“CMMI-DEV” İÇİN BİR ONTOLOJİ VE CMMI ESASLI SÜREÇ DEĞERLENDİRMENİN ONTOLOJİ TABANLI BİR ARAÇ İLE DESTEKLENMESİ

AN ONTOLOGY FOR “CMMI-DEV” AND SUPPORTING CMMI BASED PROCESS ASSESSMENT WITH AN ONTOLOGY BASED TOOL

SEMA GAZEL

Hacettepe Üniversitesi

Lisansüstü Eğitim – Öğretim ve Sınav Yönetmeliğinin

BİLGİSAYAR Mühendisliği Anabilim Dalı İçin Öngördüğü

YÜKSEK LİSANS TEZİ

olarak hazırlanmıştır.

2009

Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Bu çalışma jürimiz tarafından **BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI'nda YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Başkan	:….. Prof.Dr. Ersin Töreci
Üye (Danışman)	:….. Dr. Ebru Sezer
Üye	:….. Yrd.Doç.Dr. Mustafa Ege
Üye	:….. Dr. Özlem Albayrak
Üye	:….. Dr. Ahmet Burak Can

ONAY

Bu tez ...../...../..... tarihinde Enstitü Yönetim Kurulunca kabul edilmiştir.

Prof.Dr. ……..
Fen Bilimleri Enstitüsü Müdürü

"CMMI-DEV" İÇİN BİR ONTOLOJİ VE CMMI ESASLI SÜREÇ DEĞERLENDİRMENİN ONTOLOJİ TABANLI BİR ARAÇ İLE DESTEKLENMESİ

Sema Gazel

ÖZ

Bir yazılım ürününün geliştirilmesinde ve idamesinde kullanılan süreçlerin, bu yazılımın kalitesini büyük oranda etkilediği görüşü genelde kabul görmüş ve bu noktadan hareketle, yazılım geliştirme süreçlerinin iyileştirilmesi ve değerlendirilmesi ile ilgili referans modeller ve standartlar ortaya konulmuştur. Bir kurumun süreçleri, bu kurumun önemli varlıklarından ve bilgi birikimlerindendir. Bunların, referans modeller ve standartlar ile uyumluluğunun izlenmesi, değerlendirilmesi gibi etkinlikler de önemlidir. Süreç odaklı yönetimi benimseyen ve süreçlerini iyileştiren kurumların, bu gibi süreç etkinliklerini desteklemek amacıyla araçlar kullanmasının neredeyse bir zorunluluk olduğu söylenebilir. Bu ihtiyacı karşılamak adına tez kapsamında, CMMI-Dev için bir ontoloji geliştirilmiş ve bu ontolojiyi de kullanan ontoloji-tabanlı bir araç, mevcut bir süreç yönetimi aracı olan EPF aracına ek yetenekler kazandırılarak oluşturulmuştur. OCMQ-E (Ontology-based CMMI Mapping and Querying - EPF) olarak isimlendirilen bu yeni araç ile kurum süreçleri ve CMMI arasındaki uyumluluğunun izlenmesi ve süreç değerlendirme faaliyetinin veri toplama aşamasının desteklenmesi hedeflenmiştir. Bu hedefe ulaşmak için, kurum süreçlerinin ontolojilerinin oluşturulması, süreç ontolojileri ile CMMI ontolojisi arasında eşlemelerin yapılması ve bu eşleme bilgisinin ontoloji biçiminde saklanması sağlanmıştır. Böylece; CMMI alan bilgisi, süreç bilgisi ve CMMI-süreç eşlemeleri ile ilgili bilginin, ontoloji sorguları ile sorgulanabilmesine olanak verilmiştir.

ANAHTAR SÖZCÜKLER: CMMI, ontoloji, yazılım süreç değerlendirme aracı.

Danışman: Dr. Ebru Sezer, Hacettepe Üniversitesi, Bilgisayar Mühendisliği Bölümü, Bilgisayar Mühendisliği Anabilim Dalı

Eş Danışman: Dr. Ayça Tarhan, Hacettepe Üniversitesi, Bilgisayar Mühendisliği Bölümü, Bilgisayar Mühendisliği Anabilim Dalı

AN ONTOLOGY FOR "CMMI-DEV" AND SUPPORTING CMMI BASED PROCESS ASSESSMENT WITH AN ONTOLOGY BASED TOOL

Sema Gazel

ABSTRACT

The premise "the quality of a software product is highly influenced by the quality of the processes used to develop and maintain it" is widely accepted, and from that point of view, some reference models and standards have been developed for process improvement and assessment. Processes being used in an organization are important assets and knowledge for that organization. Process activities, such as monitoring the processes in accordance with reference models and standards, are important. To meet that need, in this thesis scope, an ontology has been developed for CMMI-Dev and an ontology-based tool using the CMMI-Dev ontology has been developed by extending the EPF, which is an existing process management tool. With that new tool, which is named OCMQ-E (Ontology-based CMMI Mapping and Querying - EPF), it is aimed to monitor the accordance between organization's processes and CMMI-Dev, and to support data collection activities in a process assessment. To reach that aim, creating the ontologies of processes of an organization, mapping process ontologies and CMMI ontology and saving the mapping information as an ontology are provided. Thus, it is enabled that CMMI domain knowledge, processes knowledge, and information about CMMI-process mapping are queried by using ontology queries.

KEYWORDS: CMMI, ontology, software process assessment tool.

Advisor: Dr. Ebru Sezer, Hacettepe University, Department of Computer Engineering

Co-advisor: Dr. Ayça Tarhan, Hacettepe University, Department of Computer Engineering

aileme...

TEŞEKKÜR

Bilgi ve deneyimleri ile katkıda bulunan danışman hocalarım Dr. Ebru Sezer'e ve Dr. Ayça Tarhan'a, hiçbir zaman desteklerini esirgemeyen değerli aileme ve dostlarıma, akademik çalışmaları desteklediği için Aselsan A.Ş.'ye en içten teşekkürlerimi sunarım.

İÇİNDEKİLER DİZİNİ

Sayfa

1. GİRİŞ 1

1.1. Problem Tanımı 1
1.2. Tez Kapsamında Geliştirilen Çözüm Önerisi 3
1.3. OCMQ-E Kullanımının Örneklenmesi 6

2. ÖN BİLGİ 8

2.1. Yazılım Süreci 8
2.1.1. Yazılım Süreçleri İçin Referans Modeller/Standartlar 8
2.1.2. Yazılım Süreç İyileştirme 10
2.1.3. Yazılım Süreç Değerlendirme 12
2.1.4. Yazılım Süreç Modelleme 17
2.1.5. SPEM (Software And System Process Engineering Meta-model) 19
2.2. Ontoloji 22
2.2.1. Ontoloji Dilleri 23
2.2.2. Ontoloji Araçları 26
2.2.3. Ontoloji İşlemleri 27
2.3. Kullanılan Araçlar ve Teknolojiler 28
2.3.1. Eclipse 28
2.3.2. EPF (Eclipse Process Framework) 31
2.3.3. Protégé 32
2.3.4. Jena 33
2.4. CMMI: Tümleşik Yetenek Olgunluk Modeli 33
2.4.1. Süreç Alanı (Process Area) 34
2.4.2. Gösterimler (Representations) 37
2.4.3. Seviyelendirme: Yetenek ve Olgunluk Seviyeleri 39

3. İLİŞKİLİ ÇALIŞMALAR 42

3.1. CMMI Ontolojisi Çalışmaları 42
3.2. Süreç Modelleme ve Değerlendirme İle İlgili Ontoloji Tabanlı Çalışmalar 44
3.3. Süreç Değerlendirmeyi Destekleyen Bazı Araçlar 49

4. CMMI ONTOLOJİSİ VE OCMQ-E 52

4.1. CMMI Ontolojisi 55
4.1.1. Amaç ve Kapsam 55
4.1.2. Ontolojinin Geliştirilmesi 55
4.2. OCMQ-E 62
4.2.1. OCMQ-E'nin Kapsamı ve Kullanımı 66
4.2.2. Kullanılan Geliştirme Ortamı 67
4.2.3. Gereksinim Analizi 67
4.2.4. OCMQ-E'nin Mimarisi ve Tasarımı 70

5. ÖRNEK ÇALIŞMA: ASELSAN A.Ş. REHİS GRUBU YAZILIM GELİŞTİRME SÜRECİ 89

5.1. Yazılım Geliştirme Süreci (YGS) 90
5.2. YGS'nin OCMQ-E'de İfade Edilmesi 91
5.3. YGS'nin Ontolojisinin Yaratılması 94

5.4. YGS ile CMMI Pratiklerinin Eşleştirilmesi 95
5.5. Sorgulamalar 97
5.5.1. CMMI Sorguları 97
5.5.2. YGS Sorguları 98
5.5.3. YGS-CMMI Eşlemesi İle İlgili Sorgular 99
6. SONUÇ 102

ŞEKİLLER DİZİNİ

Sayfa

ŞEKİL 1 - YAŞAM DÖNGÜSÜ SÜREÇ GRUPLARI [3]..... 10
ŞEKİL 2 - ISO/IEC 15504 ELEMANLARININ İLİŞKİLERİNE GENEL BAKIŞ [7]. 16
ŞEKİL 3 - SÜREÇ DEĞERLENDİRME MODEL İLİŞKİLERİ [7]..... 17
ŞEKİL 4 - SPEM 2.0 META-MODELİNİN YAPISI [6] 20
ŞEKİL 5 - ECLIPSE PLATFORM MİMARİSİ [54] 30
ŞEKİL 6 - ECLIPSE ÇALIŞMA TEZGAHI (ECLIPSE WORKBENCH) [57]..... 31
ŞEKİL 7 - EPF KULLANICI ARAYÜZÜ..... 32
ŞEKİL 8 - CMMI MODEL BİLEŞENLERİ [8]..... 34
ŞEKİL 9 - BASAMAKLI GÖSTERİMDE GENEL PRATİKLERİN KULLANIMI [8] 36
ŞEKİL 10 - SÜREKLİ GÖSTERİMDE SÜREÇ ALANLARININ DERECELENDİRİLMESİ [8]..... 37
ŞEKİL 11 - BASAMAKLI GÖSTERİMDEKİ OLGUNLUK SEVİYELERİ [8]..... 38
ŞEKİL 12 - HEDEFLENEN PROFİLLER VE DENK SEVİYELENDİRME [8] 39
ŞEKİL 13 - CMMI-SW ONTOLOJİSİNDEKİ BAZI KAVRAMLAR VE ARALARINDAKİ İLİŞKİLER [62] 43
ŞEKİL 14 - CMMI-ACQ ONTOLOJİSİNDEKİ ANA KAVRAMLAR VE İLİŞKİLER [63] 44
ŞEKİL 15 - CMMI-GAAF [66]..... 45
ŞEKİL 16 - PROJECT ASSETS ONTOLOGY (PAO) [66] 46
ŞEKİL 17 - SPO'NUN RDF ÇİZGESİ [67] 47
ŞEKİL 18 - CMMI İÇİN OLUŞTURULAN UZANTI [67]..... 47
ŞEKİL 19 - BASAMAKLI GÖSTERİME ÖZGÜ ONTOLOJİ 56
ŞEKİL 20 - SÜREKLİ GÖSTERİME ÖZGÜ ONTOLOJİ 57
ŞEKİL 21 - BİRLEŞTİRİLMİŞ CMMI ONTOLOJİSİ 58
ŞEKİL 22 - EPF VARSAYILAN PERSPEKTİF: AUTHORING..... 63
ŞEKİL 23 - OCMQ-E EŞLEME PERSPEKTİFİ..... 65
ŞEKİL 24 - OCMQ-E SORGULAMA PERSPEKTİFİ 66
ŞEKİL 25 - OCMQ-E MİMARİSİ..... 71
ŞEKİL 26 - CMMI ONTOLOJİ AĞACI GÖRÜNÜMÜ VE SEKMELER..... 73
ŞEKİL 27 - EŞLEME İŞLEMLERİ İÇİN KULLANICI ARAYÜZÜ GÖRÜNÜMÜ VE DÜĞMELER..... 73
ŞEKİL 28 - PROTÉGÉ EDİTÖR İLE SYO SINIFLARI 75
ŞEKİL 29 - ONTOLOJİLERİN YÖNETİMİ 78
ŞEKİL 30 - ONTOLOJİLİSTESİ.TXT DOSYASININ GÖRÜNÜMÜ..... 78

ŞEKİL 31 - CMMI İLE İLGİLİ SORGU ÖRNEĞİ 79
ŞEKİL 32 - SÜREÇ İLE İLGİLİ SORGU ÖRNEĞİ 80
ŞEKİL 33 - CMMI BİLEŞENİ İLE EŞLENEN SÜREÇ VARLIKLARI 81
ŞEKİL 34 - BİR SÜREÇ VARLIĞI İLE EŞLENEN CMMI BİLEŞENLERİ 82
ŞEKİL 35 - ONTOLOJİLER ARASINDAKİ İLİŞKİ 83
ŞEKİL 36 - SYO İÇİN ÖRNEK BİR BÖLÜM 84
ŞEKİL 37 - SO İÇİN ÖRNEK BİR BÖLÜM 85
ŞEKİL 38 - EO İÇİN ÖRNEK BİR BÖLÜM 86
ŞEKİL 39 - YÖNTEM İÇERİĞİ OLARAK TANIMLANAN YGS GÖREVLERİ 93
ŞEKİL 40 - YGS'NİN OCMQ-E'DEKİ İFADESİ 94
ŞEKİL 41 - YGS'NİN ONTOLOJİSİNİN OLUŞTURULMASI 95
ŞEKİL 42 - CMMI SORGUSU 98
ŞEKİL 43 - SÜREÇ SORGUSU: YGS'DEKİ GÖREVLER 99
ŞEKİL 44 - CMMI BİLEŞENİ İLE EŞLEŞTİRİLMİŞ OLAN SÜREÇ VARLIKLARI 100
ŞEKİL 45 - SÜREÇ VARLIĞI İLE EŞLEŞTİRİLMİŞ OLAN CMMI BİLEŞENLERİ 101

ÇİZELGELER DİZİNİ

ÇİZELGE 1 - IDEAL MODELİN AKTİVİTELERİ [25] 11
ÇİZELGE 2 - PDCA MODELİNİN AKTİVİTELERİ [25] 12
ÇİZELGE 3 - CMMI'DA GÖSTERİMLERE GÖRE SEVİYELERİN İSİMLENDİRİLMESİ 41
ÇİZELGE 4 - CMMI ONTOLOJİSİNDEKİ SINIFLAR 61
ÇİZELGE 5 - YENİ EKLENEN PERSPEKTİFLER, GÖRÜNÜMLER VE EKLENTİLER ARASINDAKİ İLİŞKİ 72
ÇİZELGE 6 - CMMI-SÜREÇ EŞLEME SORGULAMALARI 81
ÇİZELGE 7 - YGS'NİN BAZI ADIMLARI 90
ÇİZELGE 8 - YGS İLE CMMI PRATİKLERİNİN EŞLENMESİ 95

EKLER DİZİNİ

EK-1: SÜREÇ YAPISI ONTOLOJİSİ (SYO): SURECYAPİSİONTOLOJİSİ.OWL... 1

EK-2: BİR SÜREÇ ONTOLOJİSİ (SO) ÖRNEĞİ: YAZİLİM_GELİSTİRME_SURECİ.OWL 6

EK-3: EŞLEME ONTOLOJİSİ (EO) ÖRNEĞİ: CMMI – YGS EŞLEMESİNE İLİŞKİN ESLEMEONTOLOJİSİ.OWL 9

SİMGELER VE KISALTMALAR DİZİNİ

ARC	Appraisal Requirements for CMMI
CMMI	Capability Maturity Model Integration
CMMI-ACQ	Capability Maturity Model Integration for Acquisition
CMMI-DEV	Capability Maturity Model Integration for Development
CMMI-SW	Capability Maturity Model Integration – Software
CMMI-SVC	Capability Maturity Model Integration for Services
DAML	DARPA Agent Markup Language
DARPA	Defence Advanced Research Projects Agency
EPF	Eclipse Process Framework
GG	Generic Goal
GP	Generic Practice
IEC	International Electrotechnical Commission
IPPD	Integrated Product and Process Development
ISO	International Organization for Standardization
OWL	Web Ontology Language
REHİS	Radar, Elektronik Harp ve İstihbarat Sistemleri
SCAMPI	Standard CMMI Appraisal Method for Process Improvement
SEI	Software Engineering Institute
SG	Specific Goal
SP	Specific Practice
SPEM	Software and System Process Engineering Meta-model

1. GİRİŞ

Yazılım geliştirme; analiz, tasarım, gerçekleştirme, test ve bakım gibi etkinliklerden oluşan ve bir yazılım ürününün oluşturulması ile sonuçlanan etkinlikler kümesidir. Birbiri ile ilintili olan bu etkinlikler boyunca bazı hatalar yapılabilmekte ve hataların tespit edilmesi geciktikçe bunları düzeltmenin maliyeti de artmaktadır. Özellikle hataların çoğunun tespiti ve düzeltilmesi test aşamasına bırakıldığında, testler uzadıkça teslimat gecikmekte, ürün kalitesi düşmekte ve bakım maliyeti geliştirme maliyetini aşabilmektedir [1]. Bu problemlerin üstesinden gelebilmek için yazılım kalitesine önem verilmelidir.

Kaliteyi sağlamada iki farklı anlayış vardır; biri geleneksel olan *kalite kontrolü* anlayışı, diğeri ise gelişmiş olan *kalite güvencesi* anlayışıdır. Kalite kontrolünde hataların ayıklanmasına odaklanılmıştır. Kalite kontrolü, bir ürünün veya hizmetin tanımlanmış gereksinimleri karşılayıp karşılamadığının tespit edilmesi için kullanılan tekniklerden ve uygulanan faaliyetlerden oluşmaktadır. Kalite güvencesinde ise hataların önlenmesine odaklanılmıştır. Kalite güvencesi, bir ürünün veya hizmetin tanımlanmış gereksinimleri karşılamasının yeterli derecede güvence altına alınması için gerekli olan, planlanmış ve sistematik faaliyetlerden oluşmaktadır. Kalite, kalite kontrol anlayışında ürün ve/veya hizmet üzerinden sağlanmaya çalışılırken, kalite güvencesi anlayışında ürünü ve/veya hizmeti oluşturan sistem üzerinden sağlanmaya çalışılır [2].

Kalite güvencesi, kalitenin yazılım ürününe yerleştirilmesi için, proje yaşam döngüsü boyunca üretilen ürünlerin ve uygulanan süreçlerin, tanımlı gereksinimlere uyduğunu güvence altına almayı hedefler [3]. Proje yaşam döngüsü boyunca çeşitli etkinlikleri planlayarak, işleterek ve izleyerek hataların erken tespit edilmesine hizmet eder. Tüm bu etkinliklerin sonunda ise, kabul edilebilir hata seviyesinde, gereksinimleri karşılayan, amaçlanan kullanıma uygun, zamanında tamamlanmış, belirlenen bütçe sınırları içinde gerçekleştirilmiş ve bakımı sağlanabilen yazılım ürünlerinin geliştirilmesi hedeflenmektedir [4].

1.1. Problem Tanımı

Bir yazılım ürününün geliştirilmesinde kullanılan süreçler, bu ürününün kalitesini büyük oranda etkiler [5]. Bu dayanak noktasından hareketle, yazılım süreçlerinin

modellenmesi, değerlendirilmesi ve iyileştirilmesi ile ilgili çeşitli çalışmalar ortaya konulmuştur. Yazılım ve sistem geliştirme süreç mühendisliği meta-modeli olan SPEM [6], yazılım ve sistem süreçlerini değerlendirme çatısı olan SPICE [7], süreç referans modeli olan CMMI [8] bunlardan bazılarıdır.

Süreç iyileştirme; Deming'in "planla" – "gerçekleştir" – "kontrol et" – "iyileştir' adımlarının bir döngü halinde kurum süreçlerine uygulanması ile gerçekleşen ve süreklilik gerektiren bir faaliyettir [9]. Süreç odaklı yönetimi benimseyen ve süreçlerini iyileştirmeyi hedefleyen kurumların; süreç varlıklarını tanımlaması, bu tanımlar arasındaki ilişkileri kurması, sürekli iyileştirme sırasında ortaya çıkan değişiklikleri süreç tanımlarına ve süreç varlıkları arasındaki ilişkilere yansıtması ve tüm bunları yapılandırma (versiyon) kontrolü altına alabilmesi önemlidir [10]. Bu nedenle, kurumların bu gibi süreç yönetme faaliyetlerini yerine getirebilmesi için araçlar kullanması neredeyse bir zorunluluktur.

Süreç değerlendirme, süreç iyileştirme faaliyetlerinin önemli adımlarından biridir. Bu adımda, bir kurumun süreç tanımlarının ve uygulamalarının, bir model veya standart ile uyumu irdelenmektedir [7]. Bu irdeleme, süreç tanımlarındaki ve uygulamalarındaki güçlü, zayıf ve/veya eksik noktaların ortaya konulmasını, böylece süreçlerde iyileştirme yapılabilecek noktaların görülebilmesini sağlar. Ayrıca, süreç değerlendirme, kurumların mevcut süreç olgunluk durumlarının tespit edilebilmesi ve belgelendirilmesi açılarından da önem taşımaktadır [7]. Genellikle kurumların kalite bölümleri ve/veya bağımsız tetkikçiler tarafından gerçekleştirilen bu etkinlik; uzmanlık ve dikkat gerektiren, vakit ve emek isteyen bir etkinliktir. Özellikle birden fazla süreç referans modeline ve/veya süreç standardına uymayı hedefleyen kurumlar için bu durum daha da zorlaşmaktadır. Çünkü kurum süreçlerinin her bir model ve/veya standart ile uyumunu ayrı ayrı takip edebilmek, süreçlerde bir değişiklik yapılacağı zaman, bu değişikliğin, izlenen model(ler)den ve/veya standart(lar)dan sapmaya neden olup olmayacağını izleyebilmek gerekmektedir. Bu zorlukların üstesinden gelmek ve süreç değerlendirme faaliyetlerinin uygulanışından kaynaklanan maliyeti ve riskleri azaltmak için, bu adımın da, süreç yönetiminde olduğu gibi, yazılım araçları ile desteklenmesi gerektiği düşünülmektedir.

1.2. Tez Kapsamında Geliştirilen Çözüm Önerisi

Süreç yönetimi ve süreç değerlendirme faaliyetlerinin, aynı araç ile desteklenmesinin getireceği bazı faydalar vardır. Öncelikle, her iki faaliyette de kurumun süreç varlıkları esas alındığından dolayı, bilgi tekrarının neden olabileceği problemlerin önüne geçilebilir. Ayrıca, bilginin tek noktadan yönetilmesi, zaman ve emek kaybını engelleyerek maliyetin düşmesini sağlayabilir.

