YAZILIM MÜHENDİSLİĞİ YÖNETİM SÜRECİ ONTOLOJİSİ

Barış ULU* ve Banu DİRİ**

(*) Yildiz Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü, İSTANBUL
(**) Yildiz Teknik Üniversitesi, Bilgisayar Mühendisliği Bölümü, İSTANBUL
baris47@gmail.com, banu@ce.yildiz.edu.tr

ÖZET

Günümüz yazılım mühendisliği bilgi tedariğinin en önemli sorunu verilerin çok farklı veri depolarında söz-dizimsel olarak tutulmasıdır. Artık haline gelmiş olan bu veriler, yazılım mühendisliği uygulamalarının tekrarlı verileri kullanmasına ve her yazılım geliştirme etkinliğinin sil baştan yapılmasına neden olmaktadır. Kazanılan tecrübelerde yeniden kullanılabilirliğin ya da geri dönüşlerin olmaması yazılım mühendisliği uygulamalarının, yazılım süreç modellerinde olduğu gibi, yazılım geliştirme sürecinde de verimsiz zaman ve kaynak kullanımını beraberinde getirmektedir. Yazılım geliştirme süreçlerinde tecrübelerin ve aynı zamanda bilginin yeniden kullanımın sağlanması için yazılım mühendisliği süreç modellerinde ontoloji yaklaşımının kullanılması yazılım mühendisliği yönetim sürecine fazlaca bağlı olan yazılım mühendisliği projelerinde bilgi ve tecrübeye dayalı başarıyı getirecektir. Bu çalışmanın amacı, yazılım mühendisliği standartlarının söz varlıklarının incelenmesi ve yazılım mühendisliği taksonomisinin geliştirilmesi, bu taksonomi ile birlikte elde edilen ilişkiler ve üst-veriye dayanarak kavramsal ontoloji modelinin çıkarılmasıdır.

Anahtar Kelimeler: Anlamsal Web, Ontoloji Modelleme, Yazılım Mühendisliği Süreç Ontolojisi, Yazılım Mühendisliği Yönetim Süreci Ontolojisi

SOFTWARE ENGINEERING MANAGEMENT PROCESS ONTOLOGY

ABSTRACT

The main problem in today's information provision approach in software engineering is the usage of data that has been defined syntactically on several corporate databases. The garbage data everywhere causes software engineering applications to have redundant data and initiative from scratch for each development activity. The lack of flashback or reuse of previous experiences among software engineering applications, as well as software process models, causes inefficient time and resource consuming during software development. In order to overcome this gap, ontological approach is used for the success of the software projects that highly depend on the success of the software management process. Defining the vocabulary of software engineering standards as constraints through software process taxonomies is important for this achievement. In this paper, a conceptual ontological model is introduced with software process taxonomy.

Keywords: Semantic Web, Ontology Modeling, Software Engineering Management Process, Software Engineering Management Process Ontology

1. GİRİŞ

Günümüzde Web, kullanıcılarına yapısal olmayan, dinamik, dağıtık ve hızla büyüyen veri yığını sunmaktadır. Bu yığın, verinin ne anlama geldiğini ifade etmeye çalışan mantıksal işlenebilirlikten uzak, standartlaştırılmış bir artık (garbage) halini almıştır. Buradaki temel sorun, farklı merkezlerde bulunan verilerin birbirleri arasında bilmediğimiz mantıksal ilişkilerinin kullanıcılarına sunulmuyor olmasıdır. Verinin sunumu önemlidir ve bu arzu edilen sunum, veriler arası anlamsal ilişkilerin yaratılması ve yapılandırılması ile mümkün olmaktadır.

Anlamsal Web, ilk olarak 1994 yılındaki Uluslararası Web Konferansı[1] 'nda

[1] http://www.iw3c2.org/

Tim Berners-Lee[2] tarafından ortaya atılmıştır. Temel amaç, Web'deki boş duran bağlantıların sadece istekleri bir noktan diğer bir noktaya yönlendirmesinin yanı sıra bilgi erişimi için de kullanılabileceğinin düşünülmesidir. Sözü geçen veri yığını aslında Web'in dışındaki verilerdir; uygulamalar tarafından yönetilirler, ancak bir veri ağının parçası değildirler. Bu veri boşluğu uygulamaların sadece, Web'de ya da değil, bağlanılabilir veri depolarında daha önceden tanımlanmış söz-dizimsel veriler ile beslenmelerine neden olmaktadır.

Yazılım mühendisliği süreçleri, tecrübe yani bilginin yeniden kullanımı mümkün olmadığından dolayı her ürün/proje başlangıç safhasında sil baştan kullanılan veriler ile modellenmektedir. Bu geri dönüşümün ya da yeniden kullanımın sağlanabilmesi için yazılım mühendisliği süreçleri modellenirken her sürecin diğer süreç ile arasındaki, her süreç içindeki faaliyetlerin de diğer faaliyetler ile anlamsal ilişkilerinin belirlenmesi gerekmektedir.

Bu bildiride, giriş, sonuç ve kaynaklar dışında iki ana bölüm ile birlikte çalışmanın yol haritasını belirleyen toplam üç ana başlık bulunmaktadır. İkinci bölümde, Anlamsal Web hakkında kısa bir bilgi verilmiştir. Üçüncü bölümde ise varolan süreçlerin zayıf yönleri ile birlikte varolan yazılım mühendisliği süreç modellerine alternatif olabilecek bir model öne sürülmüştür.

