi

ile ilişkili olacak şekilde tanımlanmaktadır. Bir anlamsal çalışma alanını aynı alt kümeye odaklanan çok sayıda etmen kullanabilmektedir.

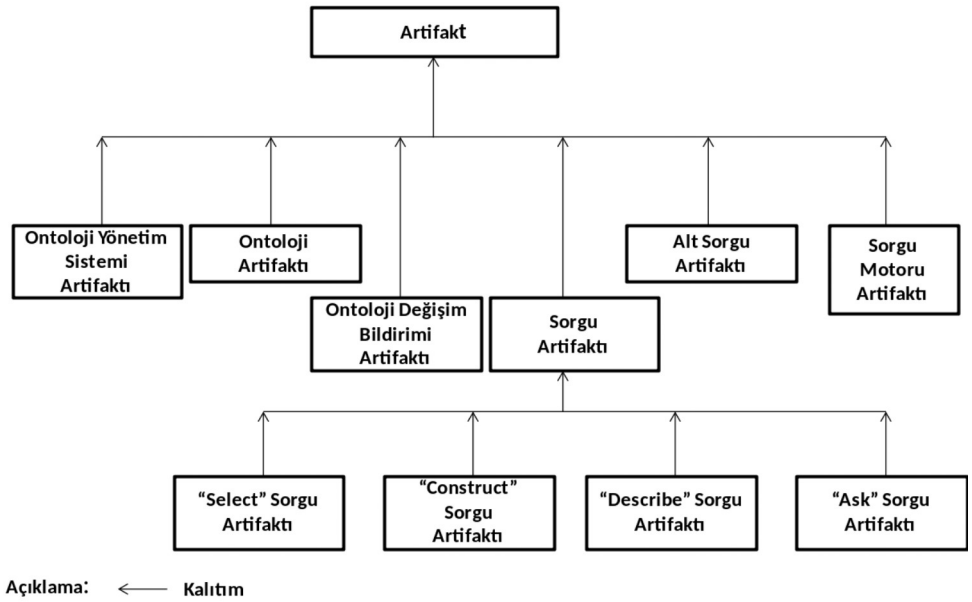
Bir anlamsal çalışma alanı başlatılırken odaklanılacak veri kümelerinin VoID ve Dady üst bilgi dokümanları sisteme verilmektedir. Bu üst bilgiler, bağlı veri sorgu motorunun kullanımına yöneliktir.

Şekil 4.2'den de görüldüğü gibi, Etmen-Anlamsal Veb Bütünleştirme Hizmetleri alt katmanında çeşitli artifaktlar yer almaktadır. Kavramsal mimaride anlamsal artifaktlar genel başlığı altında tanımlanan bu artifaktlar, A&A meta-modelindeki artifakt sınıfının özelleştirilmesi ile oluşturulmuşlardır. Şekil 4.3, bu artifaktlara ilişkin sıradüzeni vermektedir.

Anlamsal çalışma alanında standart CartAgO artifaktlarından sonra ilk olarak Ontoloji Yönetim Sistemi Artifaktı (OYSA) yaratılır. Bu artifaktın ilk görevi çalışma alanını ilklemek ve gerekli sorgu artifaktları ile ontoloji artifaktlarını yaratmaktır. OYSA, izlenen sorguların, bu sorguları izleyen artifaktların ve bu artifaktlardan hizmet alan uygulama örneklerinin (proje kapsamında etmenlerin) kayıtlarını tutmaktadır.

OYSA ilkleme işleminden sonra uygulamalardan gelen istekleri dinlemeye başlamaktadır. Bir SPARQL sorgusunun takip edilmesini isteyen bir etmen, isteğini OYSA'nın parametre olarak izlenecek SPARQL sorgusunu alan sorguyuKaydet (registerQuery) operasyonunu çalıştırarak bildirmektedir. Bu operasyon sorguyu daha sonra işlenmek üzere bir ilk giren ilk çıkar (FIFO) kuyruğuna yerleştirmektedir. Sırası gelen istek değerlendirildiğinde eğer izlenmesi istenen sorgu daha önceden izlenmeye başlanmış bir sorgu ise bu sorguyu izleyen artifaktın kimliği istemci etmene iletilmektedir. Bu noktada, etmenin kendisine kimliği döndürülen artifakta odaklanması ve bu artifaktın sonuç (result) isimli görünür özelliğindeki değişimleri dinlemesi gerekmektedir. Eğer izlenmesi istenen sorgu daha önce sisteme kaydedilmemişse, OYSA sorgunun tipine göre bir sorgu artifaktı oluşturmaktadır. Her farklı SPARQL sorgu tipi için bu sorgu tipini işleyebilen özel bir sorgu artifaktı tipi bulunmaktadır. Şekil 4.3'den de görüldüğü gibi bu SPARQL sorgu tipleri; "select", "construct", "describe" ve "ask" sorgu artifaktlarıdır. OYSA oluşturduğu artifaktın kimliğini etmene ilettikten sonra izlenen sorguyu, bu sorguyu izleyen artifaktın ve etmenin kimliklerini kaydetmektedir.

OYSA'nın oluşturduğu ve herbiri ilgili SPARQL sorgusunun sonuçlarındaki değişimleri takip eden sorgu artifaktlarının izledikleri sorgular birleşik ("federated") SPARQL sorguları olabileceğinden, artifaktların ilklenmeleri



Şekil 4.3: A&A meta-modelindeki "artifact" sınıfının özelleştirilmesi ile oluşturulan anlamsal artifaktlara ilişkin sıradüzen

sırasında sorgular öncelikle WODQA kullanılarak alt sorgulara ayrılmaktadırlar. Sorgu artifaktları elde ettikleri her alt sorgu için bir alt sorgu artifaktı ("SubQueryArtifact") oluşturmakta ve bu artifaktlara kaydolmaktadırlar. Alt sorgu artifaktları, izledikleri alt sorguları bu sorguların hedeflediği veri kümelerinin DaDy üstverilerinde yer alan periyotla çalıştırmakta ve elde ettikleri güncel sonuçları önceki periyotta elde ettikleri sonuçlarla karşılaştırmaktadırlar. Alt sorgulara gelen sonuçlar sonuç kümesi ("resultset") formatındadır. Elde edilen son sonuç kümesi ile bir önceki periyotta elde edilen sonuç kümesi arasında fark olması durumunda alt sorgu artifaktları kendilerinde kayıtlı olan sorgu artifaktlarının "monitorResults" operasyonunu çalıştırmaktadırlar. "MonitorResults" operasyonu çalışan sorgu artifaktları takip ettikleri sorgunun altsorgularını izleyen altsorgu artifaktlarından güncel sonuçları alıp WODQA aracını kullanarak birleştirmekte ve gelen sonuçlarla operasyonun önceki çalışması sırasında elde ettikleri sonuçlarla karşılaştırmaktadırlar. "Select" sorgu artifaktlarına gelen sonuçlar sonuç kümesi formatında olduğundan, "ask" sorgusu artifaktlarına gelen sonuçlar da mantıksal doğru ya da yanlış değerleri olduğundan sonuçların karşılaştırılması için özel bir yazılımsal araca gerek duyulmamaktadır. Buna karşın "construct" sorgusu artifaktları ve "describe" sorgusu artifaktlarının izledikleri sorgu tipleri RDF çizgesi formatında sonuçlar döndürdüğünden sonuç karşılaştırması için daha önce üçüncü bölümde söz edilen OWLDIFF aracını kullanmaktadırlar. OWLDIFF, RDF çizgelerindeki ve OWL ontolojilerindeki farkları bulabilen bir araçtır. Bu araç tarafından bulunan farklar yine üçüncü bölümde söz edilen ve proje kapsamında bazı eklemeler ile genişletilen ODO (OWL için Değişim Ontolojisi- "Change

Ontology for OWL") terminolojisi ile oluşturulmuş değişiklik türlerini betimleyen ek açıklamalar konularak bir fark çizgesi haline getirilmektedir.

Sorgu sonuçları değiştiğinde sorgu artifaktlarının gözlemlenebilir sonuç özelliğinin değeri değiştirilmektedir. "Select" sorgusu artifaktlarında bu değer yeni sonuç kümesi, "ask" sorgusu artifaktlarında yeni mantıksal sonuçtur. "Construct sorgusu artifaktları ve "describe" sorgusu artifaktlarında ise gözlemlenebilir sonuç özelliğinin değeri fark çizgesidir.

Sorgu artifaktlarının ilklenmeleri sırasında gerçekleştirilen bir başka işlem de sorguların kullandıkları önek ("prefix") tanımlamalarında yer alan her ontoloji için bir ontoloji artifaktının oluşturulması ya da ilgili ontolojideki değişimlerin izlenmesi için sistemde önceden bir ontoloji artifaktı oluşturuldu ise bu artifakta kaydolunmasıdır. Oluşturulan her ontoloji artifaktı kullandığı ("import" ettiği) her ontoloji için ya bir ontoloji artifaktı oluşturmakta ya da eğer ilgili ontoloji için bir artifakt mevcut ise bu artifakta kaydolmaktadır.