Yazılım süreçlerinin yönetimini ve değerlendirilmesini destekleyecek böyle bir altyapıyı sağlayabilmek için; 1) Süreç referans modellerini süreç yönetim araçlarının anlayacağı şekilde ifade etmek, 2) Süreç yönetimi aracında yer alan bir kurumun süreç varlıkları ile süreç referans modellerini eşleştirmek, 3) Kurulan bu eşleştirmeleri sorgulayabilmek, uygun bir çözüm olarak görülmektedir. Böyle bir altyapının oluşturulması için ontolojilerden faydalanmanın ise uygun bir yaklaşım olacağı düşünülmüştür. Çünkü ontolojiler, makineler tarafından anlaşılır bir dil ile; kavramların, kavramlar arasındaki ilişkilerin ve bu kavramların olgularının ifade edilebilmesini sağlarlar. Ontolojiler bilginin biçimsel olarak gösterilmesini, bu biçimsel yapılar arasında eşleştirmeler kurularak bunların ilişkilendirilmesini, biçimsel olarak gösterilen bilginin sorgulanmasını sağlayan paylaşılabilir yapılardır. Tüm bunlar göz önünde bulundurulduğunda, ontoloji tabanlı bir yaklaşımın yukarıda bahsedilen üç adımı da destekleyebileceği düşünülmüştür: 1) Süreç referans modellerinin ontolojileri oluşturulduğunda, bu modeller süreç yönetim araçlarının anlayacağı şekilde ifade edilebilmiş olurlar, 2) Ontolojilerin eşleştirilebilir yapılar olmaları, süreç referans modelleri ontolojileri ile kurum süreçleri ontolojileri arasında eşleştirme kurulabilmesine olanak verir, 3) Ontolojilerin sorgulanabilir yapılar olmaları, bir önceki aşamada kurulan süreç ve model ontolojileri arasındaki eşleştirmelerin sorgulanabilmesine olanak verir.

Bunlardan hareketle, tez kapsamında bir CMMI ontolojisi ve CMMI esaslı süreç iyileştirme ve değerlendirme faaliyetlerini destekleyecek ontoloji-tabanlı bir araç oluşturulmuştur. Bu araç, mevcut EPF [11] (Eclipse Process Framework) aracına, geliştirilen Eclipse eklentileri (plug-in) ile yeni yeteneklerin kazandırılması suretiyle oluşturulmuştur. EPF, bir Eclipse Birliği [12] projesidir. Eclipse tabanlı ve açık kaynak kodlu bir süreç yönetimi aracıdır. EPF, aynı zamanda, yazılım ve sistem

süreç mühendisliği meta-modeli olan SPEM'in [6] (Software And System Process Engineering Meta-Model) bir gerçekleştirimidir. SPEM, OMG [13] (Object Management Group) tarafından oluşturulmuş olan, yazılım ve sistem geliştirme süreç mühendisliği için bir meta-model ve bir kavramsal çatıdır. EPF aracının, Eclipse eklentileri ile genişletilebilir yapısı ve SPEM esaslı olması göz önünde bulundurularak, tez kapsamında en baştan bir uygulamanın geliştirilmesi yerine, istenen yeteneklerin eklentiler ile EPF aracına kazandırılması daha uygun bulunmuştur. Eklentiler ile kazandırılan yetenekler ise özetle şunlardır:

- CMMI ontolojisinin görüntülenmesi
- EPF kullanılarak ifade edilen kurum süreçlerinin, istendiğinde ontolojilerinin oluşturulabilmesi
- Süreç varlıkları ile CMMI bileşenleri arasında elle eşleştirmeler kurulmasına izin verilmesi
- Kurulan eşleme bilgilerinin bir ontolojide saklanması
- CMMI ontolojisi, süreç ontolojileri ve süreç-CMMI eşleme ontolojisi üzerinde bazı sorgulamaların yapılabilmesi

Çözümün gerçekleştirimi için sırasıyla aşağıda verilen adımlar gerçekleştirilmiştir:

1) CMMI-Dev v1.2 [8] için bir OWL [14] (Web Ontology Language) ontolojisi oluşturulmuştur. Sistem ve yazılım geliştiren kurumlar tarafından yaygın kabul gören[1] bir süreç referans modeli olan CMMI-Dev, Carnegie Mellon Üniversitesi, Yazılım Mühendisliği Enstitüsü [15] (SEI) tarafından oluşturulmuş bir süreç referans modelidir. Ürünlerin ve hizmetlerin, geliştirilmesinde ve bakımında uygulanan faaliyetleri adresleyen, süreçlerin iyileştirilmesinde ve süreçlerin yeteneğinin veya olgunluğunun değerlendirilmesinde kullanılabilmektedir. CMMI çatısı altında, belli bazı ilgi alanlarının ihtiyaçlarını karşılayan ve takımyıldız (constellation) olarak isimlendirilen farklı modeller oluşturulmuştur. Şu an 3 tane olan takımyıldızlar şunlardır: Geliştirme için CMMI-Dev (CMMI for Development)

[1] CMMI Published Appraisal Results : http://sas.sei.cmu.edu/pars/pars.aspx

[8], hizmetler için CMMI-SVC (CMMI for Services) [16] ve satın alma için oluşturulan CMMI-ACQ (CMMI for Acquisition) [17]. Bu çalışma kapsamında ele alınan takımyıldız, geliştirme için olan CMMI-Dev takımyıldızıdır. Çalışmada oluşturulan CMMI ontolojisi Protégé OWL Editor [18] kullanılarak oluşturulmuş ve OWL dili ile ifade edilmiştir.

2) Oluşturulan CMMI ontolojisinin, EPF arayüzünde görüntülenmesini sağlamak amacıyla, bir Eclipse eklentisi geliştirilmiş ve böylece ontoloji EPF ile bütünleştirilmiştir.

3) EPF ile bütünleştirilmek üzere geliştirilen bir başka Eclipse eklentisiyle, istendiğinde, EPF kullanılarak yazılan bir sürecin ontolojisinin oluşturulması sağlanmıştır. Ayrıca, bu eklenti ile, ontolojisi oluşturulan bir sürecin adımları ile CMMI bileşenleri arasında elle eşleştirme kurulmasına olanak verilmiştir. Grafiksel kullanıcı arayüzünden, tablo-ağaç (tabletree) yapısında görülen süreç adımları ve ağaç yapısında görülen CMMI bileşenleri arasında eşlemeler kurulurken, uygulama; seçilen nesnelerin ontolojilerdeki karşılıklarını bularak, eşleştirmeleri bunlar üzerinden yapmaktadır. Böylece kullanıcı dostu bir arayüz ile eşlemlerin kurulmasına olanak verilmiştir. Ayrıca kurulan her eşleştirme bilgisi uygulama tarafından yine bir ontolojide saklanmaktadır.

Bu eklentiye dahil edilen bir başka yetenek ise ontoloji sorgulamalarıdır. Üç farklı amaç için sorgulama yapılmaktadır; CMMI alan bilgisi ile ilgili sorgulamalar, EPF kullanılarak yazılmış ve ontolojisi oluşturulmuş bir süreç ile ilgili sorgulamalar, CMMI-süreç eşleştirmeleri ile ilgili sorgulamalar. Özellikle eşleştirmelerin sorgulanması, süreç değerlendirme faaliyetlerini destekleyecek olanlardır.

Yeni yetenekler ile genişletilmiş olan EPF, OCMQ-E (Ontology-based CMMI Mapping and Querying - EPF) olarak isimlendirilmiştir ve tez raporu boyunca bu isimle anılacaktır.

1.3. OCMQ-E Kullanımının Örneklenmesi

OCMQ-E'nin kullanımının denenmesi için, CMMI-Dev 3.Seviye Olgunluk Belgesi alma çalışmaları yürütmekte olan Aselsan A.Ş. [19] REHİS (Radar, Elektronik Harp ve İstihbarat Sistemleri) Grubu'nun YGS [20] (Yazılım Geliştirme Süreci) örnek süreç olarak seçilmiştir. Örneklemenin ilk adımında YGS, OCMQ-E'ye orijinal EPF aracından gelen mevcut yetenekler kullanılarak yazılmıştır. EPF ile, süreçler ve bu süreçlerin içerdiği varlıklar ayrı ayrı tanımlanabilmektedir. Böylece EPF, süreç varlıklarından oluşan bir kütüphane oluşturulmasına ve bu varlıkların tekrar kullanılabilirliğine olanak vermektedir. YGS, EPF ile yazılırken, YGS'nin alt aktiviteleri ayrı ayrı süreç varlıkları olarak tanımlanmış, daha sonra tüm bir süreç bu varlıklar kullanılarak ifade edilmiştir. Süreç ifade edildikten sonra, OCMQ-E'ye eklenen yeteneklerden biri olan süreç ontolojisi oluşturma yeteneği ile YGS'nin OWL ontolojisi oluşturulmuştur. Daha sonra, YGS'deki süreç adımları ile CMMI pratikleri arasındaki eşleştirmeler elle kurulmuştur. Kurulan bu eşleştirme bilgileri QCMQ-E tarafından bir ontoloji olarak saklanmaktadır. Son olarak ise, OCMQ-E'deki sorgulama yetenekleri kullanılarak, eşleştirilen ve eşleştirilmeyen CMMI pratikleri sorgulanmış, eşleştirilen CMMI pratiklerinin hangi süreç varlıkları ile eşleştirildiği listelenmiştir. Ayrıca, bir süreç varlığı ile eşleştirilen CMMI bileşenlerinin listelenmesi ile ilgili olarak, YGS'den seçilen bir süreç varlığı ile eşleştirilmiş olan CMMI bileşenlerinin listelendiği görülmüştür. Bir diğer sorgulama olan süreç ontolojisinin sorgulanmasına örnek olarak ise, YGS'deki aktiviteler, görevler, roller ve iş ürünleri listelenmiştir.

Bu tez raporunun ikinci bölümünde, öncelikle tezin dayandığı iki temel konu olan süreç ve ontoloji ile ilgili bilgi verilmiştir. Daha sonra, çalışmada kullanılan araçlara ve teknolojilere değinilmiş ve ikinci bölümün sonunda, çalışmanın ana konularından biri olan CMMI ile ilgili detaylı açıklamalara yer verilmiştir. Üçüncü bölümde, benzer çalışmalar bağlamında, CMMI ontolojisi oluşturulan çalışmalara ve süreç modelleme ve değerlendirme ile ilgili ontoloji içeren çalışmalara yer verilmiş, bunların yapılan çalışma ile olan farklarına değinilmiştir. Ayrıca üçüncü bölümün sonunda, süreç değerlendirmeyi destekleyen bazı araçlardan da bahsedilmiştir. Dördüncü bölümde, tez kapsamında oluşturulan CMMI ontolojisi ve geliştirilen OCMQ-E ayrıntılı olarak anlatılmıştır. Beşinci bölümde, ASELSAN A.Ş.

REHİS Grubundaki YGS kullanılarak OCMQ-E'nin kullanımının örneklenmesine, son bölüm olan altıncı bölümde ise, tez ile ilgili varılan sonuçlara yer verilmiştir.

2. ÖN BİLGİ

Bu tez iki ana konuya dayanmaktadır. Bunlar; s*üreç* ve *ontoloji* konularıdır. İlerleyen bölümde, öncelikle bu iki konuya yer verilmiş ve bunlarla ilgili açıklamalar alt başlıklar halinde "Yazılım Süreci" ve "Ontoloji" başlıklarının altında verilmiştir. Daha sonraki alt başlıkta, çalışmada kullanılan araçlar ve teknolojiler ile ilgili bilgi yer almaktadır. Son olarak ise, tezin dayandığı önemli noktalardan biri olan CMMI-Dev süreç referans modeli ile ilgili detaylı açıklamaların olduğu alt bölüm yer almaktadır.

2.1. Yazılım Süreci

Süreç, girdileri çıktılara dönüştüren birbiri ile ilişkili bir grup aktivitedir [21]. Yazılım süreci ise, bir organizasyonun ya da projenin, yazılım ile ilişkili aktiviteleri yönetmek, yürütmek, izlemek, kontrol etmek ve iyileştirilmek için kullandığı bir ya da bir grup süreçtir [21].

İlerleyen bölümde sırasıyla; yazılım süreçleri için referans modeller ve standartlar, yazılım süreç iyileştirme, yazılım süreç değerlendirme ve süreç modelleme konularına değinilmiştir. Son olarak da, bir yazılım ve sistem süreç mühendisliği çatısı olan SPEM ile ilgili bilgi yer almaktadır.

2.1.1. Yazılım Süreçleri İçin Referans Modeller/Standartlar

Bu bölümde, yaygın kullanılan bir süreç referans modeli olan CMMI ve uluslararası bir yazılım yaşam döngüsü süreçleri standardı olan ISO/IEC 12207 [3] standardına değinilmiştir.

CMMI

CMMI, Carnegie Mellon Üniversitesi, Yazılım Mühendisliği Enstitüsü (SEI) tarafından oluşturulmuş bir süreç referans modelidir. Ürünlerin ve hizmetlerin geliştirilmesinde ve bakımında uygulanan faaliyetleri adresleyen bu model, süreçlerin iyileştirilmesinde ve süreçlerin yeteneğinin veya olgunluğunun değerlendirilmesinde kullanılabilir [8].

Bu çalışma kapsamında bir CMMI ontolojisi oluşturulduğundan dolayı, model ile ilgili detaylı bilgi verme ihtiyacı duyulmuştur. Bundan dolayı, CMMI ile ilgili detaylı bilgi Bölüm 2.4'de ayrı bir başlık altında verilmiştir.

ISO/IEC 12207: Yazılım Yaşam Döngüsü Süreçleri

ISO/IEC 12207 [3], yazılım yaşam döngüsü süreçleri için uluslararası bir ISO [22] (International Organization for Standardization) standardıdır. Standardın, yazılım geliştirme ve bakımı için gerekli tüm işleri tanımlayan bir standart olması amaçlanmıştır. ISO/IEC 12207, sistemlerin, yazılım ürünlerinin ve hizmetlerinin satın alınması esnasında ve daha büyük bir sistemin parçası olarak veya tek başına bir yazılım ürününün temini, geliştirilmesi, işletilmesi, bakımı ve elden çıkarılması süresince; uygulanacak olan süreçleri, etkinlikleri ve görevleri içerir [3]. Bu standart, ISO/IEC 15504 (SPICE) [7] ile uyumlu süreç yetenek değerlendirmesini destekleyen bir süreç referans modeli sağlar. Bu model, belli bir süreç uygulama yaklaşımını temsil etmez ve belli bir sistem/yazılım yaşam döngüsü modelini, yöntemini ve tekniğini salık vermez. Bunun yerine, bir organizasyon tarafından iş ihtiyaçları ve uygulama alanı temel alınarak kullanılması amaçlanmıştır. Sürecin amacına ulaşması için neler yapılması gerektiği tanımlanmıştır. Süreç kapsamındaki etkinlikler ve görevler doğal bir sırada tanımlanmış olsa da bunlar, sürecin nasıl uygulanacağı konusunda yönlendirme yapmaz. Dokümanların adı, biçimi, içeriği ve materyali ile ilgili detaylar tanımlanmamıştır. Sistem ve yazılım mühendisliği – sistem yaşam döngüsü süreçlerini içeren ISO/IEC 15288 [23] standardı ile tam uyum içindedir.

Standardın tanımladığı herhangi bir sürecin amacına ulaşıldığının göstergesi olarak süreç çıktıları kullanılır. Bununla birlikte, çıktılar süreç yeteneğinin belirlemesinde ve süreç iyileştirmelerinin planlanmasında kaynak materyal olarak kullanılırlar.

Standartta, bir yazılım sisteminin yaşam döngüsü boyunca gerçekleştirilen faaliyetler yedi grupta ele alınmıştır. Bunlar;

1 – Kontrat Süreçleri

2 – Kurumsal Proje Olanak Sağlama Süreçleri

3 – Proje Süreçleri

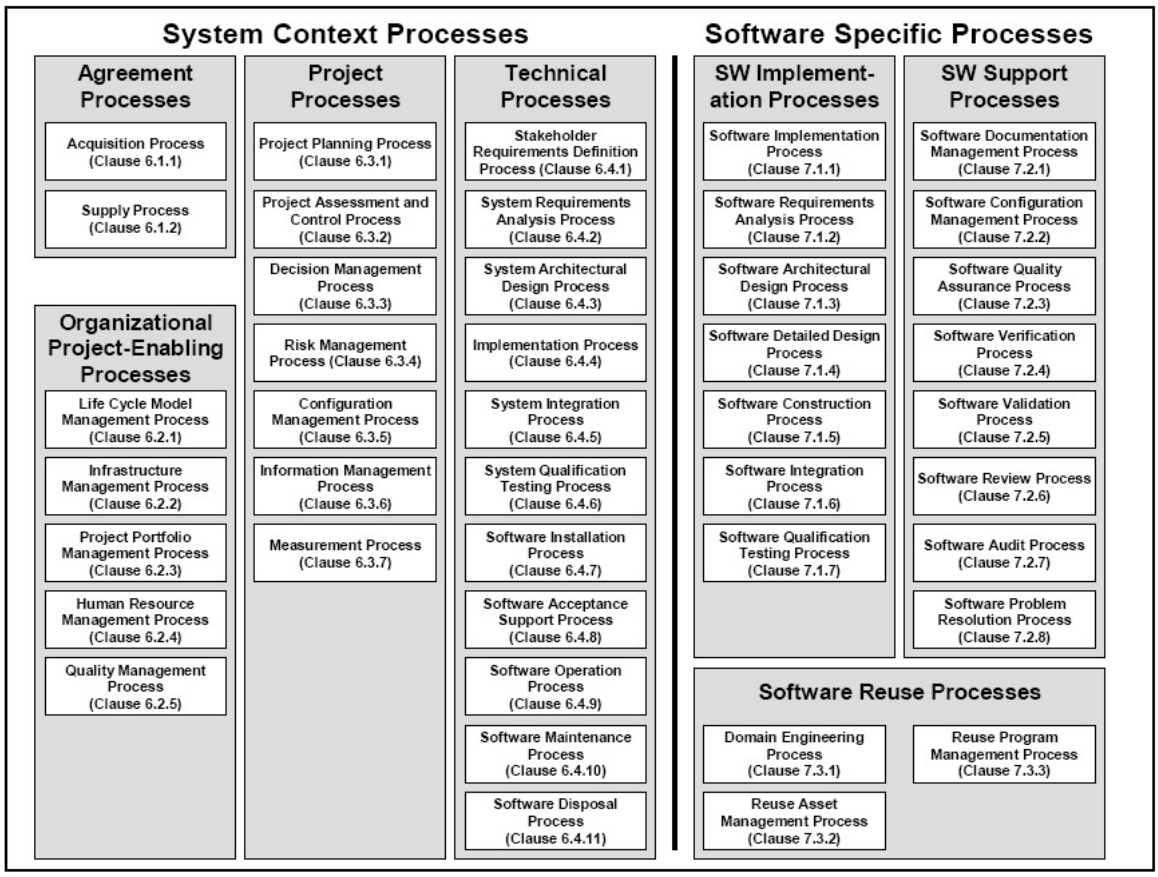
4 – Teknik Süreçler

5 – Yazılım Gerçekleştirim Süreçleri

6 – Yazılım Destek Süreçleri

7 – Yazılım Yeniden Kullanım Süreçleri

Yukarıda verilen ISO/IEC 12207'deki yaşam döngüsü süreç grupları, içerdikleri süreçler ile birlikte Şekil 1'de görülmektedir.



Şekil 1 - Yaşam Döngüsü Süreç Grupları [3]

2.1.2. Yazılım Süreç İyileştirme

Süreç iyileştirme, bir kurumun iş ihtiyaçlarını karşılamak ve iş hedeflerine daha etkin bir şekilde ulaşmasını sağlamak amacıyla, kurumun süreçlerini değiştirmek için yapılan faaliyetlerdir [21]. Bu bölümde, süreç iyileştirme ile ilgili öne çıkan modeller olan IDEAL [24] (Initiating, Diagnosing, Establishing, Acting, Leveraging) ve PDCA [25] (Plan, Do, Check, Act) ile ilgili kısaca bilgi verilmektedir.

IDEAL

IDEAL modeli, SEI tarafından geliştirilmiş olan bir organizasyonel gelişim modelidir [24]. Bu model, gelişim için disiplinli bir mühendislik yaklaşımı sağlar, gelişim programının yönetimine odaklanır ve uzun süreli gelişim stratejisi için temel

oluşturur. Model adını, tanımladığı beş fazın baş harflerinden almaktadır. Bu fazlar:

1 – Başlangıç (Initiating)

2 – Teşhis etme (Diagnosing)

3 – Kurma (Establishing)

4 – Yapma (Acting)

5 – Eğilim (Leveraging)

Her faz, daha alt görevlere de bölünebilen birçok görevden oluşur [25]. Her fazda yer alan ana aktiviteler Çizelge 1'de verilmiştir.

Çizelge 1 - IDEAL Modelin Aktiviteleri [25]

Faz	*Aktiviteler*
Başlangıç	- Süreç iyileştirme hakkında bilgilen - Başlangıç kaynakları ata - Yazılım süreç iyileştirme için altyapıyı kur
Teşhis etme	- Mevcut süreç olgunluk seviyesini belirle - Süreç tanımlarını ve metriklerini oluştur
Kurma	- Yazılım süreç iyileştirme hedeflerini ve önceliklerini oluştur - Faaliyet planlamasını yap
Yapma	- Süreçteki problemleri araştır ve çözüm geliştir - Başarılı süreç iyileştirmeleri tüm organizasyona genişlet/yay
Eğilim	- Önceki yazılım süreç iyileştirme faaliyetlerinden çıkarılan dersleri özetle - Bir sonraki iyileştirme döngüsü için kaynakları belirle

PDCA

'Plan-Do-Check-Act cycle' (Planla – Uygula – Kontrol Et – Önlem Al döngüsü) olarak da bilinen PDCA modeli, ilk kez Walter Shewhart tarafından 1920'lerde

geliştirilmiş ve daha sonra Edwards Deming tarafından popülerlik kazandırılmıştır [25]. Bu model, bir organizasyonun kalite yönetim sisteminin geliştirilmesi, uygulanması ve iyileştirilmesi için bir yaklaşım olarak kullanılabilir [25]. PDCA modeli dört faz ve yirmi iki adımdan oluşmaktadır [25]. PDCA, bir sürekli iyileştirme döngüsüdür. Bir PDCA döngüsü yürütülürken fark edilen herhangi bir gelişim, bir sonraki PDCA döngüsünün gelişim hedefi için temel oluşturmaktadır [9]. ISO (International Organization for Standardization), bu modelin aktivitelerini Çizelge 2'de verildiği gibi tanımlar [25].

Çizelge 2 - PDCA Modelinin Aktiviteleri [25]

Faz	*Aktiviteler*
Planla (Plan)	Hedefleri ve süreçleri oluştur, organizasyonun politikası ve müşteri gereksinimleri ile uyumlu olacak şekilde sonuçları dağıt.
Uygula (Do)	Çözümleri geliştir, bazı çözümleri pilot olarak uygula, seçilen çözümleri gerçekleştir.
Kontrol et (Check)	Süreçleri ve ürünleri; politikalara, hedeflere ve gereksinimlere karşı izle, ölç ve sonuçları raporla.
Önlem al (Act)	Organizasyonun süreç performansını sürekli iyileştirebilmesi için çözümü standartlaştır.