2. ANLAMSAL WEB

Anlamsal Web [1-2-3-4-5-9-11-13-21-22-24-29-30-31-32-33-53], bir kısım kullanıcıların varolan iletişim ağı içerisinde bildiklerini eklemelerine izin veren, bir kısım kullanıcıların da sorularına cevap vererek gelişen bir bilgi kümesidir. Anlamsal Web'de bilgi, doğal dil metininden farklı olarak, yapısal devam ettirilen ve bu sayede insan ve makineler tarafından kolaylıkla kullanılabilmektedir [16]. Bu bilgi kolleksiyonu, sadece Web'den tanıdığımız ortam nesneleri (Web sayfaları, görüntü, klipler, vs.) için değil bunun yanında insan, yer, organizasyon ve olaylar için de kaynak teşkil etmektedir [15].

Anlamsal Web, RDF [6-19-22-23-24-25-34] üzerine inşaa edilmiştir. RDF (Resource Description Framework), verilerin sunumunu sağlamak için üst-verileri [7-17-26-28-35-36-37-38] ifade etmeye yarayan URI (Uniform Resource Identifier) [2-23-24-34]' ları kullanmaktadır. RDF tarafından biçimlendirilen ve ilişkilendirilen bu veri kümeleri ontolojiler [20-39-40-41-42-43-44-45-46] yolu ile saklanmaktadır. Ontolojiler, dağıtık verilerin üst-verilerine dayalı olarak modellenmiş ortak iletişim araçlarıdır. Terminolojideki anlamının yanı sıra ontolojiler, bir takım aracı ya da ajanlar içindeki veri grupları olarak da ele alınmaktadır [5-14-24]. Bilgi, ontolojilerde çıkarsama [1-22-24-42-43-47-48-49-50-51-52-53-54] amacı ile tutulmaktadır. Çıkarsama, farklı veri kümelerinden elde edilen veriler ile bilginin sağlanması yöntemidir. Çıkarsama sayesinde pek çok artık verinin saklanması ihtiyacı ortadan kalkmaktadır.

3. ANLAMSAL YAZILIM SÜREÇLERİ

Yazılım mühendisliği süreçleri [18-27-55-56-57-58-63-64-67] farklı gruplarda ifade edilmektedir; bu süreçler girdiler ve çıktılar bazında birbirine sıkı sıkıya bağlıdır. Yazılım mühendisliği süreçlerinin ifade edilmesinde kullanılan pek çok model, süreç bilgi mal varlıkları [59] ile süreçlerin otomatikleştirilmesi [8] yerine iyileştirilmesine çaba harcamaktadır. Sözü geçen süreç bilgi mal varlıkları basitçe Şekil 1' de gösterilmiştir.

Şekil 1. Yazılım mühendisliği bilgi mal varlıkları

Sözü geçen süreç modelleri üst-seviye faaliyetlere odaklanmış modellerdir. Şekil 1' deki mal varlıkları arasındaki ilişkilerin tanımlanmamış olmasından dolayı pratik olamayacak kadar fazla soyut yapılardır [10]. Bu süreç modelleri, durağan tarzda bir sürecin hangi

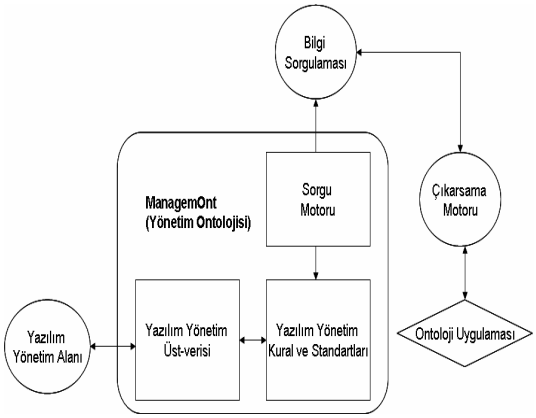
[2] http://www.w3.org/People/Berners-Lee/

adımlarının ya da safhalarının olması gerektiğini ifade etmeye çalışmaktadır. Bir sürecin modellenmesi ya da yeniden tasarlanması [60] için *nasıl?* sorusuna cevap aranmalıdır [12]. *Nasıl?* sorusuna verilecek cevaplar, süreçler arası anlamsal ilişkileri tanımlayarak süreçler arası dinamik uyarlamaları [61-62] sağlayacaktır. Bu cevaplar süreçler arası iletişimi kuvvetlendirecek, aynı zamanda da yazılım mühendisliği tecrübe [59] yani bilgilerinin yeniden kullanımını, süreçlerin olgunlaşabilmesini ve süreç modelleme [10] için standartların oluşmasını sağlayacaktır.

Geleneksel yazılım mühendisliği süreçleri, yazılım mühendisliği yönetim sürecine, risk, gereksinimler, kalite, maliyet, belgeleme, sınama ve doğrulama, ve zaman yönetimi faaliyetlerine sıkıca bağlıdır. Bu yönetim faaliyetlerinin sonuçları büyük kişisel belgeler halinde saklanmakta ve her yeni ürün ya da proje başlangıç safhasında bu belgeler, başarıların yeniden kullanılabilirliğini sağlamak amacı ile tekrar tekrar ayrıştırılmaya çalışılmaktadır. Bu ayrıştırma son derece zahmetli ve zor olduğundan dolayı her yeni ürün ya da proje sil baştan süreç planlamasına girmektedir. Bu da verimsiz zaman ve kaynak kullanımına neden olmaktadır.

Yazılım mühendisliği yönetim sürecinin anlamsal olarak modellenmesi risk tecrübelerinin, maliyet ve gereksinim analizlerinin başarılı olarak sonlandırılmış projelerden yeniden kullanımına olanak sağlamaktadır. Başarılı olmuş projelerde kullanılan proje planları, istatistiksel veriler her yeni projede zaman kaybı olmaksızın proje planlarının üretilmesine, buna bağlı olarak da proje süresince kaynak ve zamanın verimli kullanımına yardımcı olmaktadır. Anlamsal ilişkilerin belirlenmesi yolu ile doğru ya da doğruya yakın zaman çizelgeleri, görev atama ve zamanlamaları ile kalite güvence yöntemlerinin projelere kolay adaptasyonu sağlanabilmektedir. Bu adaptasyon, belgeleme, sınama ve doğrulama süreçlerinin daha önceki başarılı projelerde varolan çıktılarının yeniden kullanılabilmesi ile mümkün olmaktadır.