Ontoloji artifaktları izledikleri ontolojiyi periyodik olarak anlamsal veb üzerindeki adresinden indirmekte ve OWLDIFF aracını kullanarak karşılaştırmaktadırlar. Ontoloji versiyonlarının karşılaştırılması sırasında iki versiyon arasında oluşabilecek mantıksal farkların da yakalanması amacı ile OWLDIFF aracına Pellet çıkarsama motoru entegre edilerek çıkarsama desteği sağlanmıştır. Karşılaştırma sonucu elde edilen farklar ODO ontolojisi kullanılarak açıklanmış bir fark çizgesi olarak ifade edilmekte ve gözlemlenebilir farkÇizgesi (diffGraph) özelliğinin değerini oluşturmaktadır. Bir ontolojide mantıksal ya da yapısal düzeyde bir fark oluştuğunda bu bir iç ontoloji değişim olayı olarak adlandırılmakta ve bu olaya tekil bir olay etiketi atanmaktadır. Oluşan değişim olayı, ontoloji değişimi bildirim artifaktı'nın doğrudan ontoloji değişimi bildirimi (notify direct ontology change) operasyonu çalıştırılıp bu operasyona parametre olarak değişen ontolojinin URI'si ve değişim olayının etiketi geçirilmektedir. Zincirleme ontoloji değişimlerinin takibi de bu proje kapsamında gerçekleştirilen hedeflerden bir diğeridir. Zincirleme ontoloji değişimi takibinde amaç değişen bir ontolojiyi kullanan ("import" eden) diğer ontolojilerde oluşabilecek farklar hakkında da sistem yöneticisini uyarabilmektir. Doğrudan ontoloji değişimi olayı iki tür operasyonun çalıştırılması ile gerçekleşmektedir. İlk operasyon, ontoloji artifaktında kayıtlı olan ve değişen ontolojiyi kullanan diğer ontolojileri takip eden ontoloji artifaktlarının dış ontoloji değişimi bildirimi (notifyExternalOntologyChange) operasyonlarıdır. Diğer operasyon ise yine değişim geçiren ontolojiyi izledikleri sorguda kullanan ve dolayısıyla bu ontolojiyi izleyen artifaktta kayıtlı olan sorgu artifaktlarının ontoloji değişimi bildirim

```
PREFIX travel:<http://www.owl-ontologies.com/travel.owl>
SELECT ?subject WHERE
{?subject <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
travel:BudgetAccomodation}
```

Şekil 4.4: Ontoloji takibi durum çalışmasında izlenen sorgu

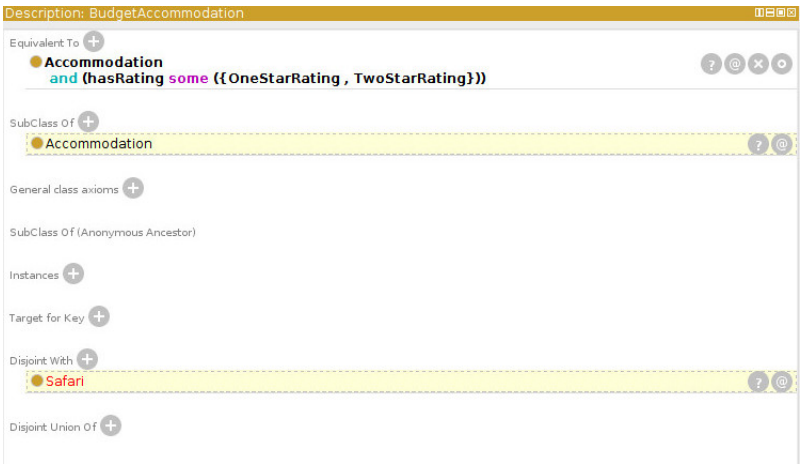
(ontologyChangeNotification) operasyonlarıdır. Bu operasyonlara oluşan olayın tekil etiketi ve değişen ontolojinin URI'si parametre olarak geçirilmektedir. Sahip oldukları dış ontoloji değişimi bildirimi operasyonları çalıştırılan ontoloji artifaktları, değişen ontolojiyi kullanmaları nedeniyle kendilerinde oluşmuş olabilecek mantıksal değişiklikler ya da hataların kendilerini kullanan sorguların sonuçlarını etkileme olasılıklarına karşı ontoloji değişim bildirim artifaktının dolaylı ontoloji değişimi bildirim (notifyIndirectOntologyChange) operasyonunu çalıştırmaktadırlar. Bu operasyon, değişim olayının etiketi, kendi URI'si ve zincirleme ontoloji değişimini tetikleyen ontolojinin URI'sini parametre olarak geçirerek çalıştırılmaktadır.

Dolaylı ontoloji değişiminden etkilenen ontoloji artifaktları, bu durumu kendilerini kullanan sorgu artifaktlarının dış ontoloji değişimi bildirim (notifyExternalOntologyChange) operasyonlarını değişim olayının etiketi, kendi URI'si ve zincirleme ontoloji değişimini tetikleyen ontolojinin URI'sini parametre olarak geçirerek çalıştırıp ilgili sorgu artifaktlarına bildirmektedirler. Ontoloji değişimi bildirim operasyonları çalışan sorgu artifaktları ise ontoloji değişimi bildirim artifaktının ontoloji değişimi bildirim (notifyOntologyChange) operasyonunu değişim olayının etiketi, zincirleme ontoloji değişimini tetikleyen ontolojinin URI'sini ve izlediği sorguyu geçirerek çalıştırmaktadırlar. Tüm bu değişim bildirimlerini alan ontoloji değişimi bildirim artifaktı ise elde ettiği her bildirimi çalışma alanının bağlı olduğu yazılıma (etmen) sinyal göndererek bildirmektedir. Böylelikle etmen sisteminin yöneticisinin durumdan haberdar edilmesi sağlanmaktadır ve eğer ontoloji değişimlerinden etkilenen sorgular var ise bu sorguların güncellenmesi gerekliliğinin gözden kaçırılması engellenmektedir.

4.2.3 Ontoloji takibini sınamaya yönelik durum çalışması

Tez kapsamında geliştirilen sistemin ontoloji takibi işlevinin sınanması amacı ile Seyahat Ontolojisi ("Travel Ontology")[1] kullanıldığı bir durum çalışması geliştirilmiştir.

[1] http://protege.cim3.net/file/pub/ontologies/travel/travel.owl

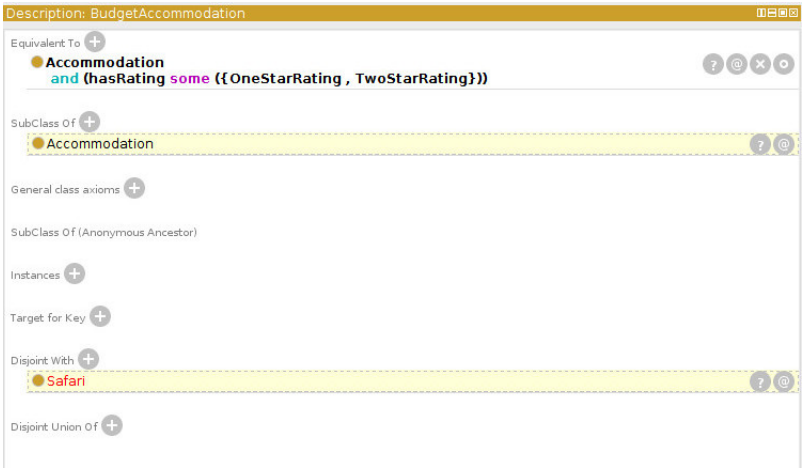


Şekil 4.5: Orijinal Seyahat Ontolojisi'nin Protege Ontoloji Editörü'ndeki görüntüsü

Geliştirilen durum çalışmasında kullanıcısı için tatilde konaklayabileceği ekonomik konaklama mekanı seçeneklerini araştıran bir etmenin önek ("prefix") kısmında seyahat ontolojisini içeren ve Şekil 4.4'te gösterilen sorguyu sisteme kaydettirdiği varsayılmıştır. Bu sorgu Seyahat Ontolojisi'nde "BudgetAccomodation" adı ile tanımlanan ekonomik konaklama mekanlarını bulmayı amaçlamaktadır. "BudgetAccomodation" sınıfı orijinal ontolojide bir yada iki yıldıza sahip konaklama mekanı ("Accommodation and (hasRating some ({OneStarRating , TwoStarRating}))") olarak tanımlanmıştır (bkz. Şekil 4.5). Durum çalışmasında orijinal ontolojiye bir de "Hotel" sınıfının bir örneği olan Hotel1 eklenmiştir. Sorgunun ontolojinin orijinal sürümü üzerinde çalıştırıldığı ilk durumda ontolojide hiç "BudgetAccomodation" sınıfı örneği bulunmadığından ve Hotel1 de üç yıldızlı tanımlandığı için boş sonuç kümesi dönmektedir.