2.1.3. Yazılım Süreç Değerlendirme

Bir kurumun yazılım süreçlerinin, referans alınan bir modele karşı uyumluluğunun disiplinli bir şekilde değerlendirilmesidir [21]. Bu bölümde, yaygın kullanımı olan bir süreç değerlendirme yöntemi olan SCAMPI (Standard CMMI Appraisal Method for Process Improvement) [26] ve bir süreç değerlendirme çatısı olan ISO/IEC 15504 [7] ile ilgili kısaca bilgi verilmektedir.

SCAMPI

SCAMPI (Standard CMMI Appraisal Method for Process Improvement), CMMI modellerine uyumluluğun değerlendirilmesi ve kalite derecelendirmesi yapılması için tasarlanmış bir değerlendirme yöntemidir [26]. SCAMPI'de süreç değerlendirme; A, B ve C sınıfı olmak üzere 3 farklı yöntem içermektedir [26].

A Sınıfı Değerlendirme: En kapsamlı olan SCAMPI değerlendirmesidir. Bu sınıf değerlendirmede hem dokümanlar hem de kişilerle yapılan görüşmeler nesnel kanıt olarak toplanmaktadır. SEI (Software Engineering Institute) onaylı bir tetkikçi liderliğinde yapılması zorunludur. Değerlendirme ekibi en az 4 kişiden oluşmalıdır. Organizasyonel birimin kapsanması bir zorunluluktur. Yalnız A sınıfı bir değerlendirme sonunda sertifikalandırma yapılabilmektedir. SCAMPI A, CMMI değerlendirme yöntemi gereksinimlerini [27] tam olarak karşılamaktadır [26].

B Sınıfı Değerlendirme: Bu sınıf değerlendirmede hem dokümanlar hem de kişilerle yapılan görüşmeler nesnel kanıt olarak toplanmaktadır. SEI onaylı bir tetkikçi liderliğinde yapılması zorunlu değildir, eğitimli ve tecrübeli bir kişi liderliğinde de yapılabilmektedir. Değerlendirme ekibi en az 2 kişiden oluşmalıdır. Organizasyonel birimin kapsanması bir zorunluluk değildir. Bu sınıf değerlendirme sonunda sertifikalandırma yapılamaz [26].

C Sınıfı Değerlendirme: Süreçlerdeki eksiklerin analiz edildiği en az kapsamlı SCAMPI değerlendirmesidir. Bu sınıf değerlendirmede sadece dokümanlar ya da sadece kişilerle yapılan görüşmeler kanıt olabilir. SEI onaylı bir tetkikçi liderliğinde yapılması zorunlu değildir, eğitimli ve tecrübeli bir kişi liderliğinde de yapılabilmektedir. Değerlendirme ekibi yalnız 1 kişiden oluşabilir. Organizasyonel birimin kapsanması zorunluluk değildir. Bu sınıf değerlendirme sonunda sertifikalandırma yapılamaz [26].

SCAMPI A; süreç iyileştirme etkinliklerinde, kurumun olgunluk seviyesinin belirlenmesinde, tedarikçi seçiminde ve süreçlerin izlenmesinde (monitor) kullanılabilir. Bu yöntem bir enstrüman ya da bir araç kullanımını zorlamamakla birlikte, böyle bir kullanım CMMI'ın organizasyondaki uygulanışının daha iyi anlaşılmasını sağlayacak ve değerlendirme etkinliği, kanıtların keşfi yerine kanıtların doğrulanması esasına göre olacaktır. Böylece değerlendirmenin etkili bir şekilde yapılmasına yardım edecektir [26].

SCAMPI A; değerlendirme için plan yapılması ve hazırlanılması, değerlendirmenin yürütülmesi/yönetilmesi ve sonuçların raporlanması olmak üzere üç fazdan oluşmaktadır. Bu yöntemin temel karakteristikleri ise şunlardır; doğruluk (accuracy), tekrarlanabilirlik (repeatability), maliyet/kaynak etkinlik (cost/resource

effectiveness), sonuçların anlamlılığı (meaningfulnes of results) ve CMMI değerlendirme gereksinimlerine uyumluluk (ARC – Appraisal Requirements for CMMI – compliance) [26].

Referans model pratiklerinin uygulandığının göstergeleri *Practice Implementation Indicators* (PII) olarak isimlendirilir [26]. Üç tip PII vardır. Bunlar:

> Doğrudan Çıktı (Direct Artifact): Bir özel ya da genel pratiğin gerçekleştirilmesinin sonucu olan somut çıktıdır.
>
> Dolaylı Çıktı (Indirect Artifact): Bir özel ya da genel pratiğin uygulanışında ortaya çıkan veya bir pratiğin gerçekleştirildiğini doğrulayan çıktıdır. Ancak, pratiğin uygulanış amacı bu çıktıyı oluşturmak değildir.
>
> Teyit (Affirmation): Bir özel ya da genel pratiğin uygulandığını ya da uygulanmadığını onaylayan ya da destekleyen sözlü veya yazılı ifadelerdir.

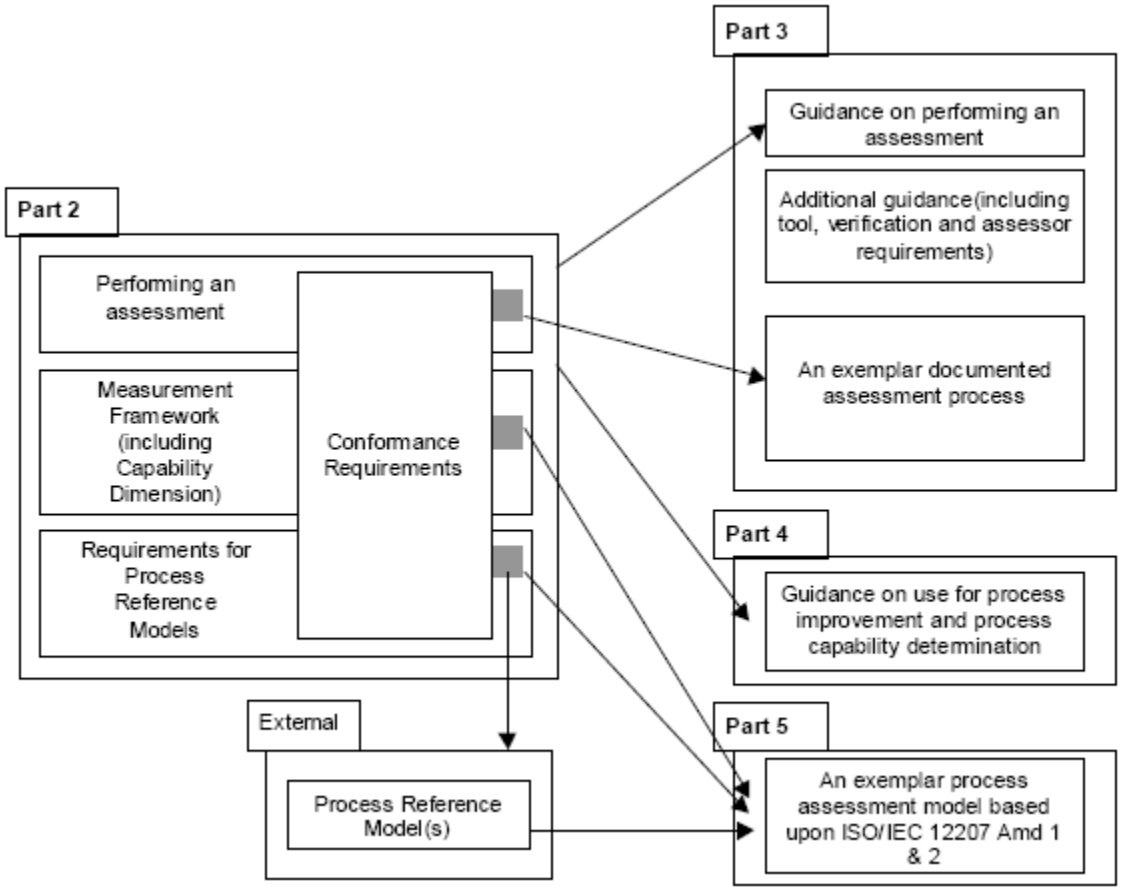
Değerlendirme ekibi, bu gösterge tiplerine göre verileri toplar ve bunları düzenler. SCAMPI A yöntemi, her bir gösterge tipi için toplanması gereken veri miktarı ile ilgili kurallar ve rehberler tanımlamaktadır [26]. Süreç değerlendirme, toplanan bu verilerin doğrulanması, doğrulama sonrasında eldeki verilere göre derecelendirmenin yapılması ve sonucun raporlanması ile sonlandırılmaktadır.

ISO/IEC 15504

ISO ve IEC (International Electrotechnical Commission) tarafından oluşturulmuş uluslararası bir standarttır. SPICE (Software Process Improvement and Capability dEtermination) olarak da bilinen ve yazılım geliştirme süreçlerinin değerlendirilmesi için kullanılan bir çatıdır [7]. Bu çatı, organizasyonlar tarafından, yazılımların satın alınması; tedariği; geliştirilmesi; işletimi; bakımı; ve desteği ile ilgili planlama; yönetme; izleme; kontrol etme; ve iyileştirme etkinlikleri için kullanılabilir [7]. ISO/IEC 15504, satın alma yapanların, tedarikçilerin ve süreç tetkikçilerinin ihtiyacını karşılamak için tasarlanmıştır. Satın alma yapanlar için, tedarikçi süreçlerinin mevcut ve potansiyel yeteneğinin belirlenebilmesinde kullanılabilir. Tedarikçiler için, kendi süreçlerinin mevcut ve potansiyel yeteneğinin, süreç iyileştirme alanlarının ve önceliklerinin belirlenebilmesinde faydalı olabilir.

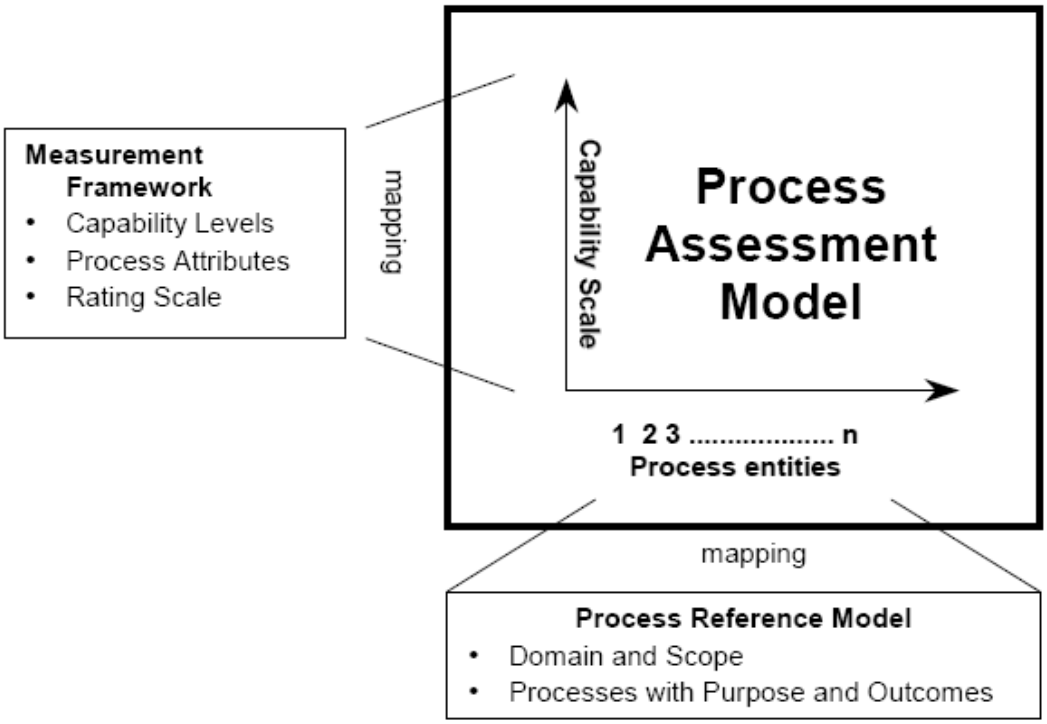
Ayrıca, süreç iyileştirme için yol haritası da çatıda tanımlanmıştır. Tetkikçiler ise, bu çatıyı değerlendirme yönetmek/yürütmek için kullanabilirler [7].

ISO/IEC 15504 [7] yedi parçadan oluşmaktadır. Bunlardan birincisi (Part 1) bilgilendirici bir parçadır. Standardı oluşturan parçaların nasıl bir araya getirilmesi gerektiğini tarif eder, bunların seçimi ve kullanımı için rehberlik sağlar. Ayrıca ISO/IEC 15504 için terimler ve tanımlar içerir. İkinci parça (Part 2) kuralcı (normatif) bir parçadır. Süreç değerlendirmenin ve süreç değerlendirmedeki süreç referans modelinin gereksinimlerini ortaya koyar. Ayrıca, süreç yeteneğinin değerlendirilmesi için bir ölçüm çatısı tanımlar. Bu ölçüm çatısında süreçlerin yetenek seviyesi altı basamakta ifade edilmiştir. Üçüncü parça (Part 3) ISO/IEC 15504'e göre yapılacak bir süreç değerlendirme için genel bakışın sağlandığı, değerlendirmenin gereksinimlerinin karşılanmasına rehberlik eden bilgilendirici bir parçadır. Dördüncü parça (Part 4), süreç iyileştirme ve süreç yetenek belirlenmesinde, süreç değerlendirmeden faydalanılması ile ilgili rehberlik sağlayan bilgilendirici bir parçadır. Beşinci parça (Part 5) ISO/IEC 12207'deki süreç referans modeli esas alınarak, süreç değerlendirmenin yapılışı ile ilgili bir örnek model sağlayan bilgilendirici parçadır. Altıncı parça (Part 6) ISO/IEC 15288'deki süreç referans modeli esas alınarak, sistem yaşam döngüsü süreçleri için örnek bir süreç değerlendirme modeli tanımlayan bilgilendirici parçadır. Bu değerlendirme modeli, 15504 ikinci parçada belirtilen süreç değerlendirme modeli gereksinimleri ile uyumludur. Yedinci parça (Part 7) organizasyonel olgunluğun değerlendirilmesi için koşulları tanımlar [7]. İkinci, üçüncü, dördüncü ve beşinci parçalar arasındaki ilişkiler, Şekil 2'de verilmiştir.



Şekil 2 - ISO/IEC 15504 Elemanlarının İlişkilerine Genel Bakış [7]

Süreç değerlendirme, seçilen süreçlerin, seçilen değerlendirme model(ler)ine karşı değerlendirilmesi ile yapılır. Değerlendirme model(ler)i, ISO/IEC 15504'ün ikinci parçasında tanımlanan gereksinimler ile uyumlu olmalıdır. Süreç referans modeli, ilgi alanına göre seçilir. Örneğin, yazılım mühendisliği alanında ISO/IEC 12207 içindeki süreç modelleri bunun için uygundur. Şekil 3'te süreç referans modeli, değerlendirme modeli ve ölçüm çatısı arasındaki ilişkiler görülmektedir.



Şekil 3 - Süreç Değerlendirme Model İlişkileri [7]

Değerlendirme süreci en az beş aktivite içerir; planlama, veri toplama, veri doğrulama, derecelendirme ve raporlama. Değerlendirme, ISO/IEC 15504 çatısının 3. parçasında tanımlanan nitelikleri taşıyan uzman bir kişinin liderliğinde ve bir takım tarafından yapılmalıdır.

Tez kapsamında yapılan çalışma ile temelde yazılım süreç değerlendirmenin veri toplama aşaması desteklenmiştir. Geliştirilen araç ile, CMMI süreç alanları ile kurum süreçleri arasında elle eşleştirmeler kurulabilmekte ve kurulan bu eşleştirmeler sorgulanabilmektedir. Sorgulamalar ile modelin kurumda ele alınışı ile ilgili bulgular eldeki eşleme verisinden otomatik olarak çekilebilir. Elde edilen bulgular, takip eden doğrulama, derecelendirme ve raporlama aşamalarının otomatik olarak yapılmasını sağlamasa da bu aşamaların elle gerçekleştirilmesini destekleyecek altyapıyı sağlamaktadır.

2.1.4. Yazılım Süreç Modelleme

Süreç modelleme; bir sürecin mimarisinin, tasarımının veya tanımının soyut gösterimidir [28]. Süreç modelleri sürecin analizine, anlaşılmasına veya kestirimine yardım edebilir. Bir süreç modeli analiz edilebilir, doğrulanabilir ve eğer oynatılabiliyorsa (enactable) modellenen sürecin benzetimi yapılabilir [28].

Curtis et al. [29] süreç modellemenin kullanımını ve faydalarını beş ana başlıkta ifade etmiştir: 1) Süreçlerin anlaşılmasına ve iletişimine hizmet eder. 2) Süreç iyileştirmeyi destekler. 3) Yazılım süreç yönetimini destekler. 4) Süreç rehberliğinin otomatikleşmesini destekler. 5) Süreçlerin işletiminin otomatikleşmesini destekler.

Yazılım süreçlerinin modellenmesi ile ilgili pek çok çalışma vardır [30]. Örneğin, Petri Nets, sonlu durum makineleri (finite state machines), ve veri akış diyagramları yazılım süreçlerinin modellenmesi için kullanılmıştır. Farklı türdeki süreç modelleri farklı kullanımlar için tavsiye edilebilir. Herhangi bir modelin, ancak belli bazı bağlamlarda faydalı olması, diğerlerinde ise daha az yardımcı olması beklenmelidir. Bir yazılım sürecinin farklı yönlerinin anlaşılabilmesi için, genellikle farklı modellere ihtiyaç duyulur [30].

Curtis et al. [29] yazılım süreçlerini uygun bir şekilde tanımlamak için, birçok farklı biçimdeki bilginin bütünleştirilmesi gerektiğini, genellikle insanların bir süreç modelinden çekmek istedikleri farklı biçimdeki bilginin; ne yapılacak, kim yapacak, ne zaman ve nerede yapacak, nasıl ve niçin yapacak, bunun yapılması kime bağlı sorularının cevaplarına karşılık geldiğini belirtmiştir. Süreç modelleme dilleri ve gösterimleri yazılım süreçleri ile ilgili bir veya daha fazla bakış açısına cevap verebilir; bunların en yaygın dördü ise *fonksiyonel*, *davranışsal, organizasyonel* ve *bilgisel* bakış açılarıdır [29].

Kellner et al. [31], ideal bir yazılım süreç modelleme yaklaşımı için gereksinimleri oluşturmak istemiştir. Bu amaçla, bir yazılım süreç modelinde olması istenen karakteristikleri tanımlamış ve listelemiştir. Bu karakteristiklerin bazıları aşağıda verilmiştir.

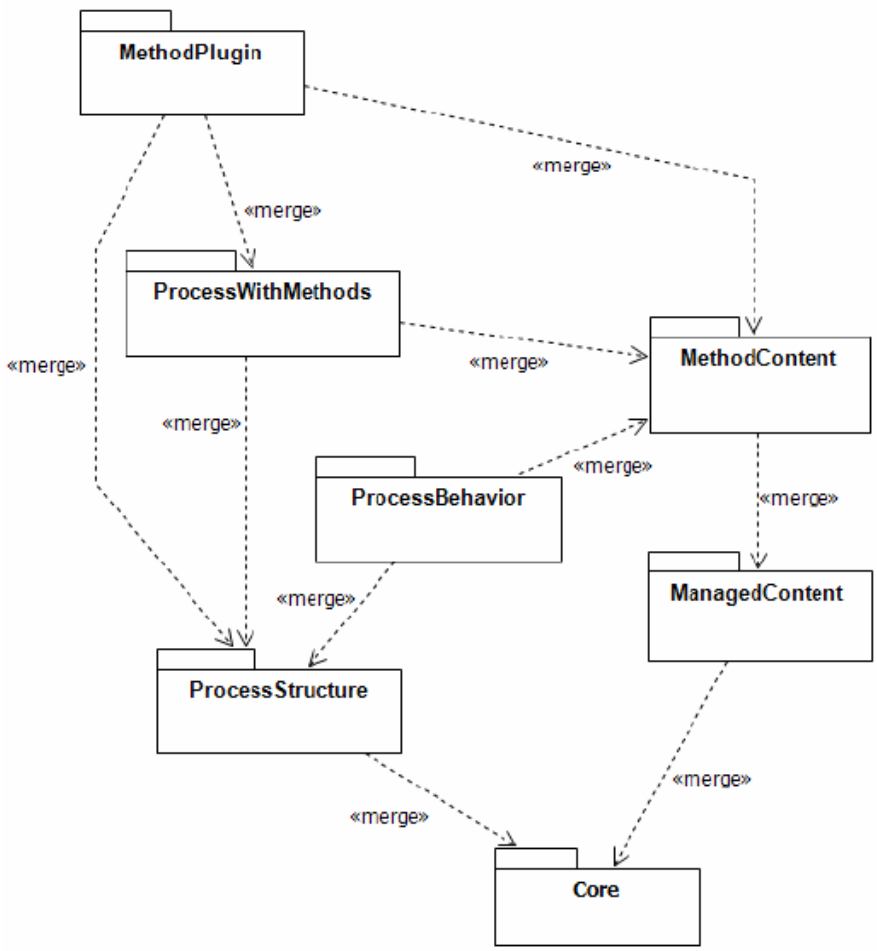
- Bilginin gösterimi için görsel yaklaşımlar (örneğin diyagramlar) kullanılmalıdır.
- Özet tanımlar yapılmasına izin vermelidir.
- Süreçleri çoklu ve birbirini bütünleyen bakışlarla desteklemelidir. Onlar bu gereksinim için faydalı olacağını düşündükleri dört bakış açısı tanımlamışlardır. Bunlar; fonksiyonel, davranışsal, gerçekleştirim ve kavramsal veri modellemedir.

- Farklı bakış açıları için farklı seviyelerde soyutlamayı desteklemelidir.
- Yapı hesaplanabilir olsun diye, sözdizimsel ve anlamsal olarak biçimsel tanımı yapılabilmelidir.
- Kapsamlı analiz yeteneği sağlamalıdır.
- Süreç tanımından süreç davranışının benzetiminin yapılabilmesine hizmet etmelidir.

2.1.5. SPEM (Software And System Process Engineering Meta-model)

SPEM [6], OMG [13] tarafından oluşturulan, bir süreç mühendisliği meta-modeli ve bir kavramsal çatıdır. Bu kavramsal çatı, geliştirme yöntemlerinin ve süreçlerinin; modellenmesi, dokümantasyonu, sunumu, yönetilmesi, karşılıklı değişimi ve uygulamaya alınması ile ilgili kavramlar sağlamaktadır. MOF [32] uyumlu olan bu meta-model, yazılım ve sistem süreçlerinin ve bunların bileşenlerinin tanımlanması için kullanılır. Meta-modelin kapsamı, herhangi bir yazılım ve sistem geliştirme süreci tanımlamak için gerekli olan elemanlar ile sınırlandırılmıştır. Belli bir alana veya disipline (örneğin proje yönetimi) özel nesne eklenmesinden kaçınılmıştır. Farklı üslûplar, kültürel geçmişler, biçimsellik seviyeleri, yaşam döngüsü modelleri ve topluluklar tarafından kullanılan geliştirme yöntemlerini ve süreçlerini ifade edebilecek şekilde olması amaçlanmıştır. Bununla birlikte, meta-modelin odağı geliştirme projeleridir. SPEM genel bir süreç modelleme dili değildir.