Şekil 2' de belirtilen yazılım mühendisliği alanı yukarıda bahsedilen diğer yazılım mühendisliği süreçlerini belirtmektedir. Bu süreçler, üst-verisi yolu ile yazılım mühendisliği yönetim süreci [18-27-55-56-57-58] ile ilişkilidirler. Bu üst-veri, IEEE [63-64] tarafından belirlenmiş kural ve standartlar ile biçimlendirilmiştir. Sorgu motoru ile kural ve standart tabanlı üst-veri üzerinde gerekli sorgulama yapılabilmekte ve sonuçlar çıkarsama motoruna iletilmektedir. Çıkarsama motoru sayesinde sorgulama sonucu elde edilmiş olan verilerden akıl yürütme ile ontoloji uygulamasının ihtiyacı olan yazılım mühendisliği yönetim bilgilerine ulaşılmaktadır. Yazılım mühendisliği kural ve standartları, sorgu motoru ve yazılım mühendisliği yönetim süreci üst-verisi, yazılım mühendisliği yönetim süreci ontolojisini oluşturmaktadır. Yazılım mühendisliği süreç ontolojisi kısaca ManagemOnt [71-72] olarak isimlendirilmiştir.

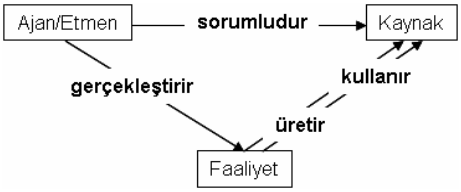


Şekil 2. Yazılım mühendisliği yönetim süreci anlamsal modeli

3.1. Yazılım Mühendisliği Süreci Üst-veri Modeli

Her yazılım mühendisliği süreci dinamik [61-62] ve otomatikleştirilmiş [65] bir bileşen olarak düşünülmelidir. Bu bileşenler, diğer bileşenlerden gelen çıktıları girdi olarak kabul ederler, içsel faaliyetlerini gerçekleştirip başka bir bileşene girdi olması için gerekli çıktıları üretirler. Bu bileşenler arasındaki girdi/çıktı alışverişleri diğer taraftan da ilişkileri belirler. Şekil 3' de belirtildiği gibi bileşen, süreç, faaliyet adı verilen içsel görevlerini gerçeklemektedir. Sürecin sahibi, ajan ya da

etmen, bu faaliyetleri gerçeklemekle yükümlüdür. Her etmen ya da ajana ait bir rol belirlenmiştir. Şekil 3' de sürecin rolü ilgili kaynaktan sorumlu olmaktır. Bu kaynaklar faaliyetler tarafından üretilir ya da kullanılırlar.



Şekil 3. Somut süreç modeli

SPEM (Software Process Engineering Metamodel) [66] yazılım mühendisliği faaliyet ve bağımlılıkları için temel bir üst-model sunmaktadır. Bu soysal (generic) model, proje planını kaynak olarak kullanır. Yazılım mühendisliği yöneticisini ajan ya da etmen olarak tanımlar. Faaliyet olarak ilgili planı üretir. Bu sayede yazılım mühendisliği süreçlerine kolaylıkla uygulanabilmektedir.

Yazılım mühendisliği organizasyonel süreçlerinden [18-27-63-64] yönetim süreci aşağıdaki şekilde listelenebilir:

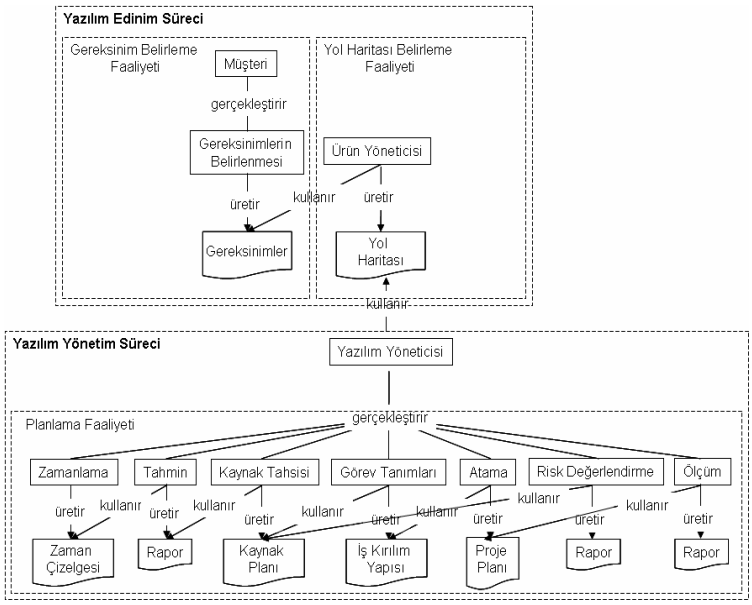
- Başlatma ve kapsam tanımı
- Planlama
- Yürütme ve kontrol
- Gözden geçirme ve değerlendirme
- Kapatma

Yukarıda listelenen tüm faaliyetlerin farklı kaynaklar ile birlikte bir ajan ya da etmeni mevcuttur. Bu ajan ya da etmen yazılım mühendisliği yöneticisi ya da ilgili faaliyeti yürüten nesnedir. Yazılım mühendisliği yönetim süreci taksonomisi, yazılım mühendisliği yönetim süreci üst-verisinin tanımlanmasına da yardımcı olacaktır. Şekil 4' de yazılım mühendisliği yönetim süreci taksonomisi verilmiştir.