Durum çalışmasının ikinci kısmında "BudgetAccomodation" sınıfının tanımlaması (bkz. Şekil 4.6) değiştirilmiştir. Yapılan değişikliğin ardından "BudgetAccomodation" sınıfı bir, iki ya da üç yıldıza sahip konaklama mekanı olarak ("Accommodation and (hasRating some ({OneStarRating , TwoStarRating, ThreeStarRating}))") tanımlanmıştır. Yapılan bu sözdizimsel değişiklik Hotel1 örneğinin bir "BudgetAccomodation" örneği olarak çıkarsanmasına neden olmaktadır. Kaydedilen sorgu ontolojinin eski versiyonunda bir ya da iki yıldızlı konaklama mekanlarını "BudgetAccomodation" olarak döndürürken, yeni versiyonda üç yıldızlı mekanlarda "BudgetAccomodation" örneği olarak döndürülmektedir. Sorguyu kaydettiren etmenin kullanıcısı için bu durum uygun olmayabilir zira üç yıldızlı mekanlar diğerlerinden daha yüksek ücretli olacaktır. Dolayısı ile ontolojide oluşan değişikliğin sorgu üzerindeki etkisinin etmen sistemine bildirilmesi gerekmektedir.

Şekil 4.4'te verilen SPARQL sorgusu sisteme kaydedildiğinde bu sorguyu



Şekil 4.6: Değiştirilmiş Seyahat Ontolojisi'nin Protege Ontoloji Editörü'ndeki görüntüsü

izlemesi için bit SELECT sorgusu artifaktı yaratılmıştır. Bu SELECT sorgusu artifaktı tarafından yaratılan ontoloji artifaktı sorgunun önek kısmında yer alan Seyahat Ontolojsi'ni değişimler için izlemeye başlamıştır. Ontoloji üzerinde değişiklik yapılmasının ardından, ontoloji artifaktı Şekil 4.7'de gösterilen gözlenebilir değişim ontolojisini oluşturmuş ve gerçekleştirilen değişimi kendisini yaratan SELECT sorgusu artifaktı ile Ontoloji Değişim Bildirim Artifaktı'na bildirmiştir. SELECT sorgusu artifaktı da Seyahat Ontolojisi'ndeki değişimden etkilenmiş olabileceğini Ontoloji Değişim Bildirim Artifaktı'na bildirmiştir. Ontoloji Değişim Bildirim Artifaktı ise kendisine gelen bu değişim uyarılarını etmen sistemindeki yönetici etmene bildirerek değişim iletimini tamamlamıştır.

4.2.4 Anlamsal ortamın başarımını sınamaya yönelik durum çalışmaları

Tez kapsamında geliştirilen sistemin, sonuçları uluslararası makale çalışmalarında kullanılmak üzere anlamsal ortamın başarımını sınamaya yönelik iki durum çalışması geliştirilmiştir. Literatürde CArtAgO ile geliştirilmiş ortamların başarımını ölçen ve değerlendiren bir çalışma bulunmamaktadır. Bu durum çalışmaları bu konudaki ilk çaba olarak nitelendirilebileceği için çalışmanın başlangıcında değerlendirme sürecinin perspektifleri belirlenmiştir:

- **Perspektif 1:** Sistemin şu temel gereksinimleri karşılayıp karşılamadığı belirlenmelidir:
  - Sisteme izlemesi için birden fazla SPARQL sorgusu kaydedilebilmelidir.

```
<rdf:RDF
xmlns:
rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:
owl="http://www.w3.org/2002/07/owl#"
xmlns:
rdfs="http://www.w3.org/2000/01/rdf-schema#"
xmlns:
change="http://omv.ontoware.org/2009/09/OWLChanges#"
xmlns:
xsd="http://www.w3.org/2001/XMLSchema#" >
<rdf:Description
rdf:about="http://www.deneme.org/ChangeOntology.owl">
        <owl:imports
        rdf:resource=
        "http://omv.ontoware.org/2009/09/OWLChanges.owl"/>
        <rdf:type
        rdf:resource=
        "http://www.w3.org/2002/07/owl#Ontology"/>
</rdf:Description>
<rdf:Description
rdf:about="http://www.deneme.org/ChangeOntology.owl#change1">
        <change:hasRelatedClass
          rdf:resource=
        "http:// travel.org/travel.owl#BudgetAccommodation"/>
        <change:hasRelatedIndividual
        rdf:resource="http:// travel.org/travel.owl#Hotel1"/>
        <rdf:type
        rdf:resource="http://omv.ontoware.org/2009/09/
        OWLChanges#removeTypeSpecification"/>
</rdf:Description>
</rdf:RDF>
```

Şekil 4.7: Seyahat Ontolojisi'nde gerçekleşen değişimin ifade edildiği değişim ontolojisi

  - Kaydedilen tüm SPARQL sorgularının sonuçları izlenmeli ve sonuçlarında herhangi bir değişiklik olması durumunda etmenlere gerekli uyarı sağlanabilmelidir. Sorgu sonuçlarının takip edilebilmesi federe SPARQL sorgularının çalıştırılmasını da içerdiğinden sorgu servisinin ayrıca değerlendirilmesine gerek yoktur.
  - Sisteme yeni tanımlanmış VoID ya da DaDy tabanlı üstveri eklenebilmeli (annotation) ve kullanılmayan üstveri sistemden silinebilmelidir.

- **Perspektif 2:** Sistemin fiziksel sınırları zorlanarak verilen bir zaman aralığında kaç tane farklı SPARQL sorgusu izleyebileceği bulunmalıdır.
- **Perspektif 3:** Altsorgu izleme yaklaşımının sistem çıktısı üzerindeki etkisi değerlendirilmelidir.

Sistemin değerlendirilmesinde bir sonraki aşama Bağlı Veri'nin tüketileceği organizasyonel bağlamın belirlenmesidir. Organizasyonel bağlam bir organizasyon dahilinde çalışan Bağlı Veri uygulamalarının kullandığı veri kümelerinin ve bu veri kümelerini tanımlayan üstverinin tanımlanmasını gerektirmektedir. Organizasyonel bağlam için minimum düzenleme tek bir Bağlı Veri uygulamasını ve bu uygulamanın ihtiyaç duyduğu Bağlı Veri kümeleri ile bunları tanımlayan VoID ve DaDy üstverilerini içermelidir. Bu minimum düzenlemeyi desteklemek amacıyla finans alanında çalışan bir yatırım şirketinde kullanılan bir Bağlı Veri uygulaması örnek durum olarak ele alınmıştır. Bu Bağlı Veri uygulamasının gereksinimleri şöyle listelenebilir:

- Uygulamanın kullanıcıları önceden tanımlanmış bir şirket kümesinin hisse senetlerinin değerlerindeki değişimleri takip edebilmelidir.
- Uygulamanın kullanıcıları New York Times gazetesinde önceden tanımlanmış bir şirket kümesi hakkında yeni haber çıkıp çıkmadığını takip edebilmelidir.
- Uygulama takip edilen bir şirketin hisse senedinin değeri değiştiğinde ya da bu şirket hakkında New York Times gazetesinde yeni bir haber çıktığında uyarı göndermelidir.

Bu Bağlı Veri uygulaması için organizasyonel bağlam New York Times, Borsa ve DBPedia veri kümelerini ve bu veri kümelerini tanımlayan VoID tanımlamalarını içermektedir. Başarımın değerlendirilmesi sırasında gerçek New York Times ve

```
SELECT ?sameCompany ?count ?value
WHERE {
<http://dbpedia.org/resource/X>
<http://www.w3.org/2002/07/owl#sameAs>
?sameCompany .
?sameCompany
<http://data.nytimes.com/elements/associated_article_count>
?count .
?sameCompany <http://stockmarket.com/elements/stockValue>
?value}
```

Şekil 4.8: Deneyler sırasında kullanılan SPARQL sorgusu

DBPedia veri kümeleri Bağlı Veri ağından indirilerek yerel olarak kullanılmış, Borsa veri kümesi ise bu durum çalışması için özel olarak oluşturulmuştur. Dolayısıyla bu üç veri kümesi durum çalışmasının gerçekleştirildiği deney ortamını meydana getirmiştir. Deneyde ayrıca her veri kümesi için ilgili kümenin değişim sıklığını belirleyen bir DaDy değişim frekansı üstverisi oluşturulmuştur. Veri kümeleri ve değişim sıklıkları şöyle düzenlenmiştir:

- DBPedia veri kümesinin DaDy değişim sıklığının "NoUpdates" (GüncellemeYok) olduğu yani hiç değişmediği varsayılmıştır. Bu değişim sıklığının seçilmesinin nedeni DBPedia veri kümesinden sadece şitket URI'sinin alınması ve deneyler sırasında şirket isimlerinin değişmeyeceğinin varsayılmasıdır.
- New York Times veri kümesinin değişim sıklığının "MidFrequent" (OrtaSıklıkta) olduğu varsayılmıştır. Deneysel nedenlerle bu sıklık on dakikaya karşılık gelecek şekilde ayarlanmıştır.
- Borsa veri kümesinin değişim sıklığının "HighFrequent" (YüksekSıklıkta) olduğu varsayılmıştır. Deneysel nedenlerle bu sıklık üç dakikaya karşılık gelecek şekilde ayarlanmıştır.