SPEM 2.0, Şekil 4'te de görüldüğü gibi, yedi ana paket içinde yapılandırılmıştır. Bu yapılanma ile model mantıksal birimlere bölünmüştür. Her birim, bağlı olduğu birimleri, sağladığı ek yapılar ve yetenekler ile genişletir. Sonuç olarak; daha alt katmanda yer alan birimler, üst birimler olmaksızın da gerçekleştirilebilir.



Şekil 4 - SPEM 2.0 Meta-Modelinin Yapısı [6]

Meta-modeldeki paketler hakkındaki özet bilgi aşağıda verilmiştir.

Çekirdek (Core): Çekirdek meta-model paketi, diğer tüm meta-model paketlerindeki sınıflar için temel oluşturan meta-model sınıflarını ve soyutlamalarını içerir.

Süreç Yapısı (Process Structure): Tüm süreç modelleri için temel tanımlar. Basit ve esnek süreç modelleri oluşturmayı destekler. Bu paketteki çekirdek veri yapısı bir kırınım (breakdown) veya her bir aktivitenin ilgili girdi/çıktı iş ürünleri ve roller ile ilişkilendirildiği iç içe aktivitelerin ayrıştırılmasıdır.

Süreç Davranışı (Process Behavior): Bu paket, süreç yapısı paketindeki durağan yapıların davranışsal modeller ile genişletilebilmesine olanak verir. Ancak, kendine ait bir davranış modelleme yaklaşımı tanımlamaz. Bunun yerine SPEM dışında tanımlanmış davranış modellerine bağlantılar sağlar. Örneğin, süreç yapısı kavramları ile tanımlanmış bir süreç, bu sürecin davranışını gösteren bir UML

aktivite diyagramı ile veya yöntem içeriği paketindeki bir iş ürünü tanımı, bu iş ürününün tipik yaşam döngüsünü gösteren bir durum makinesi (state machine) modeli ile ilişkilendirilebilir.

Yönetilen İçerik (Managed Content): Birçok durumda, geliştirme süreçleri modellerle değil, doğal dil tanımlamaları ile belgelenmekte ve yönetilmektedir. Bu paket, tanımların metinsel içeriğinin yönetilmesi için kavramlar sunmaktadır. Bu kavramlar, tek başına ya da süreç yapısı kavramları ile birlikte kullanılabilirler.

Yöntem İçeriği (Method Content): Bu paket, meta-model kullanıcılarının ve organizasyonların, herhangi bir özel süreç ya da geliştirme projesinden bağımsız olarak, bir geliştirme yöntemleri bilgi tabanı oluşturmaları için kavramlar sağlar. Paket, süreçten bağımsız olarak, yazılım geliştirme yöntemleri, teknikleri ve en iyi pratiklerin somut gerçekleşimi ile ilgili, yeniden kullanılabilir yöntem içerik elemanları için kavramlar ilâve eder. Yazılım geliştirmenin belli bir adımındaki hedefe ulaşmak için, bir işin, içinde yer alacağı geliştirme yaşam döngüsünün neresinde yer alacağından bağımsız olarak, hangi roller, kaynaklar ve sonuçlar ile nasıl bir ilişki içinde olduğunu tanımlamaya olanak verir. SPEM, yöntem içeriği tanımları ile yöntem içeriğinin geliştirme süreçlerindeki uygulanışını net bir şekilde birbirinden ayırmıştır. SPEM kullanıcıları, süreç yaratmaksızın, sadece geliştirme yöntemlerinin bilgi tabanını inşa etmek ve genel rehberlik sağlamak için, yöntem içeriği tanımlayabilirler. Geliştirme süreçleri, yeniden kullanılabilir yöntem içeriğini temel alabileceği gibi, bunları kullanmadan da oluşturulabilir. Geçici veya bir defaya mahsus olan süreçler ve süreç adımları, bu şekilde yöntem içeriği kullanılmadan tanımlanabilir.

Yöntemle Birlikte Süreç (Process with Methods): Bu paket, süreç yapısı paketindeki kavramlarla tanımlanmış süreç ile yöntem içeriği paketindeki kavramların olgularını bütünleştirmek için, olmayan yapıları ilk olarak ve mevcut yapıları yeniden tanımlar. Yöntem içeriği, yazılım geliştirme için temel yöntemleri ve teknikleri tanımlarken, süreçler bu yöntemleri ve teknikleri bir yaşam döngüsü modeli bağlamında kullanır.

Yöntem Eklentisi (Method Plugin): Bu paket, yöntem içeriği ve süreçlerden oluşan; bakım yapılabilir, büyük boyutlu, yeniden kullanılabilir ve ayarlanabilen kütüphaneler veya depolar tasarlamak ve yönetmek için kavramlar sunar.

SPEM 2.0 belirtiminde (specification) üç uyum noktası tanımlanmaktadır. Bunlar; "SPEM Complete", "SPEM Process with Behavior and Content" ve "SPEM Method Content" uyum noktalarıdır. SPEM'in gerçekleştirimini yapanlara önerilen, eğer diğer geliştirme yapanlarla başarılı bir şekilde veri değişimi yapabilmeyi hedefliyorlarsa, belli bir uyum noktası seçerek buna uygun gerçekleştirim yapmalarıdır. Bununla birlikte belirtim, gerçekleyicileri meta-modeldeki istedikleri paketleri seçerek gerçekleştirim yapacak şekilde serbest de bırakmıştır.

Bu meta-model temel alınarak geliştirilmiş araçlar vardır. SPEM'in gerçekleştirimi olan bu araçlar, geliştirme yapan organizasyonların veya tek tek projelerin, süreçlerinin geliştirilmesi veya idamesinden sorumlu olan kişiler (süreç mühendisleri, proje liderleri, proje ve program yöneticileri vb.) tarafından kullanılabilir. Birçok büyük kuruluş tarafından temel alınarak gerçekleştirimi yapılan bu meta-modelin [6] bir uygulaması da Eclipse Birliği tarafından oluşturulmuş olan EPF çatısıdır. Bu çatı ile ilgili detaylı bilgi Bölüm 2.3.2.'de yer almaktadır.

Tez kapsamında, kurum süreçlerinin ontolojisinin oluşturulması ile ilgili hedef doğrultusunda, SPEM'in kullanılması, bu meta-modelin tezin amacı ile uygunluğu, serbest kullanımı ve yaygın kabul görmüş olması göz önünde bulundurularak tercih edilmiştir. Tezde, SPEM'deki Yöntemle Birlikte Süreç paketinin bir alt kümesi esas alınarak bir ontoloji (Süreç Yapısı Ontolojisi – SYO) oluşturulmuştur. Bu alt küme; *Delivery Process, Activity, Iteration, Milestone, Phase, Role Descriptor, Task Descriptor, Team Profile* ve *Work Product Descriptor* nesnelerini içermektedir. Kurum süreçlerinin ontolojileri ise, SYO temel alınarak oluşturulmuştur. Bunun dışında, SPEM'in bir gerçekleştirimi olan EPF aracı genişletilerek OCMQ-E oluşturulduğundan, meta-model dolaylı olarak da kullanılmaktadır.

2.2. Ontoloji

Ontolojiler, belli bir bilgi alanında yer alması muhtemel olan nesneler, bu nesnelerin nitelikleri ve nesneler arasındaki ilişkiler ile ilgili içerik teorileridir [33].

Bunlar, bir alan ile ilgili bilgimizi tanımlamamız için potansiyel terimler sağlarlar. Ontolojik analiz, bilginin yapısını netleştirir ve ontolojiler bir alandaki bilginin paylaşılabilemesini sağlar [33]. Belli bir alandaki bilgi, bu alanda bilgiye ihtiyaç duyan diğer kişiler ile paylaşılabildiğinden, bilginin analizi sürecinin tekrar tekrar yapılmasına gerek kalmaz. Paylaşılabilir bir dil ile ifade edilmiş olan bir ontoloji, o alana özel bilginin gösterimi için bir temel oluşturabilir [33].

Bilgi tabanlı sistemler, geleneksel uygulamalarda; heterojen donanım platformları, programlama dilleri ve ağ protokolleri üzerindedir. Bununla birlikte, bilgi tabanlı sistemlerin birlikte işlerlik gereksinimi de vardır. Böyle sistemler, biçimsel bilgi gösterimindeki ifadeleri kullanarak işlem yapar ve iletişim kurar. Bunlar, sorgulamalar yapar ve cevap verir [34]. Ontolojiler, bilginin biçimsel olarak ifade edilmesini sağlar.

2.2.1. Ontoloji Dilleri

Son yıllarda anlamsal ağı (semantic web) gerçekleştirmek için birçok ontoloji dili geliştirilmiştir. İlk yaklaşımlar için; KIF (Knowledge Interchange Format), F-Logic, Dublin Core, CYC projesi örnek olarak verilebilir [35]. Güncel yaklaşımlar için ise; XML (eXtended Markup Language), RDF (Resource Description Framework), KA^2 (Knowledge Annotation Initiative of Knowledge Acquisition), SHOE (Simple HTML Ontology Extentions), OIL (Ontology Interchange Language), DAML (DARPA Agent Markup Language) ve OWL (Ontology Web Language) sayılabilir [35].

Bu dillerin öne çıkan bazıları ile ilgili detaylı açıklama aşağıda verilmiştir.

RDF (Resource Description Framework) ve RDFS (RDF Schema): RDF, World Wide Web Consortium (W3C) tarafından web meta-veri için geliştirilmiş bir standarttır [35]. Web'deki bilginin sunulması için bir çatı [36] ve yine bu bilginin sunulması için genel amaçlı bir dildir [37]. XML tabanlı sözdizimi kullanır. RDF, XML şema (schema) veri türleri ile gösterilen değerleri kullanabilir. Böylece diğer XML uygulamaları ile bilgi değişimine yardım eder. RDF temelde bir veri modelidir. Oldukça basit olan bu veri modeli web'deki kaynaklar (resource) hakkında ifadeler içerir. Kaynak, web'de belirteci (identifier) olan herhangi bir şeydir [38]. RDF'de bir kaynak üçlüler (triples) ile ifade edilir. Her üçlü bir özne, bir nitelik ve bir değerden oluşmaktadır. RDF, XML'in yerini alabilecek bir şey değil, XML'i kullanarak bunun

üstünde yer alan bir katmandır [35]. XML veri yapılandırmak için bir standart, RDF ise veri hakkında bir şeyler söylemek, bir diğer deyişle ona anlam vermek için bir mekanizmadır [38].

RDF şema (schema), belli bir alana özgü nitelikler ve sınıflar tanımlamak için bir mekanizma sağlar [38]. RDF şemanın içerdiği temel modelleme elemanları şunlardır [38]: (1) sınıf tanımları ve alt-sınıf ifadeleri (bu ikisi birlikte sınıf hiyerarşisi tanımlamaya izin verir); (2) nitelik tanımları ve nitelik ifadeleri (nitelik hiyerarşisi oluşturmak için); (3) alan ve aralık ifadeleri (sınıf ve niteliklerin muhtemel bir araya gelişlerinin sınırlamak için); ve (4) tür ifadeleri (bir kaynağı belli bir sınıfın bir olgusu olarak ifade etmek için). Bu elemanlar, belli bir alan ile ilgili şema oluşturmak için kullanılabilirler [38].

RDF ve RDFS birlikte, web kaynakları için bir bilgi gösterim mekanizması sağlar. Ancak, kapsamlı bir bilgi gösterim dili ile kıyaslandığında, RDFS oldukça basit kalmaktadır. Bundan dolayı veri anlamının daha net belirtilebilmesini sağlayan daha zengin dillere ihtiyaç duyulmuştur.

DAML (DARPA Agent Markup Language): Ağustos 2000'te başlayan bir programdır. Birleşik Devletler sponsorluğunda çalışmaya başlamış olan bu program ile anlamsal web'i desteklemek için bir dil ve araçlar oluşturmak hedeflenmiştir [39]. Akademisyenler, hükümet kurumları, yazılım geliştirme firmaları ve endüstriyel kuruluşlar bu programda yer almaktadır [40].

DAML; XML ve RDF'e uzantı olarak geliştirilmiştir [39]. İki parçadan oluşmaktadır. Bunlar, bir ontoloji dili ve kısıtların ifade edilebileceği ve çıkarsama kurallarının eklenebileceği bir dildir. DAML aynı zamanda SHOE, OIL, KIF, XML ve RDF gibi anlamsal web dilleri ile eşleştirmeler de içermektedir [35].

OIL (Ontology Interchange Language): DAML ile yaklaşık aynı zamanlarda başlayan bu çalışma, Avrupa Birliği Bilgi Toplumu Teknolojileri (IST - Information Society Technologies) programı tarafından finanse edilen On-To-Knowledge projesi (IST-1999-1013) ve IBROW (IST-1999-19005) çatısı altında yer almaktadır [41]. Çalışma akademik, ticari, endüstriyel ve bu konu ile ilgilenen kişiler tarafından gerçekleştirilmiştir.

OIL üç unsur üzerine kurulmuştur. Bunlar; çerçeve-tabanlı (frame-based) sistemler, tanımlama mantığı (DL-Description Logic) ve web standartlarıdır (XML, RDF) [42]. Çerçeve-tabanlı sistemlerin sağladığı zengin modelleme elemanları (primitive), anlamsal mantık tarafından sağlanan etkin çıkarsama ve biçimsel anlam bilimi (formal semantics) ve web topluluğu tarafından sağlanan sözdizimsel değişim gösterimleri için standart önerileri (XML, RDF) OIL'in temellerini oluşturur [42].

DAML+OIL: DAML ve OIL'in özellikleri birleştirilerek oluşturulmuş bir ontoloji dilidir. RDF ve RDFS üzerine inşa edilmiştir [43].

DAML ile ilgili çalışmalar Birleşik Devletler tarafından, OIL ise daha çok Avrupa Birliğindeki geliştiriciler tarafından geliştirildiğinden dolayı, DAML+OIL çalışmaları bu ikisinin bir araya geldiği bir komisyon tarafından yürütülmüştür [44]. Çalışma 2006 başlarında sonlandırılmıştır.

OWL (Web Ontology Language): Ontolojilerin web üzerinden yayınlanması ve paylaşılması için anlamsal işaretleme (markup) dilidir [45]. OWL, DAML+OIL'den türetilmiştir ve RDF üzerine kurulmuştur [45]. W3C tarafından oluşturulan Web Ontology Working Group tarafından oluşturulmuş olan OWL, farklı amaçları olan kullanıcılar ve geliştiriciler için tasarlanmış üç farklı alt dil sağlamaktadır. Bunlar;

- *OWL Lite:* Basit kısıtlamaları (constraint) olan, daha çok sınıflandırma hiyerarşisine ihtiyaç duyan kullanıcıları desteklemektedir. OWL Lite, OWL DL'den daha az biçimsel karmaşıklığa sahiptir [46].

- *OWL DL:* Hesaplanabilirliği ve karar verilebilirliği kaybetmeden en üst seviyede açıklayıcılığa ihtiyacı olan kullanıcılar içindir. Bu dilde tüm sonuçların hesaplanabilir olduğu ve hesaplamaların sonlu bir zamanda tamamlanacağı garanti edilir. OWL-DL, OWL dilinin tüm yapısını içermekle birlikte, sadece belli sınırlar dahilinde kullanılabilmektedir [46].

- *OWL Full:* RDF'in sözdizimsel özgürlüğüne ve en üst seviyede açıklayıcılığa ihtiyacı olan kullanıcılar içindir. Ancak, hesapmaların olurluluğu ile ilgili garanti yoktur [46].

Günümüzde OWL anlamsal web için standart bir ontoloji dilidir [35]. SHOE, DAML+OIL dahil olmak üzere, önceki diller ile uyumludur ve mühendislere anlamı daha güçlü ifade etme olanağı sağlar. OWL, çıkarsayıcılar tarafından kullanılarak mantıksal çıkarsama yapılabilir ve bilgi türetilebilir [35].

Tez kapsamında oluşturulan ontolojilerde dil olarak OWL seçilmiştir.

2.2.2. Ontoloji Araçları

Günümüzde ontolojileri; geliştirme, birleştirme, eşleştirme, değerlendirme, saklama ve sorgulama için geliştirilmiş birçok araç vardır [47].

Ontoloji geliştirme araçları: Bunlar yeni bir ontoloji oluşturmak veya mevcut bir ontolojiyi kullanmak için kullanılan araçlardır. Sadece metin düzenleme için değil, ontolojilerin belgelenmesi, dışa aktarımı (exportation) veya farklı biçimlerden veya kütüphanelerden içe aktarımı (importation) gibi işlemler için de kullanılırlar [47]. OntoEdit, Protégé, WebOde ve Ontolingua bunlara örnek olarak verilebilir [35] [47].

Ontoloji birleştirme (merge) ve bütünleştirme (integration) araçları: Bu araçlar aynı alandaki farklı ontolojilerin birleştirilmesi ve bütünleştirilmesindeki problemlerin çözümü için geliştirilmiş araçlardır. Aynı alanda mevcut iki ontolojiden daha kaliteli bir ontoloji oluşturulmak istendiğinde veya iki organizasyonun birleşmesi durumunda bunların kullandıkları ontolojilerin de birleştirilmesi istendiğinde, böyle bir ihtiyaç ortaya çıkabilmektedir. Ontoloji birleştirme araçlarına örnek olarak; PROMPT, ODEMerge ve Chimaera verilebilir [47].

Ontoloji değerlendirme (evaluation) araçları: Bu araçlar ontolojilerin ve ontoloji ile ilişkili teknolojilerin, verilen bir kalite seviyesinde olduğundan emin olmayı destekleyen araçlardır. Ontolojilerin bütünleştirilmesinde ve endüstriyel uygulamalardaki ontoloji-tabanlı teknolojilerde problemlerden kaçınmak için kalite güvencesi önemlidir [47].

Ontoloji tabanlı açıklama (annotation) araçları: Bu araçlar otomatik ya da yarı otomatik bir şekilde web sayfalarının içine ontoloji-tabanlı işaretlemeler (markup) eklenmesini veya bunların idame edilmesini sağlamak için tasarlanmışlardır [47].

Ontoloji depolama ve sorgulama araçları: Ontolojilerin kolay bir şekilde kullanılması ve sorgulanması için oluşturulmuş araçlardır. Geniş kabul görmesinden ve web'in bir bilgi paylaşım ortamı olarak kullanılmasından dolayı, bu bağlamda yeni ontoloji dilleri oluşturulmuştur [47].

Ontoloji öğrenme (learning) araçları: Doğal dil ile yazılmış metinlerden otomatik ya da yarı otomatik olarak ontoloji türetmek için kullanılırlar [47].

Ontoloji geliştirme/düzenleme ile ilgili olan araçlar incelendiğinde pek çok araç olduğu görülmektedir. M. Denny tarafından yapılan incelemede [48], 94 ontoloji geliştirme/düzenleme aracı 13 karakteristik özellik açısından ele alınarak incelenmiştir. Ontoloji geliştirmek için kullanılacak araç seçilirken göz önünde bulundurulması gereken ana kriterlerin [49] bazıları şunlardır; her bir araç tarafından ontoloji geliştirme süreçlerinin hangi aktivitelerinin desteklendiği, araç ile ilişkilendirilen bilgi modelinin ifade gücü, aracın yerel olarak kurulup kurulmadığı, hangi ontoloji dilleri ya da formatlarında ontoloji üretebileceği, ontolojinin hangi ortamda saklanacağı.

Bu tezde, ontoloji geliştirme aracı olarak, kullanıcı-dostu grafiksel kullanıcı arayüzü olan, desteklenen ve yaygın kullanımı olan Protégé kullanılmıştır.

2.2.3. Ontoloji İşlemleri

Uygulamalar tarafından kullanılan ontolojiler üzerinde bazı işlemler yapabilme ihtiyacı vardır. Bir uygulamanın birden fazla ontolojiyi kullanması veya farklı ontolojiler kullanan sistemlerin bütünleştirilmesi ihtiyacı gibi durumlarda, bunların hepsi ile birlikte çalışabilmek için bazı ontoloji işlemleri yapılması gerekmektedir [50]. Aşağıda, tez ile ilişkili görülen ontoloji eşleme ve ontoloji sorgulama işlemlerine değinilmiştir. Ontoloji birleştirme ile ilgili bilgiye, ontoloji eşleme işleminden farkının vurgulanması için yer verilmiştir.

2.2.3.1. Ontoloji Eşleme (Mapping)

Bir ontolojiden bir başka ontolojiye eşleme, ifadelerin bir ontolojiden diğerine nasıl bir yol ile çevrileceğinin ifade edilmesidir. Sıklıkla ontolojiler arasında kavramların ve ilişkilerin çevirisi anlamına gelmektedir. En basit anlamda ise, bir ontolojideki bir kavram ile bir başka ontolojideki bir kavram arasında kurulan eşlemedir. Ancak, her zaman böyle bire bir eşleme mümkün olamayabilmektedir. Eşleme sırasında

bazı bilgi kayıpları olabilmekle birlikte, eşleme herhangi bir tutarsızlığa yol açmaz [50].

2.2.3.2. Ontoloji Birleştirme (Merging)

Ontolojilerin birleştirmesi, mevcut ontolojilerin birleştirilerek yeni bir ontoloji oluşturulmasıdır. Geleneksel gereksinim, yeni ontolojinin orijinal ontolojilerdeki tüm bilgiyi içermesidir. Bununla birlikte, bu gereksinim her zaman tam olarak karşılanmak zorunda değildir. Çünkü, birleştirilecek olan ontolojiler birlikte tamamen tutarlı olamayabilmektedirler. Bu durumda, sonucun tutarlı olması için bilgiler orijinal ontolojilerden seçilerek yeni ontolojiye dahil edilir. Orijinal ontolojilerin terimleri arasında bir köprü görevi görmesi için, birleştirilen ontoloji yeni kavramlar ve ilişkiler içerebilir [50].

2.2.3.3. Ontoloji Sorgulama (Querying)

Ontolojiler sorgulanabilir yapılardır. Ontoloji dillerinin geliştirilmesi ile birlikte, sorgulama dilleri ve sistemleri de geliştirilmiştir. Ontoloji sunucuları, verinin arkasındaki anlamın kısmen farkındadırlar ve bu durum sunucuların sorgu ve çıkarsama gibi hizmetler/servisler sunmalarını sağlar [51]. Ontoloji dilleri ve sistemler; ifade gücü, etkinlik, ölçeklenebilirlik, performans vb. özellikleri açısından farklılıklar göstermektedir.

Ontoloji sorgulama sistemleri ve dillerine örnek olarak şunlar verilebilir; HP Laboratuvarları tarafından geliştirilen Jena2; RDF, RDFS ve OWL için çeşitli çıkarsayıcılar ve RDQL olarak isimlendirilen esnek sorgulama dili sağlamaktadır. Başka bir sistem olan OWL-QL, ontoloji sorgulama için değerli özellikler içermektedir. Bir diğer sistem RACER'dır. RACER, nRQL sorgu dilini kullanmaktadır. Aynı zamanda RDF, RDFS ve OWL ontoloji dillerinde çıkarsama yapmayı desteklemektedir. SPARQL ise W3C tarafından geliştirilmiş olan RDF sorgulama dilidir [52].