Şekil 4. Yazılım mühendisliği yönetim süreci faaliyet taksonomisi

Şekil 4'te gösterilen taksonomi oldukça büyük olduğundan dolayı sözü geçen sürecin planlama faaliyeti ele alınmış ve Şekil 5' de yazılım mühendisliği yönetim sürecinin planlama faaliyetine ilişkin üst-veri *somut süreç modeli* baz alınarak belirtilmiştir.

Şekil 5, yazılım mühendisliği yönetim süreci planlama faaliyetine ilişkin üst-veriyi yazılım mühendisliği yönetim süreci taksonomine bağlı olarak sunmaktadır. Üst-veride belirtilen her faaliyet ortak bir ajan ya da etmene sahiptir. Her faaliyetin bir girdi ve bir çıktısı bulunmaktadır.



Şekil 5. Yazılım mühendisliği yönetim süreci planlama faaliyeti üst-veri modeli

3.2. Yazılım Mühendisliği Yönetim Süreci Ontolojisi: ManagemOnt

Yazılım mühendisliği yönetim süreci için tanımlanan üst-veri modeline ait her ajan/etmen, girdi, çıktı ve faaliyet yazılım mühendisliği yönetim ontolojisi için bir kavramı ifade etmektedir. Her kavramın kendine ait ozellikleri ve diğer kavramlar ile tanımlanmış ilişkileri mevcuttur.

ManagemOnt modelinin gerçekleştirim süreci, sürece ilişkin temeller, modelleme detayları, geliştirme yöntem ve araçları ile birlikte modelin çıktıları aşağıdaki başlıklarda özetlenebilir.

3.2.1. Temeller

ManagemOnt, modelde yer alan kavramların anlamları yerine aşağıdaki sorulara [70] cevap arayan bir model olarak tasarlanmıştır:

- ManagemOnt'ta kavramlar nelerdir?
- ManagemOnt'ta kavram mirası nedir?
- ManagemOnt'ta nesneler nelerdir?
- ManagemOnt'ta ajan/etmen nedir?
- ManagemOnt'ta aktiviteler nelerdir?
- ManagemOnt'ta çıktılar nelerdir?
- ManagemOnt'taki "is-a" ve "part-of" bağlantıları nelerdir?

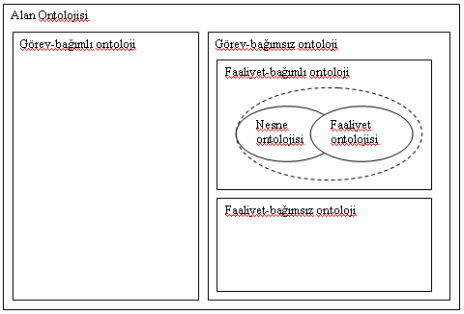
3.2.2. Modelleme Detayları

ManagemOnt modelinin ontoloji seviyeleri aşağıda listelenmiştir:

- Seviye-1: ManagemOnt ortak bir kelime hazinesi olarak modellenmiştir. Yazılım Mühendisliği Yönetim Süreci' nde yer alan kavramların bir kümesi olarak tanımlanmaktadır. Bu kavramlar bir hiyerarşi şeklinde modellenmiştir.
- Seviye-2: ManagemOnt, Yazılım Mühendisliği Yönetim Süreci' nde yer alan ilişkilerin tanımlanması ile ilişkisel bir veritabanının kavramsal şeması olarak modellenecektir. Bu ilişkiler kavramlar arası anlamsal ilişkileri ifade edecektir.
- Seviye-3: ManagemOnt içeriği sağlanacaktır. Bu içerik sayesinde ManagemOnt bir veri kümesi olarak modellenecektir.
- Seviye-4: ManagemOnt, terminoloji, kavramların anlamları, alan ontolojisi ve görev ontolojisi [68] ile bir standardizasyon sağlayacaktır. ManagemOnt, nihai olarak Seviye-4 ontolojisi olarak sonlandırılacaktır.

ManagemOnt modelinin ontoloji tipi Şekil 6' da belirtilmiştir. ManagemOnt Yazılım

Mühendisliği Yönetim Süreci alanında bir kavramsallaştırma sağlaması nedeni ile bir alan ontolojisi olarak modellenmiştir. Alan ontolojileri, Görev-bağımlı ve görev-bağımsız ontolojiler olarak sınıflandırılmaktadır [68]. Görev-bağımlı ontolojiler, bir uygulama alanında tam bir bilgiye ihtiyaç duymayan, sadece o alandaki görevlerin modellenmesi için gerçeklenmiş ontolojilerdir. ManagemOnt' ta ise alan bilgisi önemlidir. Bu alan bilgisi ile Yazılım Mühendisliği Yönetim Süreci alanındaki kavramlar detaylandırılmış, bu kavramlar temel birim olan somut süreç modeli ile modellenmiştir. Bu sebeple ManagemOnt bir görev-bağımsız ontoloji olarak modellenmiştir. Görev-bağımsız ontolojiler, içerdikleri faaliyet seviyelerine göre de sınıflandırılırlar [68]. Faaliyet-bağımlı ontolojiler, uygulama alanında varolan faaliyetlerin oluşturduğu ilişkiler yolu ile modellenirler. Bu faaliyetler, nesnelerin davranışları ve organizasyonel davranışlardır. Yazılım Mühendisliği Yönetim Süreci, alan karakteri düşünüldüğünde, nesnelerin ve organizasyonel faaliyetlerin sıkça yer aldığı bir uygulama alanıdır. Bu nedenle de ManagemOnt, durağan kavramlarının modellenmesi nedeniyle hem bir nesne ontolojisi, hem de nesneler arası ilişkilerin modellendiği bir ontoloji olması nedeniyle faaliyet ontolojisi olarak modellenmiştir.