Deneyler sırasında kullanılan SPARQL sorgusu ise Şekil 4.6'da görülebilir. Bu sorgu DBPedia URI'si verilen şirket için New York Times veri kümesinden haber sayısını, Borsa veri kümesinden ise hisse senedi değerini çekmektedir.

Uygulamanın gerçekçi olması amacı ile DBPedia ve New York Times veri kümeleri üzerinde hem DBPedia'da kaydı olan hem de New York Times gazetesinde hakkında haber çıkan şirketlerin yayınlanan haber sayılarını elde eden bir sorgu çalıştırılmıştır. Sorgu sonucunda 484 şirkete ait kayıtlar dönmüştür. Bu

kayıtlara 16 varsayımsal şirketin daha eklenmesi ile toplam 500 şirketlik bir deney kümesi oluturulmuştur. Bu 500 şirketlik küme üzerinde yukarıda verilen sorgu kullanılarak iki deney gerçekleştirilmiştir. Deneyler sırasında New York Times ve Borsa veri kümeleri bu amaçla gerçeklenmiş olan Java uygulamaları tarafından periyodik olarak güncellenmiştir.

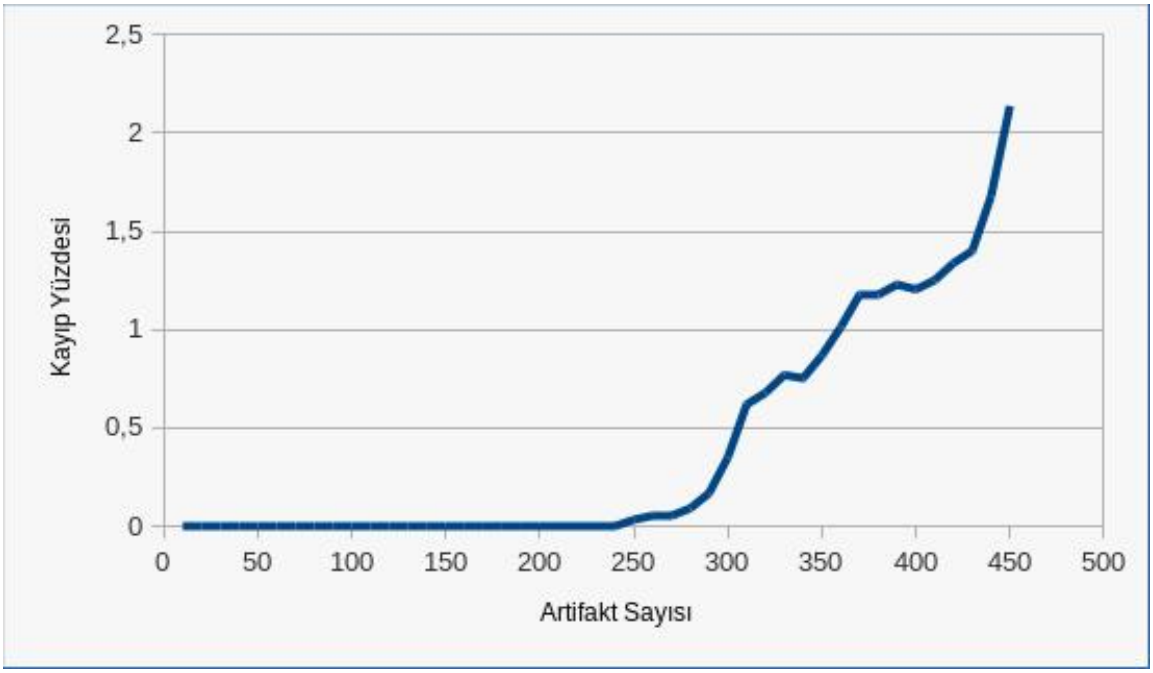
Deneylerin gerçekleştirildiği makinelerin konfigürasyonları ise şöyledir:

- Makine-1: 8 GB RAM, AMD FX-6100 6 core 1.4 GHz işlemci, Ubuntu 13.04 işletim sistemi
- Makine 2: 8 GB RAM, Intel Core i7 8 core 3.4 GHz işlemci, Ubuntu 12.04 işletim sistemi
- Makine 3: 12 GB RAM, Intel Core i7 8 core 3.2 GHz işlemci, Fedora 18 işletim sistemi

Makine 1 ve Makine 2'nin üzerine Virtuoso 7 RDF deposu kurulmuştur. Borsa veri kümesi, bu kümeye bağlı olan SPARQL uç noktası ve veri güncelleme uygulaması Makine 1 üzerinde çalıştırılmıştır. Diğer veri kümeleri, bu veri kümelerinin SPARQL uç noktaları ve New York Times veri kümesini güncelleyen uygulama Makine 2 üzerinde çalıştırılmıştır. CartAgO çalışma alanı ise Makine 3 üzerinde çalıştırılmıştır.

Deney 1

İlk deneyde izlenecek her yeni şirket için yeni bir "select" sorgusu artifaktı yaratılmıştır. Her "select" sorgusu artifaktı yukarıda verilen federe sorguyu hedeflediği üç veri kümesinin üzerinde çalıştırmıştır. Bu deneyin doğru çalışmış olması sistemin işlediğini ve çok sayıda federe sorguyu çalıştırıp izleyebileceğini, dolayısıyla birinci değerlendirme perspektifini doğrulamıştır. Bu deneyin doğru çalışmış olması sisteme yeni eklenecek veri kümeleri hakkındaki VoID ve DaDy üstverilerinin tanımlanmasının desteklendiğini de kanıtlamaktadır. Diğer yandan, sistemin aynı anda kaç farklı SPARQL sorgusunu izleyebileceği yani sistemin fiziksel limitleri ile ilgili olan ikinci perspektifin doğrulanması için, sorguları izleyecek yeni artifaktlar yaratılamayana kadar sistem her iki dakikada bir yeni bir federe sorgu kaydederek çalıştırılmıştır. Sistem 250. artifakt yaratılana kadar kayıpsız şekilde çalışmaya devam etmiştir. 250. artifakttan sonra sistem gerçekleşen bazı veri değişimlerini yakalayamamaya dolayısıyla istemci etmeni bu değişimler hakkında uyaramamaya başlamıştır. Sistem 450. artifakttan sonra yeni



Şekil 4.9: Deney 1 sırasındaki artifakt sayısı artışı karşısında oluşan kayıp yüzdesini gösteren grafik

artifakt yaratamamıştır. Deneyin sonundaki kayıp oranı %2,14'tür. Sistemi değişim kaybı grafiği Şekil 4.7'de görülebilir.

İlk deneyde sistem oluşan veri değişikliklerinin yaklaşık %98'ini yakalamıştır ancak 450 artifakt yaratıldıktan sonra yeni artifaktlar yaratılamamış ve yeni sorgular sisteme kaydedilememiştir. Bunun iki temel nedeni vardır:

- İlk olarak, federe sorgular alt sorgularına ayrılmadan iki dakikada bir çalıştırılmış, farklı veri kümelerini hedefleyen alt sorgular kendi veri kümelerinin değişim sıklıklarına uygun şekilde farklı periyotlarla çalıştırılmamıştır. Bu da sisteme ek yük getirmiştir.
- İkinci olarak ise, sistemin geliştirilmesi için kullanılan CartAgO artifaktları kendi zamanlamasına ("scheduling") göre sıraladığı çoklu iş parçacıkları ("multi-thread") halinde çalıştırmaktadır. Çoklu iş parçacıkları içeren bu sürecin ("process") üzerindeki ağır yük zamanlama mekanizmasını etkilemektedir. CartAgO çalışma alanlarının farklı makineler üzerine dağıtılması ile sorunun çözülebileceği düşünülmektedir, fakat CartAgO sistemi çalışma alanlarını desteklememektedir. Bunun yerine, birden fazla çalışma alanından oluşan bir CartAgO ortamı birden fazla makine üzerine dağıtılabilmektedir. Dolayısıyla, olası bir çözüm bir organizasyonel bağlı veri çalışma alanının birden fazla çalışma alanını içeren bir CartAgO ortamı içinde birden fazla makine üzerine dağıtılması olabilir.

```
Alt sorgu 1:
        SELECT * WHERE
        {<http://dbpedia.org/resource/X>
        <http://www.w3.org/2002/07/owl#sameAs>
        ?sameCompany}
Alt sorgu 2:
        SELECT * WHERE
        {?sameCompany
        <http://data.nytimes.com/elements/
        associated_article_count>
        ?count FILTER (?sameCompany=
        <http://data.nytimes.com/7515178947949973582>)}
Alt sorgu 3:
        SELECT * WHERE {?sameCompany
        <http://stockmarket.com/elements/stockValue>
        ?value FILTER ?sameCompany=
        <http://data.nytimes.com/7515178947949973582>)}
```

Şekil 4.10: Deneyler sırasında kullanılan federe sorgunun alt sorguları

Deney 2

Bu deneyde alt sorgu izleme yaklaşımının doğrulanmasını amaçlayan üçüncü değerlendirme perspektifi ele alınmıştır. Deney sırasında ilk deneyde kullanılan federe sorgu kullanılmış, ancak sorgu sonuçlarının izlenmesi asıl mimaride kullanılan "select" sorgusu artifaktları kullanılarak gerçekleştirilmiştir.