2.3. Kullanılan Araçlar ve Teknolojiler

2.3.1. Eclipse

Eclipse, açık geliştirme platformu oluşturmaya odaklanmış projeleri olan bir açık kaynak topluluğudur. Bu topluluğun projeleri, yazılımların yaşam döngüsü boyunca; inşası, yayınlanması ve yönetilmesi için genişleyebilir çatılar, araçlar ve

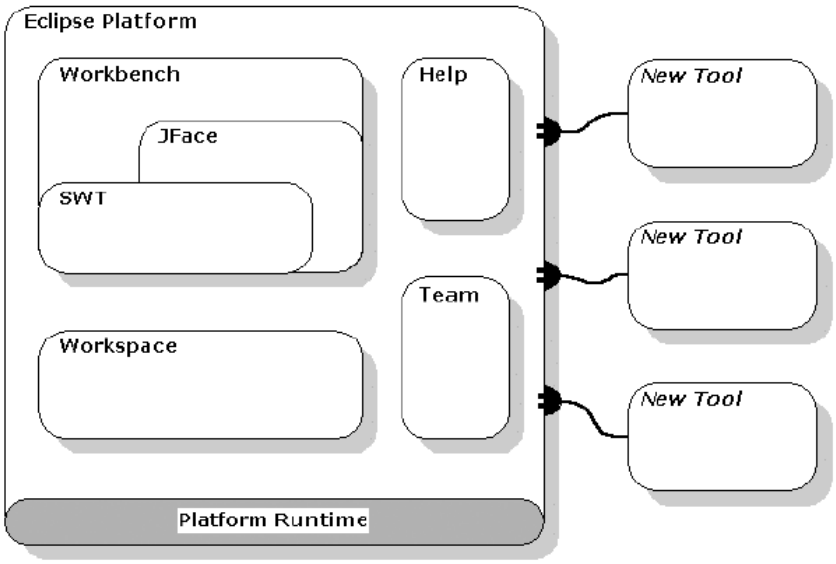
çalışma zamanlarından (runtime) oluşmaktadır. Bu projelere, kâr amacı gütmeyen ve üyeler tarafından desteklenen Eclipse Kuruluşu (Eclipse Foundation) ev sahipliği etmektedir. Eclipse Kuruluşu, Eclipse projelerinin yanında, açık kaynak topluluğunun ilerletilmesine ve birbirini tamamlayan ürünler ve hizmetler için bir ekosistem oluşturmaya yardım etmektedir [12].

Eclipse Platformu:

Eclipse platformu; Java tabanlı, genişleyebilir, açık kaynaklı bir geliştirme platformudur [53]. Bütünleşik Geliştirme Ortamı (BGO) (Integrated Development Environment – IDE) inşa etmek için gerekli olan işlevselliği içerir. Bununla birlikte, Eclipse platformunun kendisi de bileşenlerin birleşiminden oluşan bir BGO'dur [54]. Bu platform kullanılarak bir geliştirme ortamı inşa edilebileceği gibi, araçlar ve uygulamalar da geliştirmek mümkündür.

İlerleyen bölümlerde, anlaşılırlığı arttırmak için, Eclipse platformundan "Eclipse" olarak bahsedilecektir.

Eclipse'in, programlamayı destekleyecek araçların birlikte çalışmasını kontrol eden temel hizmetleri sağlayan genişleyebilir bir mimarisi vardır (Şekil 5). Araç geliştirenler, kendi eklenebilir bileşenlerini Eclipse kontratına uyacak şekilde paketleyerek bu platforma katkıda bulunabilirler. Eclipse'e eklenebilen bu bileşenlere Eclipse eklentisi (plug-in) denilmektedir [55]. Eclipse'in genişleyebilirliği, platformdaki bir grup çekirdek eklentinin, yeni eklentileri de sisteme dahil eden bir mekanizma sağlaması ile gerçekleşir.

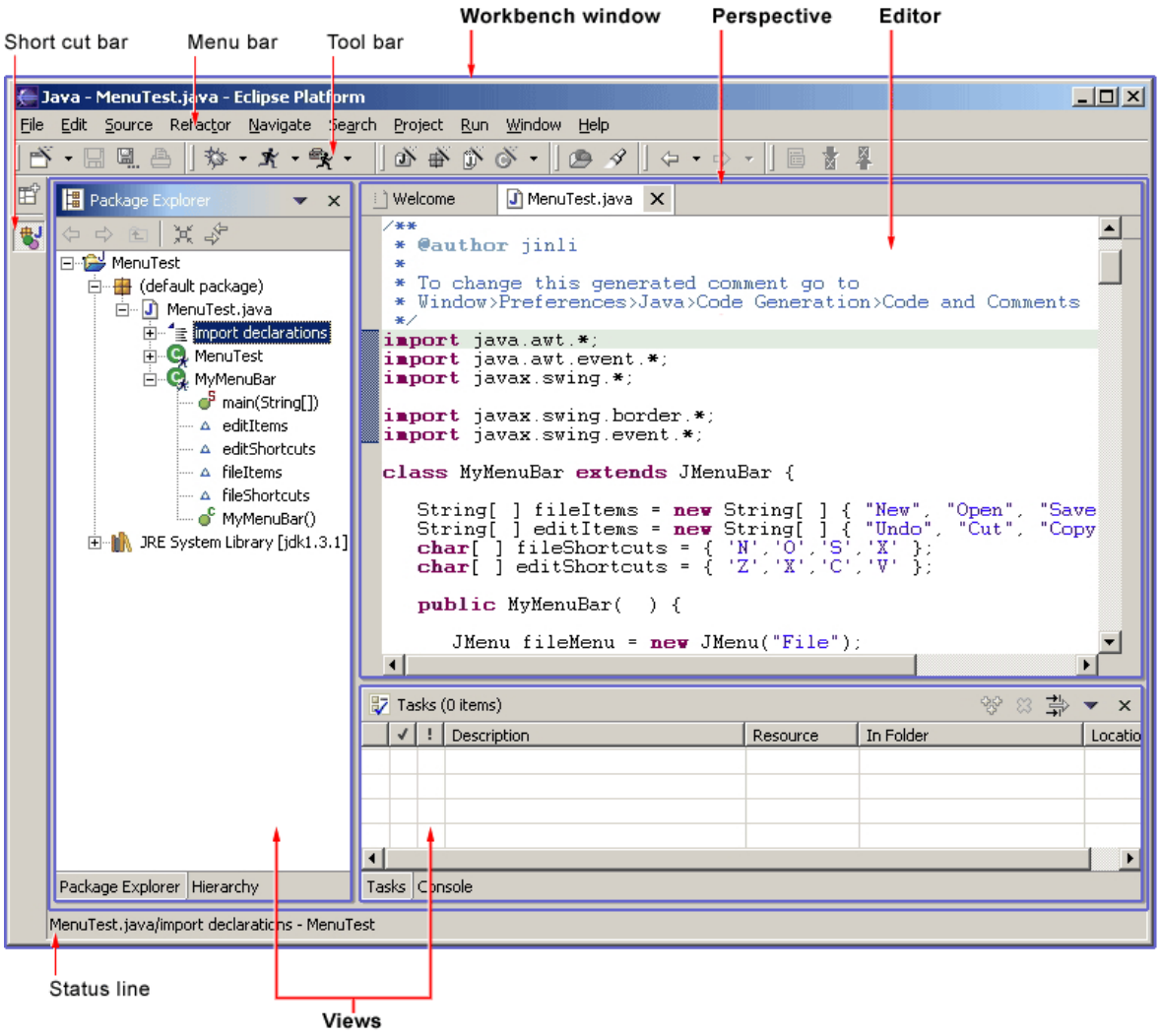


Şekil 5 - Eclipse Platform Mimarisi [54]

Eclipse Kullanıcı Arayüzü:

Eclipse'de masa üstü geliştirme ortamı, çalışma tezgahı (workbench) olarak isimlendirilir. Şekil 6'da ekran görüntüsü verilmiş olan çalışma tezgahı, araçların görünmez bir şekilde bütünleştirilmesini ve görünür bir şekilde kontrol edilebilmesini sağlar [56].

Bir çalışma tezgahı, pencereler (window) koleksiyonudur. Her pencere; bir menü çubuğu, bir araç çubuğu, bir kısayol çubuğu ve bir ya da daha fazla sayıda perspektiften oluşmaktadır [57]. Perspektifler; görünümler (views) ve metin düzenleyiciler (editors) içerirler ve belli bir menüde ve araç çubuğunda nelerin görüneceğini kontrol ederler [56]. Bir görünümün tipik kullanım sebepleri; hiyerarşik yapıdaki bilgi üzerinde dolaşma (navigation) veya bir metin düzenleyiciyi açmak ya da aktif metin düzenleyicideki bir nesnenin niteliklerinin gösterilmesini sağlamak olabilir. Bir metin düzenleyici ise, tipik olarak, bir dokümanın veya girdi nesnesinin düzenlenmesi için kullanılır. Metin düzenleyicide yapılan değişikliklerde aç-sakla-kapat yaşam döngüsü modeli izlenir. Görünümlerde ise, metin düzenleyicilerin aksine, yapılan değişiklikler anında saklanır [58].



Şekil 6 - Eclipse Çalışma Tezgahı (Eclipse Workbench) [57]

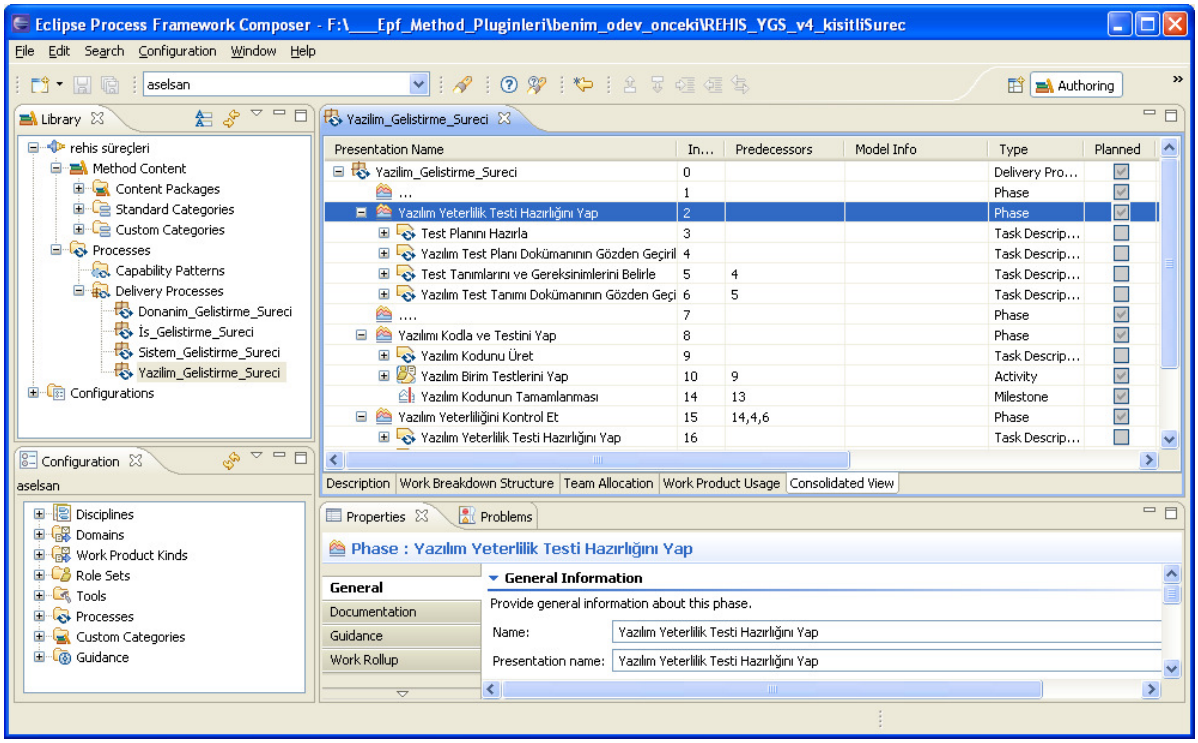
2.3.2. EPF (Eclipse Process Framework)

EPF, Eclipse Birliği [12] tarafından oluşturulmuş olan, eklentiler ile genişletilebilir bir yazılım süreç mühendisliği çatısıdır. Eclipse tabanlı ve açık kaynak kodlu olan bu çatı, EPF Composer olarak isimlendirilen ve SPEM'in bir gerçekleştirimi olan bir süreç yönetimi aracından ve bu araç kullanılarak oluşturulmuş örnek süreç eklentilerinden oluşmaktadır.

Bu rapor boyunca, okunurluğu kolaylaştırmak amacıyla, EPF Composer aracı kısaca "EPF" olarak ifade edilmiştir. Çatı bağlamında EPF'den bahsedilirken ise "EPF çatısı" olarak açıkça belirtilmiştir.

EPF, yazılım ve sistem geliştirme ile ilgili; yöntemlerin ve süreçlerin yazılabilmesini, kütüphanelerin yönetimini, yapılandırılmasını ve yayınlanmasını sağlayan bir süreç yönetimi aracıdır [59]. Bu araç, yazılım ve sistem geliştiren bir

kurumda veya bir geliştirme projesinde; süreçlerin yazılması, güncellenmesi, yayınlanması işlerinden sorumlu olan kişiler tarafından kullanılabilir. Yazılan süreçler web sayfaları olarak yayınlanabilmektedir. Şekil 7'de EPF genel ekran görünümü verilmiştir.



Şekil 7 - EPF Kullanıcı Arayüzü

EPF projesi kapsamında; OpenUP, Scrum ve XP için yöntem içerikleri ve örnek süreçler oluşturulmuş ve yayınlanmıştır. Bu süreçler, EPF ile birlikte kullanılabilmektedirler.

2.3.3. Protégé

Stanford Üniversitesi Tıp Okulu, Biyomedikal Enformatik Araştırma Merkezi tarafından geliştirilmiş, ücretsiz, açık kaynak kodlu ontoloji editörü ve bilgi-tabanı çatısıdır [60]. Java tabanlı, eklentiler ile genişletilebilen Protégé, hızlı prototipleme ve uygulama geliştirme için tak-ve-çalıştır (plug-and-play) ortamı sağlayan esnek bir zemindir. Protégé; geliştiriciler, akademisyenler, hükümet ve şirket kullanıcılarından oluşan güçlü bir topluluk tarafından desteklenmektedir.

Protégé platformu ontoloji modellemeyi, Protégé-OWL Düzenleyici ve Protégé-Çerçeveler Düzenleyici olmak üzere iki şekilde destekler. Protégé ontolojileri;

RDF(S), OWL ve XML şema dahil olmak üzere çeşitli biçimlerde (format) dışa aktarım (export) yapılabilir.

Protégé-OWL Düzenleyici: Protégé'nin OWL'yi destekleyen bir uzantısıdır. Bu metin düzenleyici; OWL ve RDF ontolojilerinin yüklenmesine ve saklanmasına; sınıfların, niteliklerin ve SWRL kurallarının görüntülenmesine ve düzenlenmesine; mantıksal sınıf karakteristiklerinin OWL deyimi olarak tanımlanmasına; çıkarsayıcının bir tanımlama mantığı sınıflayıcısı (description logic classifiers) gibi koşturulmasına ve OWL olgularının düzenlenmesine olanak verir. Protégé-OWL'nin esnek mimarisi, onun kolayca yapılandırılabilen ve genişletilebilen bir araç olmasını sağlar. Protégé-OWL Jena ile sıkı bir şekilde bütünleşiktir ve açık kaynak kodlu bir Java API'ye sahiptir.

Protégé-Çerçeveler Düzenleyici: Çerçeve-tabanlı alan ontolojileri oluşturma ve saklama, veri girişi formlarını isteğe göre değiştirme ve olgu verilerinin girişini yapmada kullacıları destekleyecek bir kullanıcı arayüzü ve bilgi sunucusu sağlar.

2.3.4. Jena

HP Laboratuvarları Anlamsal Web Araştırmaları (HP Labs Semantic Web Research) tarafından olgunlaştırılan Jena, mantıksal web inşa etmek için bir Java çatısıdır. Açık kaynak kodlu olan bu çatı, RDF, RDFS, OWL, SPARQL için programlanabilir bir ortam sunar. Ayrıca, kural tabanlı çıkarsama motoru içerir [61].

2.4. CMMI: Tümleşik Yetenek Olgunluk Modeli

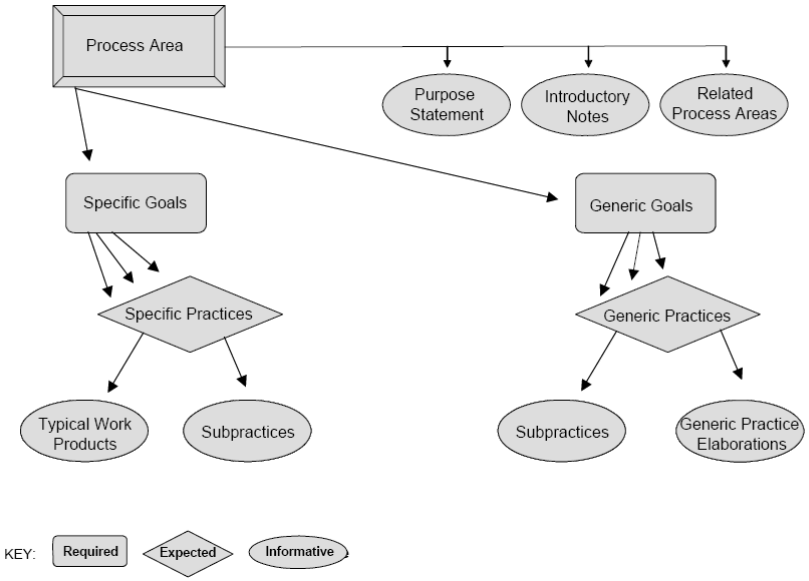
CMMI, Carnegie Mellon Üniversitesi, Yazılım Mühendisliği Enstitüsü (SEI) tarafından oluşturulmuş bir süreç referans modelidir. Ürünlerin ve hizmetlerin geliştirilmesinde ve bakımında uygulanan faaliyetleri adresleyen bu model, süreçlerin iyileştirilmesinde ve süreçlerin yeteneğinin veya olgunluğunun değerlendirilmesinde kullanılabilir. CMMI çatısı altında, belli bazı ilgili alanlarının ihtiyaçlarını karşılamak üzere *takımyıldız* (constellation) olarak isimlendirilen farklı modeller oluşturulmuştur. Şu an 3 tane CMMI takımyıldızı vardır. Bunlar; geliştirme için 'CMMI for Development' [8], hizmetler için 'CMMI for Services' [16] ve satın alma için oluşturulan 'CMMI for Acquisition' [17] takımyıldızlarıdır.

CMMI modeli temel olarak; süreç alanlarından ve bu süreç alanlarının bileşenlerinden oluşur. Bununla birlikte, modelde, süreç alanı bileşenlerinin nasıl ele alınması ve değerlendirilmesi gerektiğini anlatan iki farklı gösterim ve iki farklı kapsam yer almaktadır.

Modelde yer alan gösterimler, *basamaklı* (staged) ve *sürekli* (continuous) gösterim; kapsamlar ise, *IPPD dahil* (with IPPD) ve *IPPD hariç* (without IPPD) kapsamlarıdır. Gösterimler, modeldeki süreç alanı bileşenlerinin nasıl ele alınması ve değerlendirilmesi gerektiğini tarif eden, aynı modelin iki farklı bakış ile ele alınmasını sağlayan yaklaşımlardır. IPPD ise, bütünleşik takım faaliyetlerini kapsayacak şekilde, hedefler ve pratikler ile CMMI'daki süreç alanlarının genişletilebilmesini sağlayan bir ilâvedir.

2.4.1. Süreç Alanı (Process Area)

Belli bir alanda, birlikte uygulandığında bu alan ile ilgili gelişme sağlayacak hedefler ve bu hedeflere ulaştıran en iyi uygulamalardan oluşan ilişkili pratikler kümesidir. CMMI'daki süreç alanları, Şekil 8'de verilen bileşenlerden oluşmaktadır. Bu bileşen yapısı modeldeki tüm süreç alanları için geçerlidir.



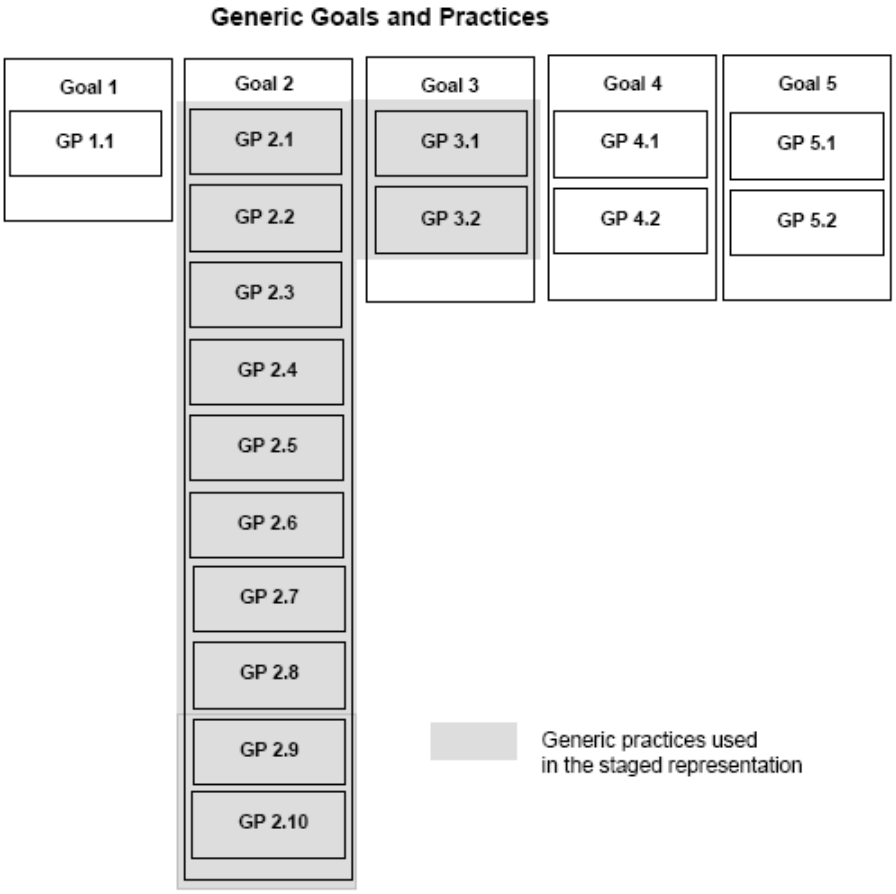
Şekil 8 - CMMI Model Bileşenleri [8]

Şekil 8'de de görüldüğü gibi, süreç alanının bileşenleri; *gerekli*, *beklenen* ve *bilgilendirici* olmak üzere 3 tiptedir. Süreç değerlendirmelerinde, gerekli ve beklenen tipteki bileşenler göz önünde bulundurulmaktadır. Hedef bileşenleri gerekli, pratik bileşenleri ise beklenen tipteki bileşenlerdir.