Şekil 6. ManagemOnt ontoloji tipi

3.2.3. Araç ve Yöntemler

ManagemOnt modeli oluşturulurken aşağıdaki metodolojiler [69] incelenmiştir:

- Uschold & King metodolojisi
- TOVE metodolojisi
- Methontology
- On-to-Knowledge metodolojisi
- AFM (Activity-First Method in Huzo) metodolojisi

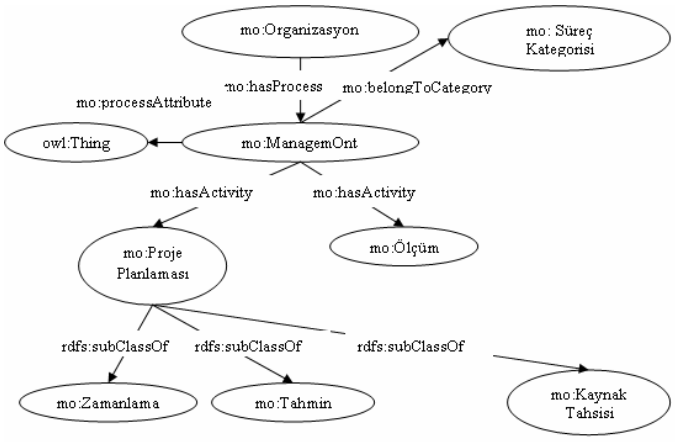
ManagemOnt'un bir alan ontolojisi olması nedeniyle de AFM (Activity-First Method in Hozo) metodolojisi ManagemOnt modellenirken kullanılmıştır.

AFM metodolojisi, görev ve alan ontolojilerinin modellenmesinde kullanılmaktadır. Bu modeller varolan teknik belgelerden yola çıkarak hazırlanmaktadır. AFM' nin en önemli prensibi, ilgili alandaki kavramların belirlenen görevlerdeki rollerinin organize edilmesidir. ManagemOnt modellenirken AFM metodolojisinde aşağıdaki adımlar takip edilmiştir:

1. Yazılım Mühendisliği Yönetim Süreci' ndeki görev birimlerinin tanımlanması
2. Yazılım Mühendisliği Yönetim Süreci' ndeki aktivitelerin organize edilmesi
3. Yazılım Mühendisliği Yönetim Süreci' ndeki görev yapısının analiz edilmesi
4. Yazılım Mühendisliği Yönetim Süreci' ndeki alan kavramlarının organize edilmesi

ManagemOnt modeli oluşturulurken aşağıdaki ontoloji geliştirme dilleri [69] incelenmiştir:

- Ontolingua
  - Herhangi bir çıkarsama yöntemi içermemektedir.
  - Kısıtlı işleve sahiptir.
  - ManagemOnt için uygun değildir.
- RDF(S)
  - Üst-veri gösterimi için kullanılır.
  - Üçleme modelini [6-9] uygular.
  - ManagemOnt için uygundur.
- OWL (DAML+OIL)
  - RDF(S)' den türetilmiştir.
  - ManagemOnt için uygundur.



Şekil 7. ManagemOnt RDF grafiği (kısmi)

ManagemOnt modellenirken aşağıdaki ontoloji geliştirme araçları [69] incelenmiştir:

- OntoEdit
- WebODE
- Protogé
- OE: Ontology Editor in Hozo

OE: Ontology Editor in Hozo, ManagemOnt modellenirken OE' nin kullandığı AFM'nin kullanılmış olması, rol kavramları, temel alan kavramları ve rol tutucu yani ajan/etmen tabanlı modelleme sağlamasından dolayı tercih edilmiştir.

3.2.4. Modelleme Çıktıları

ManagemOnt modelleme çıktıları RDF grafiği, RDF kaynak kodu ve RDF şeması kaynak kodu olarak elde edilmiştir.

ManagemOnt RDF grafiği, Şekil 4 ve Şekil 5' de belirtilmiş olan ManagemOnt taksonomisi temel alınarak Şekil 7' de kısmi olarak ifade edilmiştir.

Şekil 7' de belirtilmiş olan ManagemOnt RDF grafiğinin ontoloji geliştirme araçları ile modellenmesi ile birlikte ilişkili RDF belgesi, Şekil 7' deki kısmi grafiğin çıktısı olarak aşağıdaki şekilde elde edilmektedir.

```
<rdf_:Ajan
rdf:about="&rdf_;OSSD_Process_M
odel_00076"
        rdf_:name="Yazılım
Yöneticisi"
        rdfs:label="Yazılım
Yöneticisi">
        <rdf_:acts_on
rdf:kaynak="&rdf_;OSSD_Process_Mo
del_00096"/>
        <rdf_:acts_on
rdf:kaynak="&rdf_;OSSD_Process_Mo
del_00108"/>
</rdf_:Ajan>
<rdf_:Faaliyet
rdf:about="&rdf_;OSSD_Process_Mode
l_00096"
        rdf_:name="Proje Planlama"
        rdfs:label="Proje Planlama">
        <rdf_:condition></rdf_:conditio
n>
        <rdf_:provides
rdf:kaynak="&rdf_;OSSD_Process_Mo
del_00088"/>
        <rdf_:next_control_flow
rdf:kaynak="&rdf_;OSSD_Process_Mo
del_00108"/>
</rdf_:Faaliyet>
<rdf_:Kaynak
rdf:about="&rdf_;OSSD_Process_Mode
l_00088"
        rdf_:name="Project Plan"
        rdfs:label="Project Plan">
        <rdf_:required_by
rdf:faaliyet="&rdf_;OSSD_Process_Mo
del_00108"/>
</rdf_:Kaynak>
...
...
```