Tezin önceki bölümünde de belirtildiği üzere asıl mimarideki sorgu artifaktları federe sorguları ayrıştırıp alt sorgularını belirlemekte ve her alt sorguyu takip edecek alt sorgu artifaktları yaratmaktadırlar. Deney sırasında kullanılan federe sorgunun alt sorguları Şekil 4.8'de görülebilir. Bu alt sorgulardan ilki yerel DBPedia veri kümesi SPARQL uçnoktası üzerinde, ikincisi yerel New York Times veri kümesi SPARQL uçnoktası üzerinde, üçüncüsü ise yerel Borsa veri kümesi SPARQL uçnoktası üzerinde çalışmaktadır.

Durum çalışmasında kullanılan DBPedia veri kümesi güncellenmediğinden, bu deneyde yaratılan "select" sorgusu artifaktları DBPedia SPARQL uçnoktası üzerinde çalışan alt sorgu artifaktları yaratmamıştır. Diğer yandan, her "select" sorgusu artifaktı ikinci ve üçüncü alt sorgular için birer alt sorgu artifaktı yaratmış ve bu artifaktlara kaydolmuştur. Yaratılan alt sorgu artifaktları ilişkili oldukları SPARQL uçnoktalarını New York Times ve Borsa veri kümelerinin DaDy üst verilerinde tanımlı olan sıklıklarla sorgulamıştır. Buradaki en önemli nokta bir federe sorguya ait olan alt sorguların aynı anda çalıştırılmamış olmasıdır. Sadece verilen DaDy değişim periyotlarına göre gerekli olan alt

sorgular çalıştırılmaktadır. Herhangi bir alt sorgunun sonucu değiştiğinde ilişkili olduğu alt sorgu artifaktı kendisinde kayıtlı olan "select" sorgusu artifaktlarının "monitorResults" operasyonunu çalıştırmıştır. Bu operasyon bir federe sorgunun tüm alt sorgularından gelen sonuçları WODQA'nın sorgu sonucu birleştirme mekanizmasını kullanarak birleştirmiştir. Daha sonra elde edilen sonuç federe sorgunun bir önceki sonucu ile karşılaştırılmıştır. Karşılaştırma sırasında fark bulunması durumunda "select" sorgusu artifaktının gözlenebilir sonuç özelliği değiştirilmiş, böylelikle bu artifakta odaklanan etmenlerin değişiklikten haberdar olmaları sağlanmıştır.

Bu deneyde sistem 356. "select" sorgusu artifaktı yaratılana kadar hiçbir değişikliği kaybetmemiştir. 356. "select" sorgusu artifaktının yaratılmasından sonra sistem yeni artifakt yaratamamış, sonuç olarak sistem 356 şirketin verileri izleyebilmiştir. Her "select" sorgusu artifaktı için iki alt sorgu artifaktı yaratıldığından aslında izlenen her şirket için sistemde üç artifakt çalışmıştır. Dolayısıyla, bu deneyde çalıştırılan artifakt sayısının gerçek limiti yaklaşık 1000 (1068) artifakttır.

Sistem maksimum artifakt sayısına ulaştığında elde edilen kayıp oranı %0'dır. Çıktıdaki ("throughput") bu artışın nedeni olarak tüm federe sorgunun yerine alt sorguların ilişkili oldukları veri kümelerinin sıklıklarına göre çalıştırılması gösterilebilir. Öte yandan, sistemin izleyebildiği federe sorgu sayısı azalmıştır. Bu durum da alt sorgu artifaktlarının çalıştırılmasının ve alt sorgu sonuçlarının birleştirilmesinin sisteme getirdiği ek yükle açıklanabilir.

Deney 3 Sistemin ölçeklenebilirliği hakkında daha net bir bilgiye sahip olabilmek amacı ile deney 2 veri kümelerinin farklı sıkllıklarda değiştikleri varsayılarak üç kez daha tekrarlanmıştır. Bu deneyde kullanılan makinaların hepsinde Ubuntu 14.04 işletim sistemi çalıştırılmış olup konfigürasyonları şöyledir:

- Makine-1: 12 GB RAM, Intel Core i7 8 core 2.8 GHz işlemci
- Makine 2: 8 GB RAM, Intel Core i7 8 core 3.4 GHz işlemci
- Makine 3: 12 GB RAM, Intel Core i7 8 core 3.2 GHz işlemci

Makine 1 ve Makine 2'nin üzerine Virtuoso 7 RDF deposu kurulmuştur. Borsa veri kümesi, bu kümeye bağlı olan SPARQL uç noktası ve veri güncelleme uygulaması Makine 1 üzerinde çalıştırılmıştır. Diğer veri kümeleri, bu veri kümelerinin

Verisetlerinin Değişim Aralıkları			Artifakt Yaratma Sıklığı	Maksimum Artifakt Sayısı
Borsa Veriseti	NYTimes Veriseti	DBPedia Veriseti		
2.5	5	-	2.5	307
3.75	7.5	-	3.75	402
5	10	-	5	432

Şekil 4.11: Deney 3'te verisetlerinin değişim aralıkları, artifakt yaratma sıklığı ve ulaşılan maksimum artifakt sayısı

SPARQL uç noktaları ve New York Times veri kümesini güncelleyen uygulama Makine 2 üzerinde çalıştırılmıştır. CartAgO çalışma alanı ise Makine 3 üzerinde çalıştırılmıştır.

Yapılan deneylerdeki verisetlerinin değişim aralıkları, artifakt yaratma sıklığı ve ulaşılan maksimum artifakt sayıları Şekil 4.11'de görülebilir.

Yapılan deneylerin tümünde yakalanamayan değişiklik sayısı sıfırdır. Sorgu sıklığının artması ve sorguların Virtuoso RDF deposu üzerinde çalışmasını sağlayan VirtJena yazılımı üzerindeki yükün artması nedeni ile Borsa Veriseti'nin 2.5, NYTimes Veriseti'nin ise 5 dakika ara ile güncellendiği durumda artifakt sayısı 307'de kalmıştır. Diğer iki deneyde verisetlerinin değişim araları arttıkça ulaşılan maksimum artifakt sayısı da artmıştır. Borsa Veriseti'nin 3.75, NYTimes Veriseti'nin 7.5 dakika ara ile güncellendiği durumda 402 artifakta, Borsa Veriseti'nin 5, NYTimes Veriseti'nin 10 dakika ara ile güncellendiği durumda 432 artifakta ulaşılmıştır. Deneylerin çalıştırılması sırasında ağ trafiğinin yoğun olduğu saatlerde yukarıda verilen maksimum artifakt sayılarına ulaşılamadan deneyin sonuçlandığı durumlar olduğundan, deney sonuçlarını ağ üzerindeki trafiğin yoğunluğunun da etkilediği düşünülmektedir.

5 TARTIŞMA VE GELECEK ÇALIŞMALAR

Bağlı Veri Ağı ve Anlamsal Veb her geçen gün giderek büyümektedir. Dahası, Anlamsal Veb'deki ontolojiler ve Bağlı Veri Ağı'ndaki veri kümeleri üzerinde herhangi bir merkezi kontrol bulunmamakta, veri ve ontolojiler herhangi bir zamanda değişebilmekte hatta silinebilmektedirler. Bu kaynakların en önemli kullanıcısı durumundaki etmenler oluşabilecek bu ani değişikliklerden olumsuz yönde etkilenmektedirler. Zira, doğaları gereği otonom olan ve çevrelerinde oluşan değişimlere tepki vermesi gereken etmenlerin bu değişimlerden haberdar olmaları, dolayısıyla Anlamsal Veb ve Bağlı Veri Ağı üzerindeki veriyi sadece sorgulamaları değil aynı zamanda bu veri kaynaklarında oluşacak değişimleri de izlemeleri gerekmektedir. Bu gereksinim de hem etmen sistemlerinin çalışma zamanındaki yükünü arttırmakta, hem de etmen geliştiricilerine sahip olmaları gereken bilgi birikimi ve gerçekleştirilecek servisler açısından ek yük getirmektedir.

Giriş bölümünde de belirtildiği üzere bu tez çalışmasındaki temel hedef etmen geliştiricilerini ve etmen sistemlerini Anlamsal Veb ve Bağlı Veri Ağı'ndaki değişimlerin izlenmesinin getirdiği karmaşıklıktan soyutlamaktır. Sistemin önceki bölümde tartışılan test ve performans ölçümlerinin sonucunda tez çalışmasının başında belirlenen bu hedefe ulaşıldığı görülmüştür. Geliştirilen altyapı etmenlerin Anlamsal Veb ve Bağlı Veri Ağı ile olan etkileşimlerini tamamen soyutlamaktadır. Etmenler sorgulama ve verideki değişimlerin izlenmesi gereksinimlerini tümüyle geliştirilen altyapının sağladığı hizmetleri kullanarak karşılayabilmektedirler. Literatürde etmen geliştiricisini ve etmen yazılımlarını Anlamsal Veb ve Bağlı Veri Ağı'nın karmaşıklığından koruyan bilinen başka bir soyut ya da somut model yoktur.