Hedefler, bir hedef cümlesinden ve bu hedefe ait pratiklerden oluşmaktadır. Hedef cümlesi, varılması gereken hedefi ortaya koyarken; pratikler, o hedefe ulaşılmasını sağlayan ve en iyi uygulamalardan edinilen tecrübelere dayanılarak oluşturulmuş uygulama adımlarını göstermektedir. CMMI ile uyumluluğun ele alındığı süreç değerlendirmelerinde, gerekli bileşen olan hedeflere ulaşılması bir zorunluluk iken, beklenen bileşen olan pratiklerin aynen modeldeki gibi uygulanması bir zorunluluk değildir. Bir başka deyişle, pratikler modeldeki pratiğin yerini alabildiği ve hedefe ulaştırdığı müddetçe, kurum tarafından modeldekinden farklı olarak uygulanıyor da olabilmektedir.

CMMI'da iki tip hedef vardır; *'özel hedef'* (Specific Goal – SG) ve *'genel hedef'* (Generic Goal – GG). Özel hedefler, ait oldukları süreç alanlarına özgüdür ve o sürecin organizasyonda uygulanmasını sağlarlar. Genel hedefler ise, ait oldukları süreç alanının bir organizasyonda kurumsallaşmasına hizmet ederler. Özel hedeflerin sayısı ve içerdikleri pratikler her süreç alanında birbirinden farklı olmasına karşın, genel hedefler kurumsallaşmaya hizmet ettiğinden ve kurumsallaşma tüm süreç alanları için benzer olduğundan, genel hedeflerin sayısı ve içerdikleri pratikler her süreç alanı için benzerdir. Her süreç alanında 5 tane genel hedef vardır; GG1, GG2, GG3, GG4 ve GG5. Modeldeki süreç alanları hem basamaklı hem de sürekli gösterim için ortak olmakla birlikte, süreç alanlarının genel hedeflerinin ve genel pratiklerinin kullanılıp kullanılmayacağı ele alınan gösterime göre farklılaşmaktadır. Sürekli gösterimde genel hedeflerin ve pratiklerin hepsi kullanılırken; basamaklı gösterimde sadece GG2 ve GG3 kullanılır; GG4, GG5 ve bunların pratikleri kullanılmaz. Basamaklı gösterimde genel pratiklerin kullanımı Şekil 9'da görülmektedir.

Genel hedeflere içerik olarak bakıldığında ise; GG1'in diğer genel hedeflerden farklı olduğu görülmektedir. GG2, GG3, GG4 ve GG5 kurumsal olarak süreçlerin uygulanması, diğer bir deyişle kurumsallaşması ile ilgili hedefler olmasına karşın, GG1 sadece sürecin uygulanmasını gerektirmektedir. Bundan dolayı GG1'de, özel hedeflere (specific goal – SG) ulaşılması gerektiği belirtilmektedir. Özel hedeflere ise, özel pratiklerin (specific practice – SP) kurumda başarı ile uygulanması ile ulaşılmaktadır.



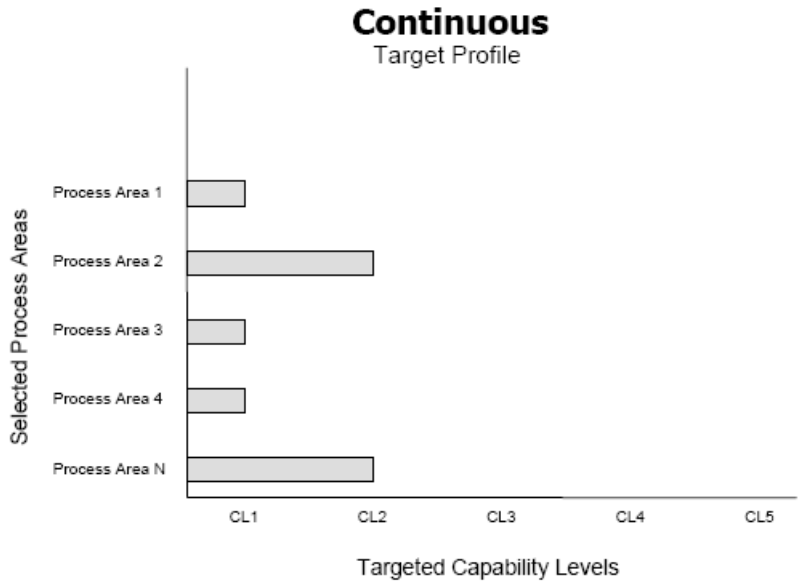
Şekil 9 - Basamaklı Gösterimde Genel Pratiklerin Kullanımı [8]

CMMI-Dev v1.2'de, tüm yazılım ve sistem geliştirme süreçleri, 'Konfigürasyon Yönetimi' (Configuration Management), 'Ölçme ve Analiz' (Measurement & Analysis), 'Proje Planlama' (Project Planning) vb. toplam 22 süreç alanında ele alınmıştır. Uygulamada, CMMI'daki bir süreç alanı, bir kurumdaki bir veya daha fazla süreçte ele alınabilirken, bunun tersi de olabilmekte, CMMI'daki birden fazla süreç alanı kurumdaki bir süreç ile karşılanıyor da olabilmektedir. Örneğin, bir kurumdaki Gereksinim Mühendisliği sürecinde, CMMI'daki Gereksinim Yönetimi ve Gereksinim Geliştirme süreç alanlarının pratikleri ele alınıyor olabilir.

Süreç alanları ile ilgili bir diğer konu da kategorilerdir. Her CMMI takımyıldızında, süreç alanları, o takımyıldıza özgü olacak şekilde bazı kategorilere ayrılmıştır. Her süreç alanı, bu kategorilerden bir ve yalnız birinde yer almaktadır. CMMI-Dev v1.2 takımyıldızında süreç alanları; Mühendislik, Süreç, Proje ve Destek olmak üzere 4 kategoride ele alınmaktadır.

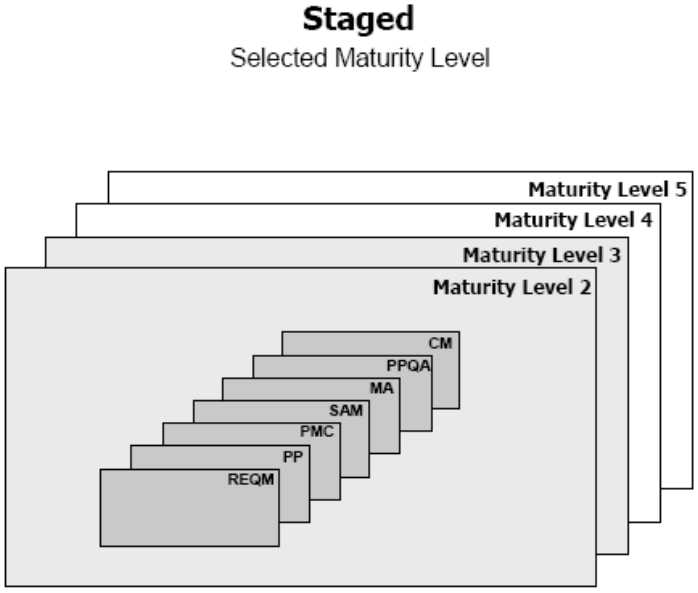
2.4.2. Gösterimler (Representations)

CMMI, modeldeki süreç alanları bileşenlerinin 2 farklı yaklaşım ile ele alınmasına olanak sağlayan 2 farklı gösterim içerir. Bunlar; *Basamaklı* (Staged) ve *Sürekli* (Continuous) gösterimlerdir. Sürekli gösterimde, süreç alanları birbirinden bağımsız bir şekilde ayrı ayrı ele alınıp referans olarak kullanılabilmekte ve derecelendirilebilmektedir. Şekil 10'da, sürekli gösterimde süreç alanlarının seçilebildiği ve ayrı ayrı seviyelendirilebildikleri görülmektedir.



Şekil 10 - Sürekli Gösterimde Süreç Alanlarının Derecelendirilmesi [8]

Basamaklı gösterimde ise süreç alanları *2.Seviye*, *3.Seviye*, *4.Seviye* ve *5.Seviye* olmak üzere dört gruba ayrılmıştır ve her seviyede ele alınması gereken süreç alanları modelde açıkça belirtilmiştir. Şekil 11'de, basamaklı gösterimin seviyelendirilmesi ve her seviyede bir grup süreç alanının yer aldığı ifade edilmektedir.



Şekil 11 - Basamaklı Gösterimdeki Olgunluk Seviyeleri [8]

Kullanılacak gösterim ile ilgili seçim yapılırken göz önünde bulundurulan hususlardan biri de süreç değerlendirmelerinin nasıl yapılmasının istendiğidir. Eğer organizasyonlar (müdürlük, direktörlük vb.) süreçlerinin kurumsal olarak değerlendirilmesini ve belgelendirilmesini istiyorlarsa, basamaklı gösterimi seçmelidirler. Bu durumda, süreç alanları modelde belirtilen gruplar halinde ele alınır. Ancak, organizasyonlar kendi seçtikleri süreç alanlarının, ayrı ayrı ya da kendi oluşturdukları süreç alanı grupları halinde değerlendirilmesini ve belgelendirilmesini istiyorlarsa, sürekli gösterimi seçmelidirler. Basamaklı gösterime göre değerlendirilen süreçler kurumsal olarak 'olgunluk seviyeleri' (*maturity levels*) ile, sürekli gösterime göre değerlendirilen süreçler ise ayrı ayrı 'yetenek seviyeleri' (*capability levels*) ile derecelendirilirler.

Olgunluk ve yetenek seviyelendirmeleri birbirinden tamamen kopuk değildir. Modeldeki bileşenlerin büyük bir kısmı her iki gösterimde de kullanıldığından, bir gösterimde yapılan derecelendirmenin diğer gösterimde hangi seviyeye karşılık geldiğini görmek mümkündür. Şekil 12'de, soldan sağa doğru; süreç alanı adı, bu adın kısaltması, ML kısaltması ile gösterilen olgunluk seviyesi (maturity level), CL kısaltması ile 5 seviyesi de görülen yetenek seviyesi (capability level) sütûnları görülmektedir. Her satırın ML sütûnunda yer alan olgunluk seviyesi için ele alınması gereken süreç alanları listenin sol tarafında, bu süreç alanları için hedeflenmesi gereken yetenek seviyeleri ise 'target profile' olarak CL sütûnlarında görülmektedir.

Name	Abbr	ML	CL1	CL2	CL3	CL4	CL5
Requirements Management	REQM	2					
Project Planning	PP	2	Target Profile 2				
Project Monitoring and Control	PMC	2					
Supplier Agreement Management	SAM	2					
Measurement and Analysis	MA	2					
Process and Product Quality Assurance	PPQA	2					
Configuration Management	CM	2					
Requirements Development	RD	3					
Technical Solution	TS	3					
Product Integration	PI	3					
Verification	VER	3					
Validation	VAL	3			Target Profile 3		
Organizational Process Focus	OPF	3					
Organizational Process Definition +IPPD	OPD +IPPD	3					
Organizational Training	OT	3					
Integrated Project Management +IPPD	IPM +IPPD	3					
Risk Management	RSKM	3					
Decision Analysis and Resolution	DAR	3					
Organizational Process Performance	OPP	4			Target Profile 4		
Quantitative Project Management	QPM	4					
Organizational Innovation and Deployment	OID	5			Target Profile 5		
Causal Analysis and Resolution	CAR	5					

Şekil 12 - Hedeflenen Profiller ve Denk Seviyelendirme [8]

2.4.3. Seviyelendirme: Yetenek ve Olgunluk Seviyeleri

Yetenek seviyeleri 0 ile 5 arasında (0 ve 5 dahil), olgunluk seviyeleri ise 1 ile 5 arasında (1 ve 5 dahil) değerler alabilmektedir. Her iki seviyelendirmede de 5 en yüksek seviyedir. İster olgunluk ister yetenek derecelendirmesi olsun, bir seviyede başarılı sayılabilmek için, bu seviyede karşılanması gerekenlerin yanısıra, önceki seviyelerin gerekleri de karşılanmalıdır.

Yetenek seviyelendirmesinde; her seviye bir genel hedef (generic goal – GG) ile ilişkilendirilmiştir. Bir süreç alanının 1., 2., 3., 4. ve 5. yetenek seviyesinde olması için sırasıyla bu süreç alanının 1., 2., 3., 4. ve 5. genel hedefine ulaşılıyor olması gerekmektedir. CMMI'daki tüm süreç alanlarının 1. genel hedefinde, bu süreç alanının özel hedeflerine (specific goal – SG) başarıyla ulaşılması gerektiği belirtilmektedir. Bir süreç alanının uygulanmasını hedefleyen özel hedeflerine ulaşılması için ise özel pratiklerin (specific practice – SP) tam olarak uygulanıyor olması beklenir. Eğer bir kurumda bir süreç alanındaki tüm özel pratiklerin uygulandığı herhangi bir uygulama yoksa, bu süreç alanının yetenek seviyesi sıfırdır. Özel hedeflere ulaşılmışsa, bu süreç alanı 'uygulanan' bir süreçtir ve 1. yetenek seviyesindedir. Bir süreç alanının 2. yetenek seviyesinde olabilmesi için; 'uygulanan' bir süreç olması ve 2. seviye genel pratiklerinin uygulanması yoluyla 2. genel hedefine ulaşılıyor olması gerekmektedir. 2. genel hedefine ulaşılan, diğer bir deyişle 2. yetenek seviyesinde olan bir süreç, 'yönetilen' bir süreçtir. Bir süreç alanının 3. genel hedefine ulaşıldığında ise, bu süreç 3. yetenek seviyesinde olan, diğer bir deyişle 'tanımlı' bir süreçtir. 4. genel hedef ile 4. yetenek seviyesine ulaşılır ve 'nicel ölçülen süreç' olarak isimlendirilir. Son olarak, 5. genel hedef ile 5. yetenek seviyesine ulaşılır ve 'en iyileşen süreç' olarak isimlendirilir. Her seviye için, bir önceki seviyeye ilâve olarak ilgili seviyedeki hedefe ulaşılıyor olması, bunun için de bu hedefe ait genel pratiklerin uygulanıyor olması gerekmektedir.

Olgunluk seviyelendirmesinde ise; süreçlerin tüm bir organizasyon olarak uygulanması değerlendirilir. Bundan dolayı bir sürecin uygulanıyor olması yetmez, bu sürecin organizasyonel olarak uygulanması önemlidir. Bundan dolayı sertifikalandırılan ilk seviye, kurumsallaşmanın başladığı 2. seviyedir ve '2. olgunluk seviyesi' olarak isimlendirilir.

Bir organizasyonun süreçlerinin 2. olgunluk seviyesinde olması için; 2. seviyedeki yedi süreç alanının 'yönetilen süreç' seviyesinde olması, diğer bir deyişle bu süreç alanlarının hepsindeki 2. genel hedeflere ulaşılmış olması gerekmektedir. Bunun için ise; bu yedi süreç alanındaki tüm özel pratiklerin ve 2. seviye genel pratiklerinin uygulanıyor olması beklenmektedir.

Bir organizasyonun süreçlerinin 3. olgunluk seviyesinde olması için; 2. ve 3. seviyedeki toplam on sekiz süreç alanının hepsinin 'tanımlı süreç' seviyesinde olması, diğer bir deyişle bu süreç alanlarının hepsinin 3. genel hedeflerine ulaşılmış olması gerekmektedir. Bu hedefe ulaşılabilmesi için ise; süreç alanının tüm özel pratiklerinin, 2. ve 3. seviye genel pratiklerinin uygulanıyor olması beklenmektedir.

Süreçlerin 4. olgunluk seviyesinde olması için; 3. olgunluk seviyesine ilâve olarak 4. olgunluk seviyesindeki iki süreç alanının 'tanımlı süreç' seviyesinde olması ve bu iki süreç alanının adreslediği şekilde, 2. ve 3. seviye süreç alanlarından seçilen bazı süreçler veya alt süreçler için 4. seviyedeki iki tanımlı sürecin uygulanması beklenmektedir.

Süreçlerin 5. olgunluk seviyesinde olması için; 4. olgunluk seviyesine ilâve olarak, 5. olgunluk seviyesindeki iki süreç alanının 'tanımlı süreç' seviyesinde olması ve bu iki süreç alanının adreslediği şekilde, 2. ve 3. seviye süreç alanlarından seçilen bazı süreçler veya alt süreçler için 5. seviyedeki iki tanımlı sürecin uygulanması gerekmektedir.

Gösterimlere göre seviyelerin isimlendirilmesi Çizelge 3'te verilmiştir.

Çizelge 3 - CMMI'da Gösterimlere Göre Seviyelerin İsimlendirilmesi

Seviye	*Sürekli Gösterim* *Yetenek Seviyeleri*	*Basamaklı Gösterim* *Olgunluk Seviyeleri*
Seviye – 0	Eksik	–
Seviye – 1	Uygulanan	Başlangıç
Seviye – 2	Yönetilen	Yönetilen
Seviye – 3	Tanımlı	Tanımlı
Seviye – 4	Nicel Yönetilen	Nicel Yönetilen
Seviye – 5	En İyileşen Süreç	En İyileşen Süreç

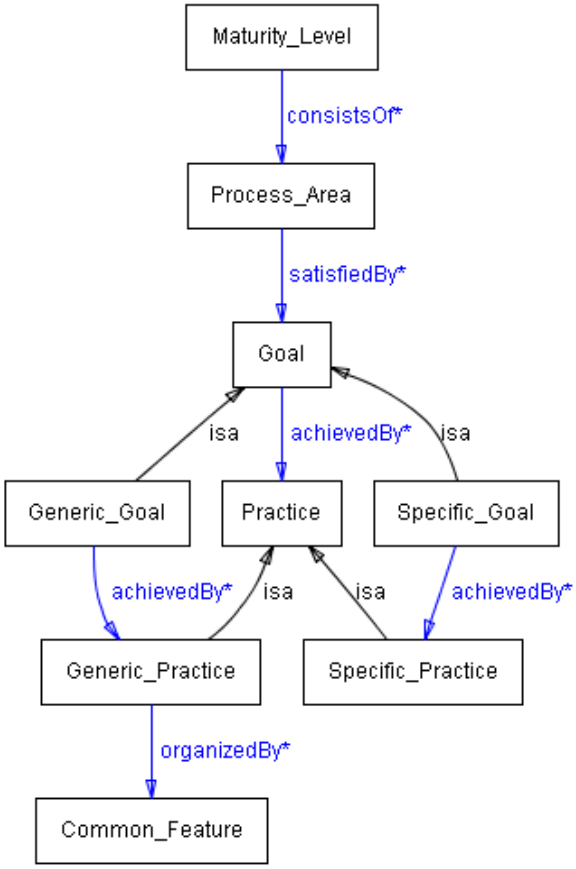
3. İLİŞKİLİ ÇALIŞMALAR

Tez kapsamında, bir CMMI ontolojisi ve kurum süreç yapısı ontolojisi oluşturulduğu için bu bölümde CMMI ontolojisi çalışmalarına ve süreç modelleme ontolojisi içeren çalışmalara odaklanılmıştır. Ayrıca, bölümün sonunda, çalışmanın hedefi ile ilişkili olduğu için, süreç değerlendirmeyi destekleyen araçlara da değinilmiştir.

3.1. CMMI Ontolojisi Çalışmaları

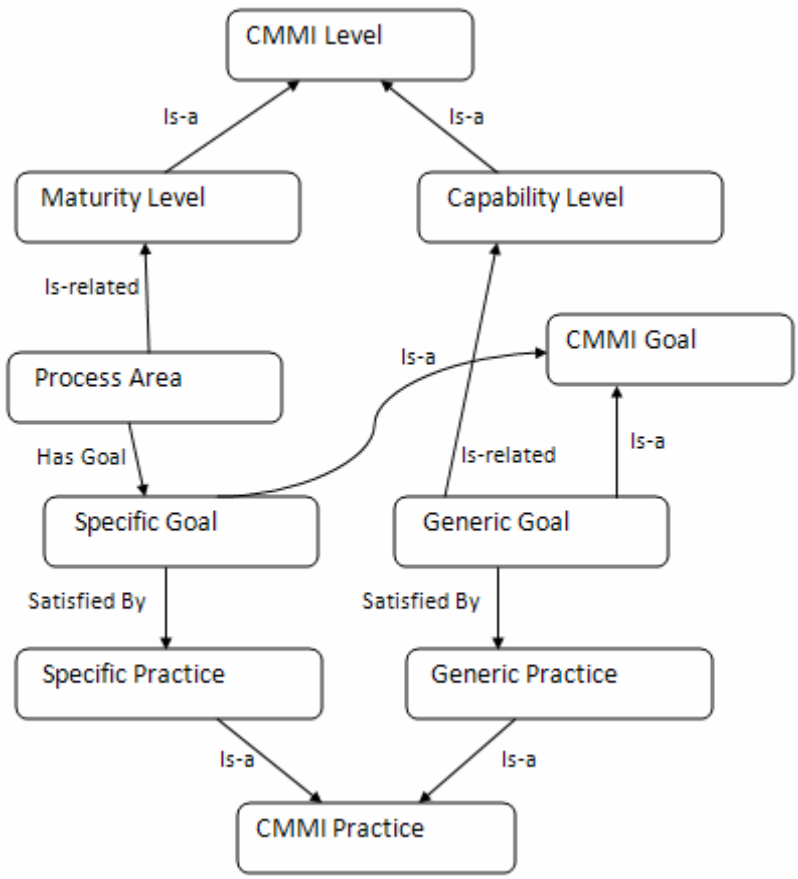
Literatürde, doğrudan CMMI'ı ele alan ve bu modelin ontolojisini oluşturmayı hedefleyen iki çalışma vardır. Bunlardan ilki, *Soydan et al.* [62] tarafından CMMI-SW v1.1 için oluşturulan ontolojinin yer aldığı çalışmadır. Diğeri ise, *Sharifloo et al.* [63] tarafından CMMI-ACQ takımyıldızı için oluşturulan ontolojinin anlatıldığı çalışmadır.

Soydan et al. çalışmasında, CMMI-SW v1.1'deki iki gösterimden biri olan basamaklı gösterim için bir OWL ontolojisi oluşturulmuş ve sürekli gösterim çalışma kapsamı dışında tutulmuştur. Bu çalışmada yer alan ontoloji ile ilgili bir görünüm Şekil 13'te verilmiştir.



Şekil 13 - CMMI-SW Ontolojisindeki Bazı Kavramlar ve Aralarındaki İlişkiler [62]

Sharifloo et al. tarafından oluşturulan CMMI-ACQ alan bilgisini gösteren bir ontoloji, SUMO [64] üst ontolojisi temel alınarak geliştirilmiş ve SOU-KIF [65] dilleri ile ifade edilmiştir. Şekil 14 bu ontolojideki ana kavramları ve bunlar arasındaki ilişkileri göstermektedir.