Elde edilen RDF belgesinin ajan/etmen, kaynak ve faaliyetler arasındaki ilişkilerinin tanımlanabilmesi için ilgili RDF şeması aşağıdaki şekilde gerçeklenmelidir.

```
<rdfs:Class
rdf:about="&rdf_;Faaliyet"
          rdfs:comment=""
          rdfs:label="Faaliyet">
         <rdfs:subClassOf
rdf:kaynak="&rdfs;Kaynak"/>
</rdfs:Class>
<rdfs:Class rdf:about="&rdf_;Ajan"
          rdfs:comment=""
          rdfs:label="Ajan">
         <rdfs:subClassOf
rdf:kaynak="&rdfs;Kaynak"/>
</rdfs:Class>
<rdfs:Class
rdf:about="&rdf_;Kaynak"
          rdfs:comment=""
          rdfs:label="Kaynak">
         <rdfs:subClassOf
rdf:kaynak="&rdfs;Kaynak"/>
</rdfs:Class>
<rdf:Property
rdf:about="&rdf_;acts_on"
          rdfs:comment=""
          rdfs:label="acts on">
         <rdfs:range
rdf:kaynak="&rdf_;Faaliyet"/>
          <rdfs:domain
rdf:kaynak="&rdf_;Ajan"/>
</rdf:Property>
...
...
```

4. GELECEK ÇALIŞMA

ManagemOnt isimli çalışma IEEE 12207.0' da detaylı olarak belirlenmiş olan yazılım mühendisliği standartlarının incelenmesi ile başlamıştır. Bu standartlar yazılım mühendisliği yönetim sürecinin üst-verisinin modellenmesinde kullanılmıştır. Elde edilen üst-veri yolu ile yazılım mühendisliği yönetim süreci ve diğer süreçler arası anlamsal ilişkiler belirlenmiştir. Bu anlamsal ilişkiler yolu ile elde edilen kavramlar ile birlikte sürece ait kuralların, IEEE 12207.0 standartlarına uygun olarak tanımlanması tamamlanmıştır. İlgili sürecin tanımlanan kuralları yolu ile sürecin elde edilmiş olan durağan kavramları Şekil 7' de belirtilen RDF grafiği temel alınarak yazılım mühendisliği yönetim sürecine ait girdi ve çıktıların modele dahil edilmesi yolu ile somut süreç modeli tabanlı yazılım mühendisliği yönetim sürecine ait ontoloji modeli elde edilmiştir.

Verilerin sorgulanabilmesi için sorgu motoru geliştirilmesi ve elde edilen verilerden bilgiye erişecek olan çıkarsama motorunun yazılım mühendisliği uygulama alanındaki kurallar çerçevesinde kullanılması hedeflenmektedir.

Son olarak, elde edilen çıkarsanmış bilgilerin kullanılacağı Anlamsal Yazılım Yönetim Birimi isimli ontoloji uygulaması geliştirilecektir. Uygulamanın amacı, günümüz yazılım mühendisliği süreç ve yönetim araçlarından farklı olarak doğru veri ile söz-dizimsel sonuçlar üreten değil, veri ile anlamsal sonuçlar üretebilen bir alternatif sunmaktır.

5. SONUÇLAR

Bilgi yönetiminin çok önemli olduğu günümüzde, bilgi dağıtık ve karmaşık yapıdadır. Varolan teknolojilerin bilgiler arasındaki ilişkileri ortaya koyması, doğru ve güncel verilere ulaşması güçtür. Ulaşılan bilgi ile ilişkisi bulunan diğer bilgilerin bilinmediği durumda eldeki bilgi de yararsız hale gelmektedir.

Yazılım kuruluşlarında tamamlanan projeler hakkındaki bilgiler, kuruluşların tanımladığı havuzlarda toplanmaktadır. Ancak gelecekte bu bilgilerin kullanılması güç olmakta ve kullanılmak istense de adaptasyon ve söz dizimsel zorluklardan dolayı vazgeçilmektedir. Bunun sonucunda yazılım mühendisliği yönetim süreci tarafından başarılı projelerde ya da ürün geliştirme süreçlerinde kullanılan plan, zamanlama ya da görevlendirme çizelgeleri ile yeni projelere uyarlanamamaktadır.

Bu bildiride, yazılım mühendisliği yönetim sürecinin sahip olduğu verilerin tanımlanması ve bu verilerin diğer yazılım mühendisliği süreçleri ile olan anlamsal ilişkilerinin belirlenmesinin önemi tartışılmıştır. Elde edilen üst-veriler, yazılım mühendisliği yönetim ontolojisinin verileri olarak kullanılmıştır. Varolan iş akış yönetimi ve

belgeleme yönetimi modellerine benzer olarak yazılım mühendisliği yönetim süreci üst-verisi elde edilirken somut süreç modeli kullanılmıştır. Ancak sözü edilen modellerden farklı olarak somut süreç modelinden elde edilmiş olan üst-veri modeli baz alınarak gerçekleştirilmiş olan ManagemOnt ile olmayan ya da bütünüyle tanımlanmamış süreç verilerinden çıkarsama yolu ile yeni bilgiler elde edilebilecektir.

Çıkarsama yolu ile başlanacak yazılım mühendisliği projeleri daha önceden başarılı olmuş projelerin ontolojiler üzerinde kayıtlı olan verileri sorgulanarak oluşturulacaktır. Ontolojiler sayesinde oluşturulmuş olan anlamsal süreç ilişkileri ile birlikte gelecekte yapılacak olan yazılım mühendisliği projeleri geçmişte başarılı olmuş projlerin verileri kullanılarak sağlanabilecek, yazılım mühendisliği yönetim bilgileri doğru veri – doğru bilgi yargısından uzaklaşarak elde veri olmasa da doğru bilgiye ulaşabilmeyi sağlayacaktır.