Gerçekleştirilen sistem literatüre diğer katkıları ise şunlardır:

- Etmen sistemlerinin Anlamsal Veb ve Bağlı Veri uzayı ile iç kökenli bir ortam üzerinden etkileşmelerini sağlayan ilk çalışmadır.

- Federe sorguların itme-çekme tabanlı izlenmesini olanaklı kılan ilk sistem çalışmasıdır. Literatürde, geliştirilen sisteme benzer sorgu tabanlı itme-isteme yaklaşımını kullanan sistemler (Passant and Mendes, 2010; Popitsch and Haslhofer, 2011) bulunmasına karşın, bu sistemler federe sorguları ve sorguların bağlı oldukları ontolojilerdeki değişimleri izleyememektedirler.

Geliştirilen sistem kavramsal hedeflerini karşılıyor olsa da önceki bölümdeki performans ölçümlerinin değerlendirilmesinde de belirtildiği üzere ölçeklenebilirliği düşüktür. Bunun olası nedenlerinden biri sistemin altyapısını oluşturan CartAgO sisteminin prototip geliştirme amaçlı olması (Ricci et al., 2007) dolayısıyla da ölçeklenebilirliğinin düşük olmasıdır. Bu nedenle gelecek çalışmalar içinde ilk sırayı önerilen sistemin CartAgO altyapısı kullanılmadan, daha ölçeklenebilir bir sürümünü gerçekleştirmek almaktadır. Bir diğer planlanan çalışma ise sistemin etmen ortamları içinde verilmesi gereken ontoloji eşleme, veri kökeni ("provenance"), arama, Bağlı Veri Ağı üzerinde gezinerek ("crawling") bilgi depolama gibi hizmetleri de verecek şekilde genişletilmesidir.

KAYNAKLAR DİZİNİ

Akar, Z., Halaç, T. G., Ekinci, E. E., and Dikenelli, O., 2012, Querying the web of interlinked datasets using VOID descriptions, in *WWW2012 Workshop on Linked Data on the Web, Lyon, France, 16 April, 2012*

Alexander, K. and Hausenblas, M., 2009, Describing linked datasets-on the design and usage of void, the'vocabulary of interlinked datasets, in *In Linked Data on the Web Workshop (LDOW 09), in conjunction with 18th International World Wide Web Conference (WWW 09)*

Antoniou, G., d'Aquin, M., and Pan, J. Z., 2011, Semantic web dynamics, *J. Web Sem.* **9(3)**, 245 pp.

Berners-Lee, T. and Connolly, D., 2004, *Delta: an ontology for the distribution of differences between RDF graphs*

Berners-Lee, T., Hendler, J., and Lassila, O., 2001, The semantic web, **284(5)**, 34 pp.

Bizer, C., 2009, The emerging web of linked data, *Intelligent Systems, IEEE* **24(5)**, 87 pp.

Bizer, C., Heath, T., and Berners-Lee, T., 2009, Linked data - the story so far, *Int. J. Semantic Web Inf. Syst.* **5(3)**, 1 pp.

Borst, W., 1997, *Construction of Engineering Ontologies*, *Ph.D. thesis*, Centre of Telematica and Information Technology, University of Tweente

Carroll, J. J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., and Wilkinson, K., 2004, Jena: Implementing the semantic web recommendations, in *Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers &Amp; Posters*, WWW Alt. '04, New York, NY, USA, ACM, 74–83 pp.

Dabrowski, M., Griffin, K., and Passant, A., 2011, Approaches for real-time integration of semantic web data in distributed enterprise systems, in *Proceedings of the 5th IEEE International Conference on Semantic Computing (ICSC 2011), Palo Alto, CA, USA, September 18-21, 2011*, 47–50 pp.

Demazeau, Y., 1999, Multi-agent systems methodology, in *CEEMAS*

Dikenelli, O., 2014, Where are all the semantic web agents: Establishing links between agent and linked data web through environment abstraction, in *E4MAS 2014*

Dolia, P. M., 2010, Integrating ontologies into multi-agent systems engineering (mase) for university teaching environment, *Journal of Emerging Technologies in Web Intelligence* **2(1)**, 42 pp.

Erdur, R. C., Alatli, O., Halaç, T. G., and Dikenelli, O., 2013, Monitoring

KAYNAKLAR DİZİNİ (Devam)

the dynamism of the linked data space through environment abstraction, in *Proceedings of the 9th International Conference on Semantic Systems*, ACM, 81–88 pp.

Erdur, R. C., Dikenelli, O., Alatlı, O., Ekinci, E. E., and Akar, Z., 2012, Integrating linked data space with agents using the environment abstraction, in *Computer Software and Applications Conference Workshops (COMPSACW), 2012 IEEE 36th Annual*, IEEE, 625–630 pp.

Fitzpatrick, B., Slatkin, B., and Atkins, M., 2010, *Pubsubhubbub core 0.3–working draft*, https://pubsubhubbub.googlecode.com/svn/trunk/pubsubhubbub-core-0.3.html, (Erişim Tarihi: 23 Haziran 2015)

Flouris, G., Manakanatas, D., Kondylakis, H., Plexousakis, D., and Antoniou, G., 2008, Ontology change: Classification and survey, *The Knowledge Engineering Review* **23(02)**, 117 pp.

Flouris, G. and Plexousakis, D., 2005, *Handling Ontology Change: Survey and Proposal for a Future Research Direction*, Technical report, Institute of Computer Science, FORTH, Greece

Foundation, A. S., 2015, *ARQ - A SPARQL Processor for Jena*, https://jena.apache.org/documentation/query/, (Erişim Tarihi: 24 Haziran 2015)

Garshol, L. M. and Borge, A., 2013, Hafslund sesam–an archive on semantics, in *The Semantic Web: Semantics and Big Data*, Springer, 578–592 pp.

Gearon, P., Passant, A., and Polleres, A., 2013, Sparql 1.1 update. w3c recommendation, 21 march 2013, *World Wide Web Consortium. Retrieved from http://www. w3. org/TR/sparql11-update*

Genesereth, M. R. and Ketchpel, S. P., 1994, Software agents, *Commun. ACM* **37(7)**, 48 pp.

Gießmann, L., Küster, M. W., and Ludwig, C., 2009, Isidor-ui: Generating a user interface with topic maps constraint language and javascript object notation, *Linked Topic Maps* 207 pp. pp.

Gilbert, L. S., 1999, Where is my brain? distributed cognition, activity theory, and cognitive tools., in *Proceedings of National Convention of the AECT*, Houston, Texas, USA., ERIC

Görlitz, O. and Staab, S., 2011, SPLENDID: SPARQL endpoint federation exploiting VOID descriptions, in *Proceedings of the Second International Workshop on Consuming Linked Data (COLD2011), Bonn, Germany, October*

KAYNAKLAR DİZİNİ (Devam)

23, 2011

Group, W. O. W., 2012, Owl 2 web ontology language document overview

Gruber, T. R., 1993, A translation approach to portable ontology specifications, *Knowledge acquisition* **5(2)**, 199 pp.

Haase, P. ve Sure, Y., 2004, *D3.1.1.b state of the art on ontology evolution*, Technical report, Karlscruher Institute für Technologie

Haslhofer, B. and Schandl, B., 2010, Interweaving oai-pmh data sources with the linked data cloud, *International Journal of Metadata, Semantics and Ontologies* **5(1)**, 17 pp.

Hausenblas, M., 2009, Exploiting linked data to build web applications, *IEEE Internet Computing* **13(4)**, 68 pp.

Hausenblas, M., 2010, *Dataset Dynamics Vocabulary*, http://vocab.deri.ie/dady, (Erişim Tarihi: 23 Şubat 2014)

Hayes-Roth, B., 1995, An architecture for adaptive intelligent systems, *Artificial Intelligence* **72(1–2)**, 329 pp.

Heath, T. and Bizer, C., 2011, Linked data: Evolving the web into a global data space, *Synthesis lectures on the semantic web: theory and technology* **1(1)**, 1 pp.

Heath, T., Hausenblas, M., Bizer, C., Cyganiak, R., and Hartig, O., 2008, How to publish linked data on the web, in *Tutorial in the 7th International Semantic Web Conference, Karlsruhe, Germany*

Heylighen, F., Heath, M., and Van, F., 2004, The emergence of distributed cognition: a conceptual framework, in *Proceedings of collective intentionality IV*

Huhns, M. N. and Singh, M. P., 1997, Agents on the web: Ontologies for agents, *IEEE Internet Computing* 1

J., O., V., P., and M. P., L., 2003, *FIPA modeling area: Environment*, Technical report, Foundation for Intelligent Physical Agents

Käfer, T., Abdelrahman, A., Umbrich, J., O'Byrne, P., and Hogan, A., 2013, Observing linked data dynamics, in *The Semantic Web: Semantics and Big Data*, Springer, 213–227 pp.

Kaptelinin, V. and Nardi, B., 2006, Activity theory in a nutshell, *Acting with technology: Activity Theory and interaction design* 29–72 pp. pp.