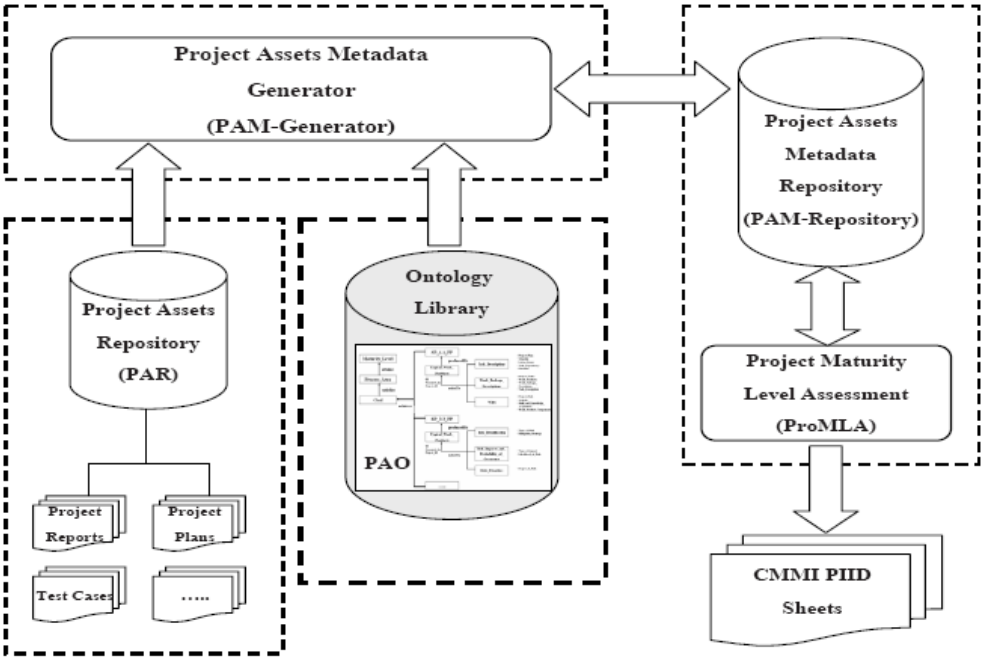
Şekil 14 - CMMI-ACQ Ontolojisindeki Ana Kavramlar ve İlişkiler [63]

Bu tezde oluşturulan CMMI ontolojisi, CMMI'daki iki gösterimi de içerdiği ve versiyon olarak en güncel olan versiyon 1.2 (CMMI-Dev v1.2) ele alındığı için Soydan et al.'un çalışmasından ayrılmaktadır. Sharifloo et al.'un çalışmasından ise, CMMI-Dev takımyıldızına odaklanıldığı için farklıdır.

3.2. Süreç Modelleme ve Değerlendirme İle İlgili Ontoloji Tabanlı Çalışmalar

Literatürde, süreç modelleme ve ontoloji konuları ayrı ayrı incelendiğinde, pek çok çalışma vardır. Ancak, süreç modelleme ontolojisi olarak bu iki konunun birleştirildiği çalışmalara odaklanıldığında, diğer bir deyişle bu iki konunun kesişim kümesine bakıldığında, mevcut çalışma sayısı oldukça azalmaktadır. Bu bölümde, bu kesişim kümesinde yer alan çalışmalar içinden, teze yakın olanlarına yer verilmiştir.

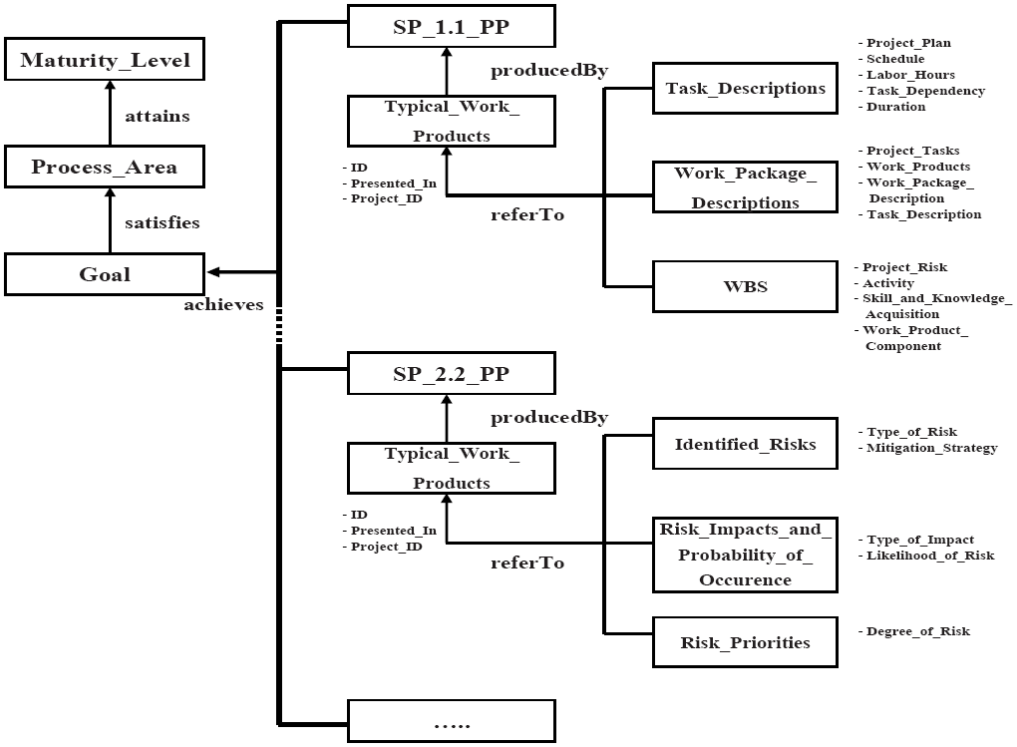
Rungratri ve Usanavasin [66], otomatik olarak CMMI fark analizi (gap analysis) yapmayı hedefledikleri ve CMMI-GAAF (CMMI v.1.2 based Gap Analysis Assistant Framework) olarak isimlendirdikleri bir çatı oluşturma ile ilgili bir çalışma yürütmektedir. CMMI-GAAF ile ilgili bir şekil Şekil 15'te verilmiştir.



Şekil 15 - CMMI-GAAF [66]

CMMI-GAAF kapsamında PAO (Project Assets Ontology) oluşturulmuştur. PAO; CMMI süreç alanları ile proje varlıklarının birleştirilmesi için bir ontolojidir. Çatıda yer alan diğer birimler: PAR (Project Assets Repository); proje varlıklarının bulunduğu bir depo, PAM-Generator (Project Assets Metadata Generator); proje varlıkları deposundaki (PAR) bilgiler ile PAO'yu birleştiren bir üretici, PAM Repository; proje varlıkları meta-veri deposu ve ProMLA (Project Maturity Level Assessment); proje olgunluk seviyesi değerlendirmesi yapan bir birimdir.

PAO, Soydan et al. [62] tarafından geliştirilen CMMI ontolojisi temel alınarak oluşturulmuştur. Oluşturulan bu ontolojinin bir bölümü Şekil 16'da görülmektedir. PAO ile, proje varlıkları ile CMMI'ın eşleştirilmesi için bir altyapı oluşturulmaya çalışılmıştır. Bir üretici (generator), proje varlıkları ve PAO'yu kullanarak, seviye değerlendirmesine girdi oluşturacak bir proje varlık meta-verisi oluşturmaktadır.



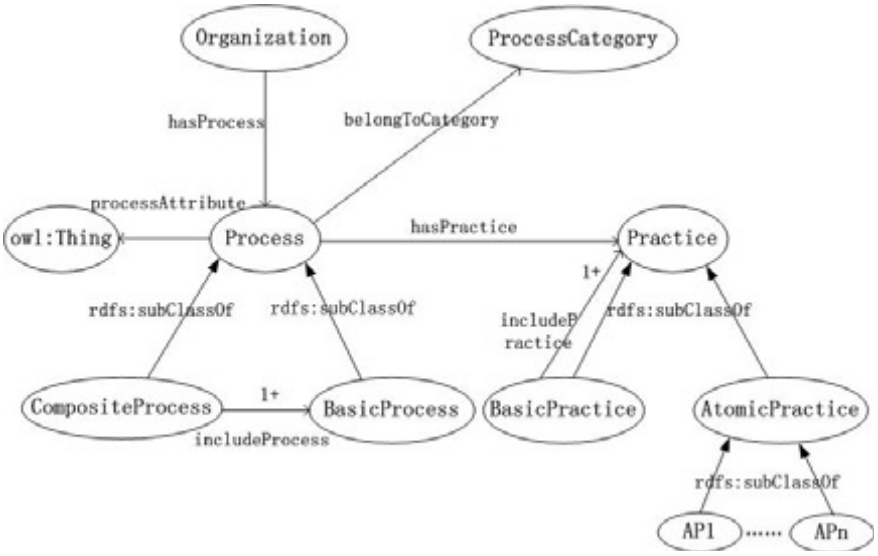
Şekil 16 - Project Assets Ontology (PAO) [66]

PAO ile özel ve genel hedeflerin, özel ve genel pratiklerin ele alındığı belirtilmiştir. Ancak, genel hedeflerin ve genel pratiklerin nasıl ele alındığı net bir şekilde verilmemiştir. Özellikle, CMMI'da genel pratikler bağlamında Tipik İş Ürünleri tanımlanmadığından, genel hedeflerin ve genel pratiklerin bu yapı ile nasıl ele alındığı konusu açıklanmamıştır. Tipik İş Ürünleri ile ilgili bir diğer nokta, bu model bileşeninin *bilgilendirici* bir bileşen olması ve pratiklerin tipik iş ürünleri ile karşılanıyor olmasının bir zorunluluk olmamasıdır. Bu açıdan, bir kurumda tipik iş ürünü ile karşılanmayan pratikler olduğunda, bunun nasıl ele alınacağı konusu net değildir. Bu tezde ise; CMMI varlık hiyerarşisindeki genel ve özel pratiklerin altında yer alan varlıklar ele alınmamış, genel ve özel pratikler ile süreç adımlarının ve varlıklarının eşleştirilmesi uygun bulunmuştur.

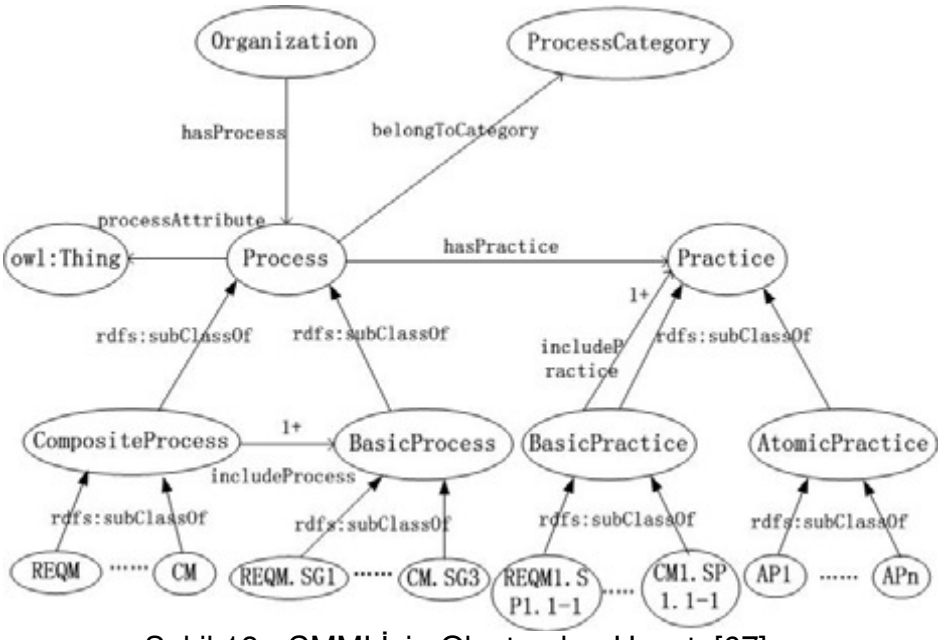
Ayrıca Rungratri ve Usanavasin [66] çalışmasında, CMMI ile proje varlıklarının eşleştirilmesine; bu tez kapsamında ise, CMMI ile kurumun süreç tanımlarının ve varlıklarının eşleştirilmesine odaklanılmış olması, bu iki çalışmayı birbirinden ayırmaktadır.

Liao et al. [67] ontoloji tabanlı yazılım süreç modelleme çatısı oluşturmuştur. Bu çalışmada süreç atomik pratikler ile ifade edilmiştir. Çalışma kapsamında OWL tabanlı bir yazılım süreç ontolojisi (software process ontology – SPO) geliştirilmiş,

bu ontoloji temel alınarak CMMI-SW v1.1 sürekli gösterim ve ISO/IEC 15504 için özel uzantılar oluşturulmuştur. Ayrıca, bir kurumun kendi süreç modelinin de SPO kullanılarak inşa edilebileceği belirtilmiştir. SPO'ya ve CMMI için oluşturulan uzantıya ilişkin RDF çizgeleri sırasıyla Şekil 17 ve Şekil 18'de verilmiştir.



Şekil 17 - SPO'nun RDF Çizgesi [67]



Şekil 18 - CMMI İçin Oluşturulan Uzantı [67]

Tez kapsamında, SPO temel alınarak web tabanlı bir süreç değerlendirme aracının geliştirildiğinden bahsedilmektedir, ancak çalışmanın sonuçları henüz yayınlanmamıştır (02.08.2009).

Liao et al. belli bir modele ya da standarda ait alan bilgisine bağlı kalmadan genel bir yazılım süreç ontolojisi oluşturmayı hedeflemiştir. Bu tez kapsamında ise CMMI

alan bilgisini yansıtan bir ontoloji oluşturulmuştur. Ayrıca, Liao et al. kurum süreçlerinin de oluşturdukları genel süreç ontolojisi ile ifade edilebileceğini belirtmiştir. Tezde ise, kurum süreçlerinin SPEM esas alınarak ifade edilmesi uygun görülmüş ve bu bağlamda kurum süreçlerini yazmak için SPEM'in bir gerçekleştirimi olan EPF aracı kullanılmıştır.

J. G. Doheny ve I. M. Filby [68] bir kavramsal süreç modelleme çerçevesi ve yazılım geliştirme süreçlerinin modellenmesini ve değerlendirilmesini destekleyici bir araç tanımlamışlardır. Bu çerçeve, süreç ontolojisi üzerine oturtulmuştur. Ontolojinin, yazılım geliştirme süreçlerinin modellenmesi için kullanılmasıyla birlikte, yazılım geliştirme süreci ile ilgili standartların ve yazılım mühendisliğindeki en iyi pratikler ile oluşturulan diğer gösterimlerin modellenmesi için de kullanılabileceği belirtilmiştir. Bu ontolojide kavramlar 3 ana kategoride ele alınmıştır. Bunlar; çıktılar (*artifacts*), aktiviteler (*activities*) ve ajanlardır (*agents*).

Doheny ve Filby çalışmasında [68], Liao et al. tarafından da hedeflendiği gibi, süreç modelleme, süreç referans modelleri ve süreç standartları için genel bir yazılım süreç modelleme ontolojisi oluşturulmuştur. Liao et al. ile benzer sebeplerle bu tezden ayrılmaktadır.

Garcia et al. [69] yazılım süreçlerini modellemenin ve ölçmenin entegre yönetimi için bir çerçeve önermiştir. Bu çalışmada, yazılım süreçlerinin modellenmesini ve ölçümünü bütünleştirmek için, tüm modellerin ve meta-modellerin aynı kavramsallaştırmayı (nesneler, kavramlar, varlıklar ve bu varlıklar arasındaki ilgilenilen alanlarda olması beklenen ilişkileri) temel alması gerektiği iddia edilmiştir. Çalışmada ayrıca, yazılım süreç modellerinin kavramsal mimarideki yeri açıklanmış, *tanımlayıcı yazılım süreç modelleme ontolojisi* olarak adlandırılan ontolojinin, kavramsal çatıdaki yeri verilmiştir. Bu ontolojinin, SPEM referans alınarak oluşturulacak bir ontoloji olduğunun belirtilmesiyle birlikte, çalışmada ontolojinin detaylarına yer verilmemiştir.

Garcia et al. çalışması, süreç tanımlama için SPEM'i esas alan bir ontolojinin kullanılması noktasında bu tez ile benzerlik taşımaktadır. Bununla birlikte, tezde esas odaklanılan noktanın süreç değerlendirme olması, Garcia et al. çalışmasında

odaklanılan konunun süreç modelleme ve ölçme olması, süreç değerlendirmenin ise çalışma kapsamında yer almaması, bunları birbirinden ayırmaktadır.

Lee et al. [70] ontoloji tabanlı çok ajanlı sayısal hesaplama ile ilgili bir çalışma yapmış ve CMMI değerlendirmesi için bu çalışmanın bir uygulamasını geliştirmiştir. Bu uygulamada CMMI ile ilgili özet bir değerlendirme raporu oluşturmak hedeflenmiştir. Bu bağlamda, CMMI'daki PPQA süreç alanını esas alan bir Kalite Güvencesi Ontolojisi, yine aynı kişiler tarafından oluşturulan Alan Ontoloji Yapısı (Domain Ontology Structure) temel alınarak oluşturulmuştur.

Lee et al. tarafından yapılan bir başka çalışmada [71], proje çalışanlarının işlerinin tamamlama yüzdelerini bulup çıkaran, bu bilgiden de kişilerin performansını tespit eden bir sistem geliştirilmesi hedeflenmiştir. Bunun için, OIDSA olarak isimlendirilen ontoloji tabanlı akıllı bir karar destek ajanı oluşturulmuştur. OIDSA için geliştirilen ontoloji, CMMI'daki REQM, PP, PMC süreç alanlarına ait bilginin Alan Ontoloji Yapısı (Domain Ontology Structure) temel alınarak ifade edilmesi ile oluşturulmuştur.

CMMI modelinin kendine özgü bir yapılanması (CMMI'daki bileşen yapısı) vardır ve modeldeki en iyi pratikler bu yapılanmaya uygun olarak ifade edilmişlerdir. Lee et al. tarafından yapılan iki çalışmada da, ontolojiler, CMMI'ın bu yapısına göre değil, oluşturdukları Alan Ontoloji Yapısı temel alınarak geliştirilmiştir. Bu tez kapsamında oluşturulan CMMI ontolojisi ise, başka bir üst ontoloji kullanılmadan, CMMI modelinin kavramları ve yapılanması göz önünde bulundurularak oluşturulmuştur. Çalışmalar bu noktalarda birbirinden ayrılmaktadır.

3.3. Süreç Değerlendirmeyi Destekleyen Bazı Araçlar

Bu bölümde süreç iyileştirme ve değerlendirme faaliyetlerini destekleyen araçlarla ilgili kısa açıklamalara yer verilmiştir. Araçlar, yazılım kalite iyileştirmenin otomatik desteklenmesi ile ilgili bazı bildiriler [25] [66] ve SPI Partners [72] web sayfası göz önünde bulundurularak seçilmiş ve incelenmiştir. Leung et al. [25] yazılım kalite iyileştirmeyi destekleyen 46 aracı inceledikleri çalışmada, araçların çoğunun veri saklama ortamı için MS Excel ve MS Access kullandığını belirtmiştir.

Appraisal Assistant [73]: Griffith Üniversitesi, Yazılım Kalitesi Enstitüsü tarafından geliştirilmiş bir yazılım uygulamasıdır. Bu uygulama, süreçlerin yeteneğinin veya organizasyonların süreç olgunluğunun değerlendirilmesini desteklemektedir. CMMI değerlendirme gerekleri ve ISO/IEC 15504'ün gereklerine uygun bir yaklaşım izlenerek oluşturulmuştur. Model ve eşleştirme verileri MS Access veri tabanında tutulmaktadır.

CMMiPal v1.0 [74]: Chemuturi Consultants [75] tarafından geliştirilmiş olan araç, CMMI pratikleri ile kurum süreçlerinin eşleştirilerek süreç değerlendirmeyi destekleyen bir araçtır. Model ve eşleştirme verileri MS Access veri tabanında tutulmaktadır.

CMM-Quest v1.2 [76]: HM&S IT-Consulting [77] tarafından geliştirilmiş olan ve CMMI-Dev v1.2'ye göre yapılacak bir süreç değerlendirme faaliyetinin hazırlık, verilerin toplanması, analizi ve raporlanması ile ilgili adımları destekleyen bir yazılım aracıdır. Hazırlık aşaması için süreç alanı ve seviye seçilmesi, veri toplama için metin ekranlar, analiz için grafikler, raporlama için MS Word ve HTML formatında rapor üretme yetenekleri sağlamaktadır. Kurumsal süreçlerin modellenmesi için bir altyapısı yoktur.

SPICE 1-2-1 [78]: HM&S IT-Consulting [77] tarafından geliştirilmiş olan ve ISO/IEC 15504'e göre süreç değerlendirmeyi destekleyen bir yazılım aracıdır.

SPiCE-Lite Tool [79]: HM&S IT-Consulting [77] tarafından geliştirilmiş, ISO 15504 ile uyumluluğu değerlendirilmesini destekleyen bir araçtır. Veriler, veritabanında saklanmaktadır.

Model Wizard [80]: Integrated System Diagnostics (ISD) Incorporated [81] tarafından geliştirilmiş Windows tabanlı bir uygulamadır. Bu ürün, kullanıcıların süreç modellerini bir veritabanına kaydetmesini sağlar [82].

Appraisal Wizard [83]: Integrated System Diagnostics (ISD) Incorporated [81] tarafından geliştirilmiş Microsoft Windows tabanlı istemci-sunucu mimarili bir yazılım ürünüdür. Bu ürünün amacı, süreçlerin değerlendirilmesinin, tetkikinin ve uyumluluk kontrolünün; planlanması, hazırlanılması, verilerin toplanması ve birleştirilmesi, raporlanması ile ilgili yönetimi ve yürütmeyi desteklemektir [84].

Model Wizard uygulaması ile bütünleşik çalışabilmektedir. Bu üründe de veriler bir veritabanında saklanmaktadır.

CMMI v1.2 Self Assessment Tool [85]: Management Information Systems bvba [86] (bvba; Belçika'da sınırlı sorumlu bir organizasyon biçimidir) tarafından MS Excel ile oluşturulmuş değerlendirme aracıdır.

4. CMMI ONTOLOJİSİ VE OCMQ-E

Bir yazılım ürünün geliştirilmesinde ve idamesinde kullanılan süreçlerin, bu ürünün kalitesi üzerindeki etkisi ve süreç iyileştirmenin ve değerlendirmenin önemi daha önceki bölümlerde ifade edilmişti. Bu bağlamda, yazılım süreç iyileştirme ve değerlendirme faaliyetlerini desteklemek amacıyla yazılım araçlarından faydalanılabileceği de vurgulanmıştı. Aynı araç ile hem yazılım süreç iyileştirme hem de süreç değerlendirme etkinlikleri desteklendiğinde ise bunun ek yararlar sağlayacağına değinilmişti.

Süreç referans modelleri ve/veya standartları ile yazılım süreç modelleme araçları bütünleştirilirse, bu araçların süreç değerlendirmeyi desteklemesi için önemli bir adım atılmış olacaktır. Buna ilâve olarak, bütünleştirilen referans modellerin ve/veya standartların, kurumsal süreçler ile eşleştirilmesi ve bu eşleştirmelerin sorgulanabilmesi de böyle bir amaca ulaşmakta önemlidir. Ontolojiler, böyle bir amaca ulaşmak için uygun bir çözüm olacaktır.