6. KAYNAKLAR

[1]. Allen, J., *Making a Semantic Web*, http://www.netcrucible.com/semantic.html, 2001

[2]. Barners-Lee, T., *Uniform Resource Identifiers (URI): Generic Syntax*, W3C Publications, http://www.ietf.org/rfc/rfc2396.txt, 1998

[3]. Berners-Lee, T., *What The Semantic Web can Represent*, W3C Publications on Semantic Web, http://www.w3.org/1999/xhtml, 1998

[4]. Berners-Lee, T., *Semantic Web on XML*, W3C Publications on Semantic Web, http://www.w3.org/2000/Talks/1206-xml2k-tbl/Overview.html, 2000

[5]. Berners-Lee, T. et. al., *The Semantic Web*, The Scientific American Publications on Semantic Web, http://www.sciam.com/print_version.cfm?articleID=00048144-10D2-1C7084A9809EC588EF21, 2001

[6]. Brickley D., ve Guha R. V., *Resource Description Framework (RDF) Schema Specification*, W3C Publications on Semantic Web, http://www.w3.org/TR/1999/PR-rdf-schema-19990303/, 1999

[7]. Ceri, S., ve Pelagatti, G., *Distributed Databases*, McGraw Hill, US, 1984

[8]. Curtis, W. et. al., *Process Modelling*, ACM Publications, 1992

[9]. Daconta, M. C., Obrst, L. J. ve Smith, K. T., *The Semantic Web*, Wiley Inc., US, 2003

[10]. Doheny, J. G., ve Filby, I. M., *A Framework and Tool for Modelling and Assessing Software Development Processes*, Artificial Intelligence Applications Institute (AIAI) Publications, 1996

[11]. Dumbill, E., *Building the Semantic Web*, The XML.COM Publications on Semantic Web, http://www.xml.com/pub/a/2001/03/07/buildingsw.html, 2001

[12]. Eric S. K. Yu ve Mylopoulos J., *Understanding why in Software Process Modelling*, Analysis and Design, 16th International Software Engineering Conference Publications, 1994

[13]. Fensel, D., Hendler, J., Lieberman, H. ve Wahlster, W., *Spinning the Semantic Web*, MIT Press, US, 2003

[14]. Gruber, T., *What is an Ontology*, http://ksl-web.stanford.edu/people/gruber/, 1993

[15]. Guha, R. ve McCool R., *TAP: A Semantic Web Platform*, http://tap.stanford.edu/tap.pdf, 2003

[16]. Hawke, S., *How the Semantic Web Works*, W3C Publications on Semantic Web, http://www.w3.org/2002/03/semweb/, 2002

[17]. Hay, D. C., *Data Model Patterns*, Dorset House Publishing, US, 1996

[18]. Humphrey, W. S., Managing the Software Process, SEI Series, US, ISBN: 0-201-18095-2, 1998

[19]. Miller et. al., *Resource Description Framework (RDF) v1.173*, W3C Publications on Semantic Web, http://www.w3.org/RDF/, 2006

[20]. Noy F. N. ve McGuiness D. L., *Ontology Development 101: A Guide to Creating*

Your First Ontology, Stanford Uni. Publications, 2004

[21]. Özsu, M. T. ve Valduriez, P., *Distributed Database Systems*, Prentice Hall, US, 1991

[22]. Palmer, S. B., *The Semantic Web: An Introduction*, W3C Publications on Semantic Web, http://www.w3.org/, 2001

[23]. Palmer, S. B., *The Semantic Web: Taking Form*, Semantic Web Agreement Group Publications, http://infomesh.net/2001/06/swform/, 2001

[24]. Swartz, A., *The Semantic Web in Breadth*, http://logicerror.com/semanticWeb-long, 2002

[25]. Swick, R., *Resource Description Framework (RDF) Model and Syntax Specification*, W3C Publications on Semantic Web, http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/, 1999

[26]. Tannenbaum, A., *Metadata Solutions*, Addison Wesley, US, 2002

[27]. Thayer, R. H. ve Sommerville, I., *Software Engineering Volume 2: Supporting Processes*, IEEE Series, US, 2002

[28]. Tozer, G., *Metadata Management*, Artech House, US, 1999

[29]. Berners-Lee, T. ve Miller, E., *The Semantic Web lifts off*, ERCIM News Issue: 51, ERCIM EIGG, France, 2002

[30]. Antoniou, G., Harmelen, van F., *A Semantic Web Primer*, The MIT Press, UK, ISBN: 0-262-01210-3, 2004

[31]. Passin, B. T., *Explorer's Guide to the Semantic Web*, Mining Co., US, ISBN: 1-932394-20-6, 2004

[32]. Alesso, P. H. ve Smith, C. F., *Thinking on the Web*, Wiley, US, ISBN: 0-471-76814-6, 2006

[33]. Javies, D., Fensel, D., Harmelen, van F., *Towards the Semantic Web*, Wiley, US, ISBN: 0-470-84867-7, 2003

[34]. Powers, S., *Practical RDF*, O'Reilly, US, ISBN: 0-596-00263-7, 2003

[35]. Sumpter, R. M., *Data Management*, Lawrence Livermore National Lab., 1994

[36]. Thangarathinam, T., Wyant, G., Gibson, J. ve Simpson, J., *Metadata Management: The Foundation for Enterprise Information Integration*, Intel Technology Journal Volume: 8 Issue: 04, US, 2004

[37]. Sun Microsystems, *Metadata Management: An Essential Ingredient for Information Lifecycle Management*, Sun Mic., 2005

[38]. NISO Press, *Understanding Metadata*, NISO Press Booklets, US, ISBN: 1-880124-62-9, 2004

[39]. Spyns, P., Meersman, R. ve Jarrar, M., *Data Modelling versus Ontology Engineering*, STARLab, Belgium, 2002