KAYNAKLAR DİZİNİ (Devam)

Kaptelinin, V. and Nardi, B., 2012, Activity theory in hci: Fundamentals and reflections, *Synthesis Lectures Human-Centered Informatics* **5(1)**, 1 pp.

Klein, M., 2004, *Change management for distributed ontologies*, *Ph.D. thesis*, SIKS, the Dutch Graduate School for Information and Knowledge Systems

Klein, M. and Fensel, D., 2001, Ontology versioning on the semantic web, in *Stanford University*, 75–91 pp.

Klein, M., Fensel, D., Kiryakov, A., and Ognyanov, D., 2002, Ontology versioning and change detection on the web, in *In 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02*, 197–212 pp.

Klein, M., Kiryakov, A., Ognyanov, D., and Fensel, D., 2003, Finding and characterizing changes in ontologies, in *Conceptual Modeling—ER 2002*, Springer, 79–89 pp.

Kolomvatsos, K. and Hadjiefthymiades, S., 2008, Ontologies and intelligent agents: A powerful bond, *Chapter in 'The Semantic Web for Knowledge and Data Management: Technologies and Practices'(eds Z. Ma, H. Wang), IDEA Group Inc*

Konev, B., Lutz, C., Walther, D., and Wolter, F., 2008a, Cex and mex: Logical diff and semantic module extraction in a fragment of owl, in *4th OWL Experiences and Directions Workshop (OWLED-2008DC)*

Konev, B., Walther, D., and Wolter, F., 2008b, The logical difference problem for description logic terminologies, in *Automated Reasoning, 4th International Joint Conference, IJCAR 2008, Sydney, Australia, August 12-15, 2008, Proceedings*, 259–274 pp.

Kremen, P., Smid, M., and Kouba, Z., 2011, Owldiff: A practical tool for comparison and merge of OWL ontologies, in *2011 Database and Expert Systems Applications, DEXA, International Workshops, Toulouse, France, August 29 - Sept. 2, 2011*, 229–233 pp.

KWTR, 2011, *KWTR: ontology versioning*, http://semanticweb.org/wiki/KWTR:_ontology_versioning, (Erişim tarihi: 11 Kasım 2011)

Langridge, Stuart ve Hickson, I., 2002, *Pingback 1.0 Specification*, Technical report

Lee, T. B., 2006, *Linked Data*, http://www.w3.org/DesignIssues/LinkedData.html, (Erişim Tarihi: 23 Haziran 2015), Erişim: 4 Şubat 2015

KAYNAKLAR DİZİNİ (Devam)

Liang, Y., Alani, H., Dupplaw, D., and Shadbolt, N., 2006a, An approach to cope with ontology changes for ontology-based applications, in *Second Advanced Knowledge Technologies DTA Symposium*, Event Dates: January 2006

Liang, Y., Alani, H., and Shadbolt, N., 2006b, Ontologies change and queries break: Towards a solution, in *International Conference on Knowledge Engineering and Knowledge Management-Managing Knowledge in a World of Networks*

Liang, Y. D., 2006, *Enabling active ontology change management within semantic web-based applications*, Doktora tezi ilerleme raporu

Ma, Z. and Wang, H., 2009, *The Semantic Web for Knowledge and Data Management*, Chapt. Ontologies and Intelligent Agents: A Powerful Bond, 49-73 pp., IGI Global

McGuinness, D. L., Van Harmelen, F., et al., 2004, Owl web ontology language overview, *W3C recommendation*

Moore, G. and Marius, L., 2015, *SDShare - A Protocol for the Syndication of Resource Descriptions*, http://www.sdshare.org/spec/sdshare-20120710.html, (Erişim Tarihi: 23 Haziran 2015)

Neches, R., Fikes, R., Finin, T. W., Gruber, T. R., Patil, R. S., Senator, T. E., and Swartout, W. R., 1991, Enabling technology for knowledge sharing, *AI Magazine* **12(3)**, 36 pp.

Noy, N. F., Chugh, A., Liu, W., and Musen, M. A., 2006, A framework for ontology evolution in collaborative environments, in *The Semantic Web - ISWC 2006, 5th International Semantic Web Conference, ISWC 2006, Athens, GA, USA, November 5-9, 2006, Proceedings*, 544–558 pp.

Noy, N. F. and Klein, M. C. A., 2004, Ontology evolution: Not the same as schema evolution, *Knowl. Inf. Syst.* **6(4)**, 428 pp.

Noy, N. F. and Musen, M. A., 2002, PROMPTDIFF: A fixed-point algorithm for comparing ontology versions, in *Proceedings of the Eighteenth National Conference on Artificial Intelligence and Fourteenth Conference on Innovative Applications of Artificial Intelligence, July 28 - August 1, 2002, Edmonton, Alberta, Canada.*, 744–750 pp.

Noy, N. F. and Musen, M. A., 2004, Ontology versioning in an ontology management framework, *IEEE Intelligent Systems* **19(4)**, 6 pp.

Odell, J., Parunak, H. V. D., Fleischer, M., and Brueckner, S., 2003, Modeling agents and their environment: The physical environment, *Journal of Object*

KAYNAKLAR DİZİNİ (Devam)

Technology **2(2)**, 43 pp.

Omicini, A., Ricci, A., and Viroli, M., 2006, Agens faber: Toward a theory of artefacts for mas, *Electronic Notes in Theoretical Computer Science* **150(3)**, 21 pp.

OWLdiff, 2015, http://krizik.felk.cvut.cz/km/owldiff, (Erişim: 4 Şubat 2015)

Palma, R., Haase, P., Corcho, Ó., and Gómez-Pérez, A., 2009, Change representation for OWL 2 ontologies, in *Proceedings of the 5th International Workshop on OWL: Experiences and Directions (OWLED 2009), Chantilly, VA, United States, October 23-24, 2009*

Palma, R., Haase, P., Wang, Y., and d'Aquin, M., 2008, *D1.3.1 Propagation Models and Strategies*, Technical report, UPM

Parunak, H. V. D., 1997, " go to the ant": Engineering principles from natural multi-agent systems, *Annals of Operations Research* **75**, 69 pp.

Passant, A. and Mendes, P. N., 2010, sparqlpush: Proactive notification of data updates in RDF stores using pubsubhubbub, in *Proceedings of the Sixth Workshop on Scripting and Development for the Semantic Web, Crete, Greece, May 31, 2010*

Peter F. Patel-Schneider, I. H. and Grau, B. C., 2006, Owl 1.1 web ontology language overview, *W3C Recommendation*

Peter Plessers, O. D. T. and Casteleyn, S., 2007, Understanding ontology evolution: A change detection approach, *J. Web Sem.* **5(1)**, 39 pp.

Platon, E., Mamei, M., Sabouret, N., Honiden, S., and Parunak, H. V. D., 2007, Mechanisms for environments in multi-agent systems: Survey and opportunities, *Autonomous Agents and Multi-Agent Systems* **14(1)**, 31 pp.

Plessers, P. and Troyer, O. D., 2005, Ontology change detection using a version log, in *The Semantic Web - ISWC 2005, 4th International Semantic Web Conference, ISWC 2005, Galway, Ireland, November 6-10, 2005, Proceedings*, 578–592 pp.

Popitsch, N. and Haslhofer, B., 2011, Dsnotify–a solution for event detection and link maintenance in dynamic datasets, *Web Semantics: Science, Services and Agents on the World Wide Web* **9(3)**, 266 pp.

Quilitz, B. and Leser, U., 2008, Querying distributed RDF data sources with SPARQL, in *The Semantic Web: Research and Applications, 5th European Semantic Web Conference, ESWC 2008, Tenerife, Canary Islands, Spain, June*

KAYNAKLAR DİZİNİ (Devam)

1-5, 2008, Proceedings, 524–538 pp.

Rao, A. and Georgeff, M., 1992, Social plans: Preliminary report, in E. Werner and Y. Demazeau (eds.), *Proceedings of the Third European Workshop on Modelling Autonomous Agents and Multi-Agent Worlds (MAAMAW-91)*, pp. 57-76 pp.

Ricci, A., Piunti, M., Acay, L. D., Bordini, R. H., Hübner, J. F., and Dastani, M., 2008a, Integrating heterogeneous agent programming platforms within artifact-based environments, in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, International Foundation for Autonomous Agents and Multiagent Systems, 225–232 pp.

Ricci, A., Piunti, M., and Viroli, M., 2011, Environment programming in multi-agent systems: an artifact-based perspective, *Autonomous Agents and Multi-Agent Systems* **23(2)**, 158 pp.

Ricci, A., Piunti, M., Viroli, M., and Omicini, A., 2009, Environment programming in cartago, in *Multi-Agent Programming:*, Springer, 259–288 pp.

Ricci, A., Viroli, M., and Omicini, A., 2006, Programming mas with artifacts, in *Programming multi-agent systems*, Springer, 206–221 pp.

Ricci, A., Viroli, M., and Omicini, A., 2007, Cartago: A framework for prototyping artifact-based environments in mas, in *Proceedings of the 3rd International Conference on Environments for Multi-agent Systems III*, E4MAS'06, Berlin, Heidelberg, Springer-Verlag, 67–86 pp.

Ricci, A., Viroli, M., and Omicini, A., 2008b, The a&a programming model and technology for developing agent environments in mas, in *Programming multi-agent systems*, Springer, 89–106 pp.

Russell, S. J. and Norvig, P., 2003, *Artificial Intelligence: A Modern Approach*, Pearson Education, 2 edition

Sanz, R. and López, I., 2006, *A Survey on Ontologies for Agents, From Theory to*, Technical report, Universidad Politecnica De Madrid

Schmachtenberg, M., Bizer, C., and Paulheim, H., 2014, *State of the LOD Cloud 2014 Version 0.4*, http://linkeddatacatalog.dws.informatik.uni-mannheim.de/state/, (Erişim Tarihi: 23 Haziran 2015), Mannheim, Germany

Schwarte, A., Haase, P., Hose, K., Schenkel, R., and Schmidt, M., 2011a, Fedx: A federation layer for distributed query processing on linked open data, in

KAYNAKLAR DİZİNİ (Devam)

The Semanic Web: Research and Applications - 8th Extended Semantic Web Conference, ESWC 2011, Heraklion, Crete, Greece, May 29 - June 2, 2011, Proceedings, Part II, 481–486 pp.

Schwarte, A., Haase, P., Hose, K., Schenkel, R., and Schmidt, M., 2011b, Fedx: Optimization techniques for federated query processing on linked data, in *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, 601–616 pp.

Sebastian Tramp, P. F., Ermilov, T., and Auer, S., 2010, Weaving a social data web with semantic pingback, in *Knowledge Engineering and Management by the Masses - 17th International Conference, EKAW 2010, Lisbon, Portugal, October 11-15, 2010. Proceedings*, 135–149 pp.

Shadbolt, N., Hall, W., and Berners-Lee, T., 2006, The semantic web revisited, *Intelligent Systems, IEEE* **21(3)**, 96 pp.

Shinavier, J., 2010, Optimizing real-time RDF data streams, *CoRR* abs/1011.3595

Stojanovic, L., 2004, *Methods and tools for ontology evolution*, *Ph.D. thesis*, Karlsruhe Institute of Technology

Stojanovic, L., Maedche, A., Motik, B., and Stojanovic, N., 2002, User-driven ontology evolution management, in *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings*, 285–300 pp.

Trifa, V., Guinard, D., Davidovski, V., Kamilaris, A., and Delchev, I., 2010, Web messaging for open and scalable distributed sensing applications, in *Web Engineering, 10th International Conference, ICWE 2010, Vienna, Austria, July 5-9, 2010. Proceedings*, 129–143 pp.

Tudorache, T., 2015, *Collaborative Protege*, http://protegewiki.stanford.edu/wiki/Collaborative_Protege, (Erişim tarihi: 23 Haziran 2015)

Tudorache, T., Noy, N. F., and Musen, M. A., 2008, Collaborative protege: Enabling community-based authoring of ontologies., Vol. 401 of *CEUR Workshop Proceedings*, CEUR-WS.org

Tummarello, G., Morbidoni, C., Bachmann-Gmür, R., and Erling, O., 2007, Rdfsync: Efficient remote synchronization of RDF models, in *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*, 537–551 pp.

KAYNAKLAR DİZİNİ (Devam)

Tunnicliffe, S. and Davis, I., 2015, *Changeset*, http://vocab.org/changeset/schema.html, (Erişim Tarihi: 23 Haziran 2015)

Tury, M. and Bielikova, M., 2006, An approach to detection ontology changes, in *Workshop Proceedings of the 6th International Conference on Web Engineering, ICWE 2006, Palo Alto, California, USA, July 11-14, 2006*, 14 pp.

Umbrich, J., Villazón-Terrazas, B., and Hausenblas, M., 2010, Dataset dynamics compendium: A comparative study, in *Proceedings of the First International Workshop on Consuming Linked Data, Shanghai, China, November 8, 2010*

Viroli, M., Omicini, A., and Ricci, A., 2005, Engineering mas environment with artifacts, in *2nd International Workshop "Environments for Multi-Agent Systems"(E4MAS 2005)*, AAMAS, 62–77 pp.

Volkel, M., 2006, *D2. 3.3. v2 SemVersion Versioning RDF and Ontologies*, Technical report, University of Karlsruhe

Volkel, M. and Groza, T., 2006, Semversion: An rdf-based ontology versioning system, in *Proceedings of the IADIS international conference WWW/Internet*, Vol. 2006, 44 pp.

Weyns, D. and Holvoet, T., 2005, On the role of environments in multiagent systems, *Informatica (Slovenia)* **29(4)**, 409 pp.

Weyns, D., Omicini, A., and Odell, J., 2007, Environment as a first class abstraction in multiagent systems, *Autonomous agents and multi-agent systems* **14(1)**, 5 pp.

Weyns, D., Vizzari, G., and Holvoet, T., 2005, Environments for situated multi-agent systems: Beyond infrastructure, in *Environments for Multi-Agent Systems II, Second International Workshop, E4MAS 2005, Utrecht, The Netherlands, July 25, 2005, Selected Revised and Invited Papers*, 1–17 pp.

Wikipedia, Activity theory, http://en.wikipedia.org/wiki/Activity_theory, (Erişim Tarihi: 23 Haziran 2015)

Wikipedia, *Socially Distributed Cognition*, http://en.wikipedia.org/wiki/Socially_distributed_cognition, (Erişim Tarihi: 23 Haziran 2015)

Wooldridge, M. and Jennings, N., 1995, Intelligent agents: Theory and practice, *The knowledge engineering review* **10(02)**, 115 pp.

Y., H. J. S., P., H., R., P., and M., S., 2005, Omv–ontology metadata vocabulary,

KAYNAKLAR DİZİNİ (Devam)

in *ISWC*, Jens Hartmann, York Sure, Peter Haase, Raul Palma and Mari del Carmen Su

Yu, L., 2011, *A developer's guide to the semantic Web*, Springer Science & Business Media

Zablith, F., Antoniou, G., d'Aquin, M., Flouris, G., Kondylakis, H., Motta, E., Plexousakis, D., and Sabou, M., 2015, Ontology evolution: a process-centric survey, *Knowledge Eng. Review* **30(1)**, 45 pp.

ÖZGEÇMİŞ

Adı Soyadı: Oylum ALATLI
Doğum Tarihi: 1979
Doğum Yeri: İzmir
Uyruğu: T.C.

EĞİTİM

Yüksek Lisans: Ege Üniversitesi
Bilgisayar Mühendisliği Bölümü, 2003

Lisans: Ege Üniversitesi
Bilgisayar Mühendisliği Bölümü, 2000

Lise: Karşıyaka Anadolu Lisesi, İzmir, 1996

İLGİ ALANLARI

Çoklu Etmen Sistemleri, Anlamsal Veb, Bağlı Veri

EKLER

EK1: Türkçe-İngilizce Terimler Sözlüğü

açıklama	annotate
altsınıf	subclass
AltSorguArtifaktı	SubQueryArtifact
anlamsal ağ	semantic network
atomik operasyon	atomic operation
AtomikDeğişim	AtomicChange
betik	script
bilgi tabanı	knowledgebase
bilişimsel	computational
birleştirme	merge
birlikte işleyebilirlik	interoperability
çıkarma	removal
çıkarsama	reasoning
çizge	graph
değer kümesi	range
değişiklik yayılımı	change propagation
değişim	change
değişim operasyonu	change operation
değişim tanımlaması	change specification
dinamikler	dynamics
dipnot	annotation
ekleme	addition
eşleme	mapping
eşşekilli	isomorphic
Etmen Yönetim Birimi	Agent Management System
EvrimKavramı	EvolutionConcept
gerçek	fact
hariç tutma	exclusion
içerir	consists of
içerme	subsumption
ifade	statement
ifade gücü	expressiveness
ilişkiliOlanVarlık	hasRelatedEntity
işlem	transaction
kaynak	resource
kompozit operasyon	composite operation

KompozitDeğişim	CompositeChange
kümelenme	aggregation
kütük	log
mesaj içeriği ontolojisi	message content ontology
nesne	object
olumsuzlama	negation
ontoloji	ontology
OWL için Değişim Ontolojisi	Change Ontology for OWL
öncekiDeğişimiVardır	hasPreviousChange
önek	prefix
örnek	instance
ÖrnekVersiyonu	InstanceVersion
ÖzellikVersiyonu	PropertyVersion
özne	subject
sabit nokta	fixed point
sade	literal
sıradüzensel	hierarchical
sonuç kümesi	resultset
söz eylem	speech act
söz varlığı	vocabulary
Talep Etkileşim Protokolü	Request Interaction Protocol
tanım kümesi, alan	domain
tanımlama	definition, identification
Tanımlama Mantığı	Description Logic
ters	inverse
uygulananAksiyom	AppliedAxiom
üçlü	triple
üyelik	membership
VarlıkDeğişimi	EntityChange
Veri Ağı	Web of Data
versiyon	version
yansıma	image
yansımalı	reflexive
yapı	frame
yerleşik	built-in
yüklem	predicate