[40]. Kalinichenko, L., Missikoff, M., Schiapelli, F. ve Skvortsov, N., *Ontological Modelling*, Proceedings of the 5th Russian Conference on Digital Libraries RCDL2003, Russia, 2003

[41]. Gardner, S. P., *Ontologies ve Semantic Data Integration*, DDT Volume: 10 Number: 14, 2005

[42]. Cruz, I. F. ve Xiao, H., *The Role of Ontologies in Data Integration*, ADVIS Lab., US, 2005

[43]. Mizoguchi, R. ve Ikeda, M., Towards *Ontology Engineering*, Technical Report AI-TR-96-1, I.S.I.R, Jp, 1998

[44]. Mizoguchi, R., *Tutorial on Ontological Engineering*, I.S.I.R, Jp, 2004

[45]. Ding, Y. ve Fensel, D., *The Thematic Network for the Semantic Web*, ERCIM News Issue: 51, ERCIM EIGG, France, 2002

[46]. Valle, D. E. ve Brioschi, M., *An Ontology-Oriented Solution for Knowledge-Intensive Organization*, ERCIM News Issue: 51, ERCIM EIGG, France, 2002

[47]. Ramakrishnan, R., Gehrke, J., *Database Management Systems 2nd Edition*, McGraw Hill, US, ISBN: 0-07-246535-2, 2000

[48]. Djuric, D., Gasevic, D., Damjanovic, V. ve Devedzic, V., *MDA-based Ontological Engineering*, GOOD OLD AI research Group, Serbia and Montenegro, 2005

[49]. Cranefield, S. ve Purvis, M., *UML as an Ontology Modelling Language*, IJCAI 99 Proceedings, Nz, 1999

[50]. Patel-Schneider, P. F. ve Horrocks, I., *Position Paper: Comparison of Two Modelling Paradigms in the Semantic Web*, WWW2006, US, 2006

[51]. Dieng-Kuntz, R., *Corporate Semantic Webs*, ERCIM News Issue: 51, ERCIM EIGG, France, 2002

[52]. Pahl, C., *Ontologies for Semantic Web Components*, ERCIM News Issue: 51, ERCIM EIGG, France, 2002

[53]. Staab, S., *The Semantic Web Revisited*, IEEE Computer Society 1541-1672/06, Germany, 2006

[54]. Berners-Lee, T., *The Semantic Web Roadmap*, W3C, US, 1998

[55]. Pressman, R. S., *Software Engineering: A Practitioner's Approach 5th Edition*, McGraw Hill, US, ISBN: 0-07-709677-0, 2000

[56]. Pfleger, S. L., *Software Engineering: Theory ve Practice 2nd Edition*, Prentice Hall, US, ISBN: 0-13-029049-1, 2001

[57]. Behrofoz, A. ve Hudson, F. J., *Software Engineering Fundamentals*, Oxford Press, UK, ISBN: 0-19-510539-7, 1996

[58]. Abran, A., Moore, J. W., Bourque, P., Dupuis, R. ve Tripp, L. L., *SWEBOK: Software Engineering Body of Knowledge Trial Version*, IEEE Computer Society, US, ISBN: 0-7695-10000-0, 2001

[59]. Mohame, A. H., Lee, S. P. ve Salim, S. S., *An Ontology-based Knowledge Model for Software Experience Management*, Journal of Knowledge Management Practice, 2004

[60]. Scacchi, W., *Understanding Software Process Redesign Using Modeling*, Analysis and Simulation, Software Process Improvement and Practice, Wiley, US, 2000

[61]. Gnatz M., Marschall, F., Popp, G., Rausch, A. ve Schwerin, W., *Common Meta-Model for a Living Software Development Processes*, Munich Technical University ZEN research project, Germany, 2002

[62]. Ramil, J. F., Lehman, M. M., *Modeling Process Dynamics in Software Evolution Processes*, SCE 99 Proceeding, US 1999

[63]. ISO/IEC 12207, *Software Lifecycle Processes – Implementation Considerations*, IEEE/EIA Guide 12207.2-1997, US, 1998

[64]. ISO/IEC 12207, *Software Lifecycle Processes*, IEEE/EIA Guide 12207.0-1996, US, 1998

[65]. Mi, P. ve Scacchi, W., *A Knowledge-based Environment for Modeling and Simulating Software Engineering Processes*, IEEE Transactions on Knowledge and Data Engineering Volume: 2 No:3, 1041-4347/90/0900-0283$01.00, US, 1990

[66]. OMG, *The Software Process Engineering Metamodel (SPEM)*, OMG Documents: ad/2001-03-08, US, 2001

[67]. Gustafson, D., *Software Engineering*, McGRaw Hill, US, ISBN: 0-07-137794-8, 2002

[68]. Mizoguchi, R., *Tutorial on Ontology Engineering – Part1*, The Institute of Scientific and Indutrial Research, Osaka, 2006

[69]. Mizoguchi, R., *Tutorial on Ontology Engineering – Part2*, The Institute of Scientific and Indutrial Research, Osaka, 2006

[70]. Mizoguchi, R., Tutorial on Ontology Engineering – Part3, The Institute of Scientific and Indutrial Research, Osaka, 2006

[71]. Ulu, B. ve Diri, B., *Software Process Ontology*, International MultiConference of Engineers and Computer Scientists 2007, IMECS 2007, 21-23 Mart 2007, Hong Kong, sf. 1110-1115, ISBN: 978-988-98671-4-0

[72]. Ulu, B. ve Diri, B., *Yazılım Yönetim Ontolojisi*, 3. Ulusal Yazılım Mühendisliği Sempozyumu, UYMS 2007, 27-30 Eylül 2007, Ankara, sf. 103-108, ISBN: 978-9944-89-337-4