Ontolojiler, süreç modellerinin/standartlarının biçimsel ve paylaşılabilir bir dil ile ifade edilebilmesine olanak sağlayacaktır. Böylece, modeller/standartlar, yazılım araçları tarafından anlaşılabilir ve bu analizin tekrar tekrar yapılmasına gerek kalmayacak şekilde paylaşılabilir olacaklardır. Ayrıca, bu modeller/standartlar için oluşturulacak ontolojiler, bu konuda çalışan kişiler için ortak bir dil de oluşturacaktır. Yazılım araçlarına dahil edilen model/standart ontolojileri, ontolojilerin sorgulanabilir olması sayesinde, model/standart alan bilgisinin sorgulanmasına da imkân verecektir.

Bir kurumun süreç modellerinin de ontolojilerle ifade edilmesinin bazı faydaları vardır. Liao et al., ontolojilerin, süreç modellerinin kullanımı ile ilgili bazı problemleri nasıl bertaraf edeceğini ve zorlukları nasıl kolaylaştıracağını şu şekilde ortaya koymuştur [67]:

1. *Süreç modellerinin biçimsel tanımı:* Süreç modelleri, model yapısının ve süreç çatısının kesin ve biçimsel bir şekilde tanımından yoksundur. Mevcut süreç modellerinde, süreç tetkiklerindeki ve uygulamalarındaki; belirsizlik, değişkenlik, çok fazla öznellik ve yanlışlıklardan kaynaklanan problemler

vardır. Ontolojiler, kavramsal ve terminolojik karmaşayı ortadan kaldırır ve yazılım süreçlerine özel gösterim sözcükleri sağlar.

2. *Uyumluluk ve dönüşümlülük:* Modeller arasındaki uyumluluk ve dönüşümlülük, yazılım kurumları için temel gereksinimlerdir. Mevcut süreç modelleri için ontolojiler yaratarak ve ontoloji uyumlama teknikleri kullanarak, mevcut modelleri değiştirmenin maliyeti olmaksızın uyumluluk sorununu çözebiliriz.

3. *Süreç niteliklerinin karşılaştırmalı değerlendirmesi:* Anket yaparak veri toplamak zor olduğu için, mevcut modeller için literatürde bulunan karşılaştırmalı değerlendirme raporları azdır. Ontoloji ve anlamsal web ile Internet'ten veri toplamak ve bazı alanlardaki yazılım süreçlerinin karşılaştırmalı değerlendirmelerini oluşturmak daha kolay olacaktır.

Kurum süreçlerinin ontolojilerle ifade edilmesinin bir başka faydası da, süreç referans modellerinin/standartlarının ontolojilerle ifade edildiği bir durumda, bunlarla eşleştirilebilir ve bu eşlemelerin sorgulanabilir olmasına imkân sağlayacak olmasıdır. Bu ise, süreç iyileştirme ve değerlendirme faaliyetlerine yardımcı olacak bir altyapı oluşturabilir.

Bu düşüncelerden hareketle, önerilen çözümün bir gerçekleştirimi olan ve yazılım süreç iyileştirme ve değerlendirme faaliyetlerini destekleyecek ontoloji-tabanlı bir araç geliştirilmesi hedeflenmiştir.

Bu aşamada süreç referans modeli olarak CMMI-Dev v1.2 seçilmiştir. CMMI-Dev, yaygın kabul gören, yazılım ve sistem geliştirme süreçlerinin iyileştirilmesi için kullanılan bir süreç referans modelidir. Kurum süreçlerinin ontolojilerinin oluşturulması adımı için ise SPEM'in esas alınması uygun görülmüştür. SPEM, OMG tarafından oluşturulmuş, yazılım ve sistem süreç mühendisliği meta-modelidir. Birçok büyük firma tarafından esas alınarak araç geliştirilen, kapsamlı bir meta-modeldir. SPEM esas alınarak yazılmış bir kurum sürecinin ontolojisinin oluşturulması, hedeflenen amaç için uygun olacaktır.

Bu noktada EPF süreç yönetimi aracı, bu hedefe ulaşmada bir başlangıç noktası oluşturacak yapıya sahiptir. SPEM'in bir gerçekleştirimi olan, Eclipse tabanlı, açık kaynak kodlu olan bu araç, yazılım eklentileri ile genişletilebilir yapısı ile, uygun bir

başlangıç noktası olarak görülmüştür. Bu düşüncelerden hareketle geliştirilen iki Eclipse eklentisi ile hedeflenen çalışma gerçekleştirilmiştir.

Çözümün adımları şöyledir;

- Hedeflenen amaca hizmet edecek bir CMMI ontolojisi Protégé-OWL Editor kullanılarak oluşturulmuştur.
- Geliştirilen bir Eclipse eklentisi ile, bu ontoloji EPF ile bütünleştirilmiş ve ontolojinin EPF kullanıcı arayüzünde görüntülenmesi sağlanmıştır.
- Geliştirilen bir başka Eclipse eklentisi ile şunlar sağlanmıştır;
    - EPF'de yazılan bir sürecin OWL ontolojisinin oluşturulması,
    - CMMI ve süreç ontolojilerinin eşleştirilebilmesi ve bu eşleme bilgisinin ontoloji olarak saklanması,
    - CMMI ontolojisinin, süreç ontolojilerinin ve eşleme ontolojisinin sorgulanarak önemli görülen bazı bilgilerin listelenebilir olması.

Eklentiler ile genişletilen EPF aracına, OCMQ-E (Ontology-based CMMI Mapping and Querying EPF) takma adı verilmiştir. OCMQ-E'nin geliştirimi tamamlandıktan sonra, kullanımı ASELSAN A.Ş. REHİS Grubu Yazılım Geliştirme Süreci ile örneklenmiştir. Yazılım Geliştirme Süreci, OCMQ-E'nin mevcut EPF yetenekleri kullanılarak yazılmış, ontolojisi oluşturulduktan sonra süreçteki adımlar CMMI pratikleri ile eşleştirilmiştir. Sorguların çalıştırılması ile ilk olarak, CMMI alan bilgisinin sorgulanması bağlamında; CMMI'daki süreç alanları, bir süreç alanında yer alan hedefler, pratikler ve bunlarla ilgili kısa açıklamaların kullanıcı arayüzünde görüntülenmesi sağlanmıştır. Daha sonra, süreç ontolojisinin sorgulanmasına örnek olarak, Yazılım Geliştirme Süreci'ndeki aktiviteler, görevler, iş ürünleri ve roller listelenmiştir. Son olarak ise, süreç değerlendirme ve iyileştirme faaliyetlerinde kullanılması öngörülen CMMI-süreç eşleme bilgilerinin sorgulanması ve sonucun listelenmesi ile örnekleme tamamlanmıştır. Bu sorguların ilki, seçilen bir CMMI pratiği ile eşleştirilmiş olan süreçlerin ve süreç varlıklarının listelenmesini sağlayan sorgudur. Bu sorgu bağlamında, hiçbir süreç varlığı ile eşleştirilmemiş olan CMMI pratikleri de listelenebilmektedir. Diğer sorgu

ise, seçilen bir süreç varlığı ile eşleştirilmiş olan CMMI bileşenlerinin listelenmesini sağlayan sorgudur.

CMMI ontolojisinin ve OCMQ-E'nin geliştirilmesi ile ilgili ayrıntılar, ilerleyen bölümde iki alt başlıkta ele alınmaktadır. Birinci alt başlıkta, geliştirilen CMMI ontolojisi ile ilgili detaylı açıklamalara, ikinci alt başlıkta ise, OCMQ-E ile ilgili detaylara yer verilmiştir.

4.1. CMMI Ontolojisi

4.1.1. Amaç ve Kapsam

Bu tez kapsamında oluşturulan CMMI ontolojisinin birincil amacı, CMMI-Dev v1.2'ye göre yapılacak bir süreç değerlendirme faaliyetini destekleyecek şekilde, kurumsal süreç tanımları ile CMMI'ın eşleştirilmesine olanak sağlayacak bir ontoloji olmasıdır. Bununla birlikte, ikincil amaç olarak da, bu ontolojinin CMMI-Dev v1.2'nin alan bilgisini mümkün olduğunca yansıtması hedeflenmiştir.

Ontoloji oluşturulurken model, CMMI-Dev v1.2'deki iki gösterimi de içeren, ancak IPPD'yi hariç tutan bir kapsamda ele alınmıştır.

Ontoloji, Protégé-OWL Editor v3.3.1 kullanılarak oluşturulmuş ve OWL dili ile ifade edilmiştir.

4.1.2. Ontolojinin Geliştirilmesi

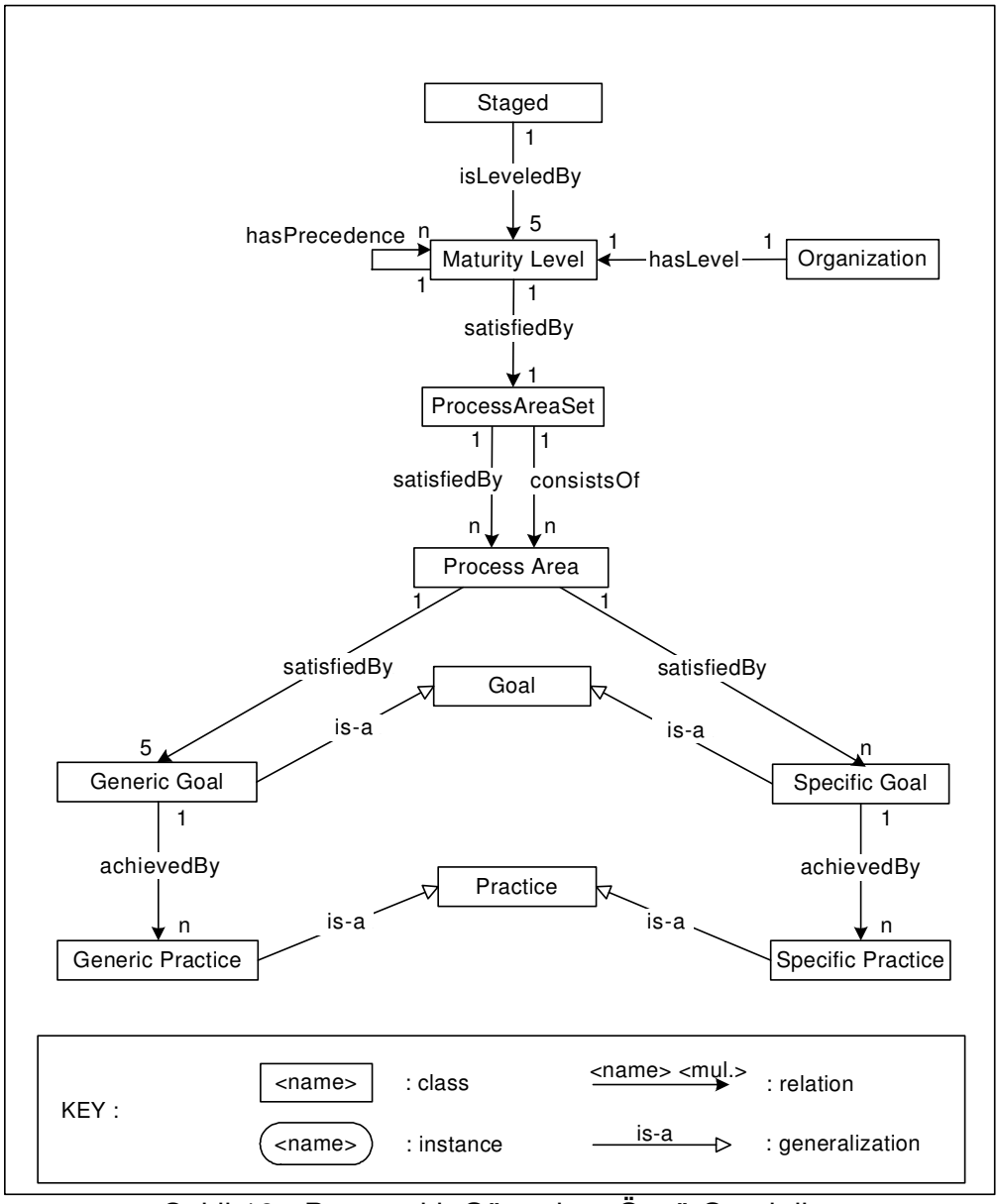
Çalışmanın başlangıcında, CMMI'daki gösterimlerin (basamaklı ve sürekli) her birine özgü ontolojiler oluşturulmuştur. Daha sonra, oluşturulan bu iki ontoloji incelendiğinde, bunların ayrılan kısımları olmasına rağmen ortak kısımlar da içerdiği ve her iki gösterimi de kapsayacak şekilde bunların birleştirilebileceği görülmüştür. Böyle bir birleştirmenin, bir gösterimden diğer gösterime geçiş yapılabilmesine olanak sağlayacağından dolayı, faydalı olacağı düşünülmüştür. Örneğin, bir gösterime göre tespit edilen seviyenin, diğer gösterimde hangi seviyeye karşılık geldiği kolayca görülebilecektir.

İlerleyen bölümde, önce gösterimler için ayrı ayrı oluşturulan ontolojilere kısaca yer verilmiştir. Daha sonra ise, birleştirilmiş CMMI Ontolojisi detaylandırılarak anlatılmıştır. Birleştirilmiş CMMI Ontolojisindeki ilişkiler için verilen açıklamalar,

gösterimler için ayrı ayrı oluşturulan ontolojilerde yer alan ilişkilerin anlaşılması için de yeterli bilgi içermektedir.

Basamaklı Gösterim İçin Oluşturulan Ontoloji

Basamaklı gösterime özgü ontolojideki kavramlar ve bu kavramlar arasındaki ilişkiler Şekil 19'da verilmiştir. Sadece bu gösterime özel olan kavramlar; Basamaklı Gösterim (Staged), Olgunluk Seviyesi (Maturity Level), Organizasyon (Organization) ve Süreç Alanı Kümesi (ProcessAreaSet) kavramlarıdır. Süreç Alanı (Process Area), Hedef (Goal), Pratik (Practice), Genel Hedef (Generic Goal), Genel Pratik (Generic Practice), Özel Hedef (Specific Goal), Özel Pratik (Specific Practice) kavramları ise her iki gösterimde de ortak olan kavramlardır.

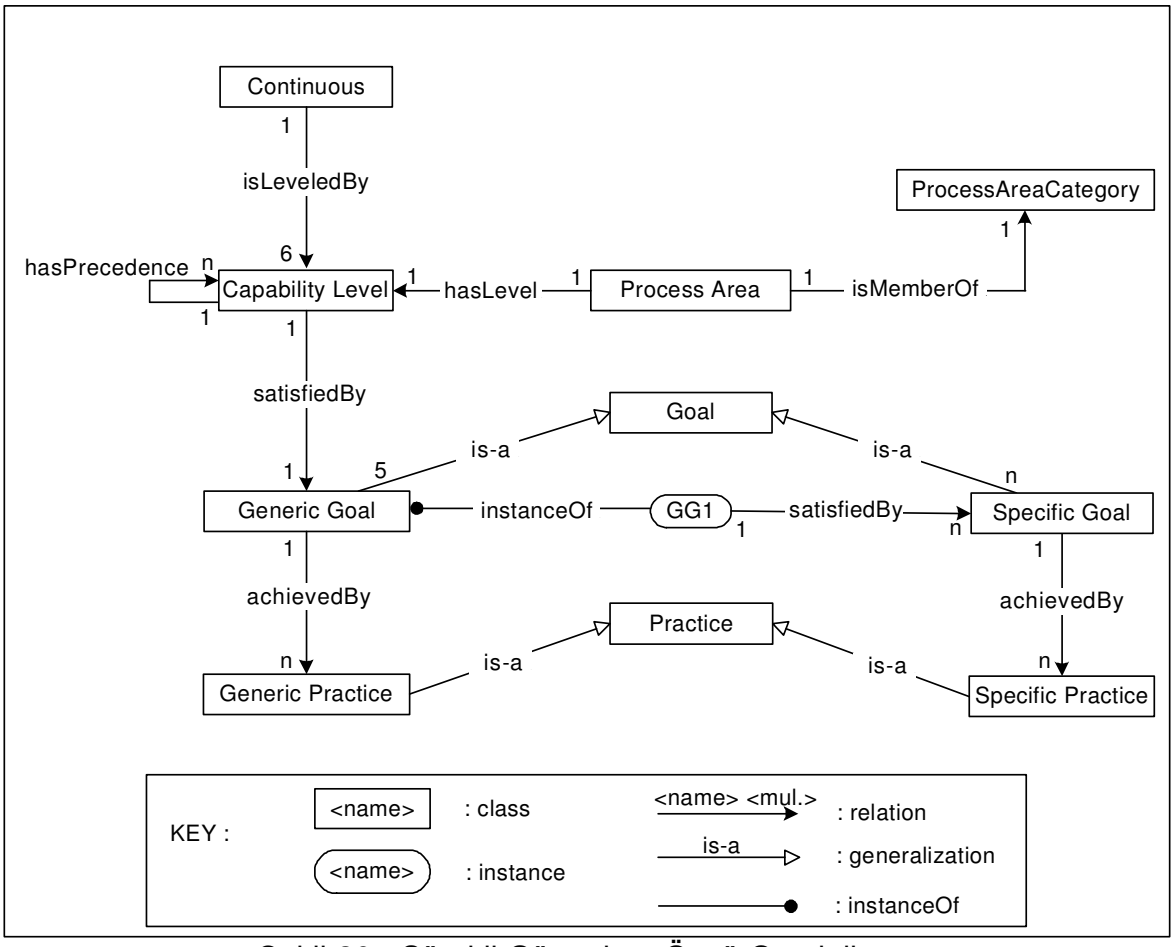


Şekil 19 - Basamaklı Gösterime Özgü Ontoloji

Süreç Alanı ile Genel Hedef ve Özel Hedef arasında yer alan *satisfyBy* ilişkisi dışında, gösterimlere özgü ontolojilerdeki ortak kavramların ilişkileri de ortaktır.

Sürekli Gösterim İçin Oluşturulan Ontoloji

Sürekli gösterime özgü ontolojideki kavramlar ve bu kavramlar arasındaki ilişkiler Şekil 20'de verilmiştir. Sadece bu gösterime özel olan kavramlar; Sürekli Gösterim (Continuous), Yetenek Seviyesi (Capability Level), Süreç Alanı Kategorisi (ProcessAreaCategory) kavramlarıdır. Diğerleri ise her iki gösterimde de ortaktır. İlişkiler bağlamında ise, Genel Hedef ve Özel Hedef nesnelerine doğru olan *satisfyBy* ilişkileri hariç, gösterimlerdeki ortak kavramların ilişkileri de ortaktır.



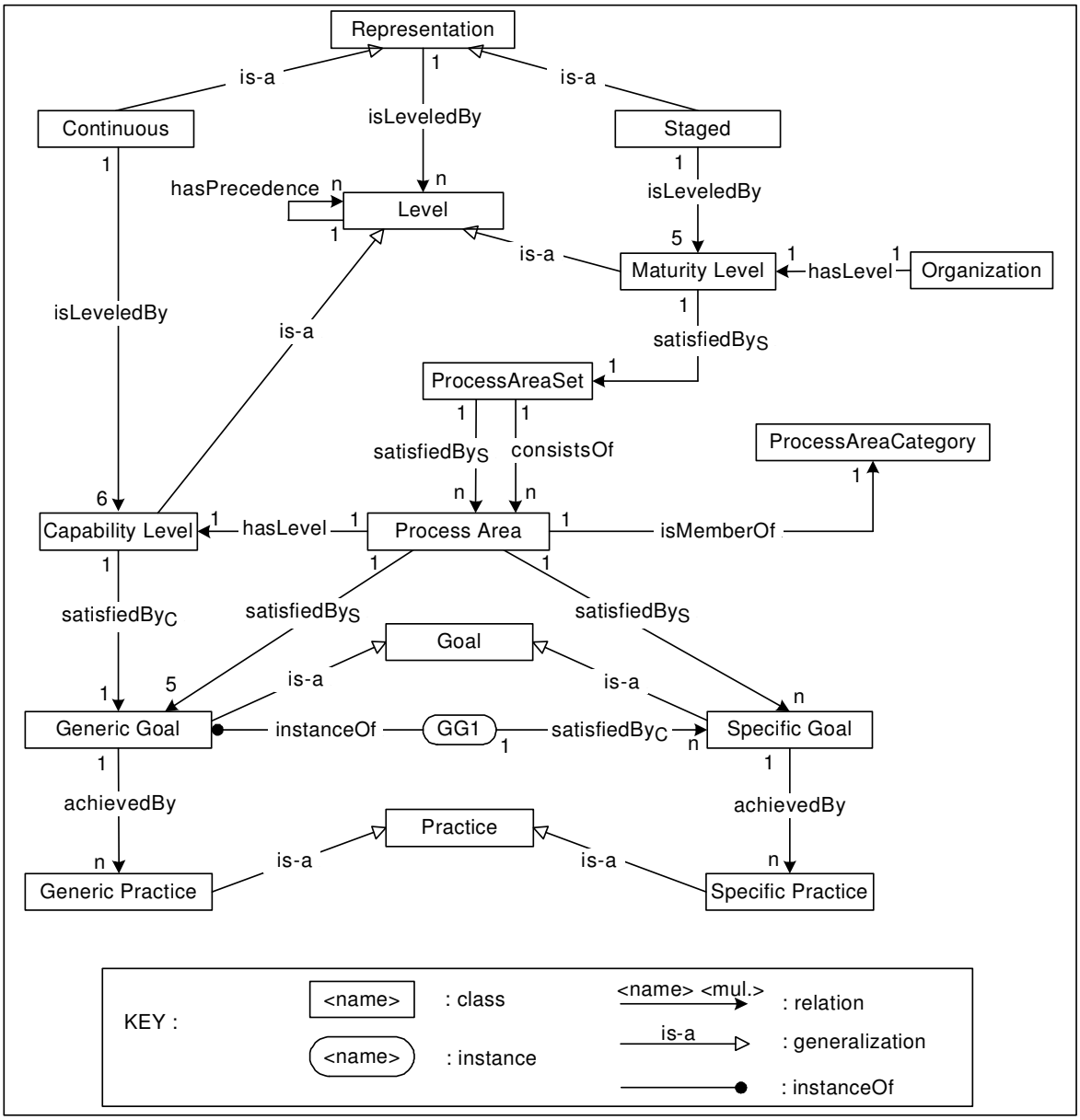
Şekil 20 - Sürekli Gösterime Özgü Ontoloji

Gösterimler için ayrı ayrı oluşturulan ontolojiler incelendiğinde görüleceği üzere, bazı kavramlar bir gösterime özel, bazıları ise ortaktır. Bunlar birleştirilirken, ikisini de kapsaması hedeflendiği için, tüm kavramlar birleştirilmiş CMMI Ontolojisine taşınmıştır. Genel Hedef ve Özel Hedef nesnelerine doğru olan *satisfyBy* ilişkileri

hariç, gösterimlere özgü ontolojilerdeki ortak kavramlar arasındaki ilişkiler de ortaktır.

İki Gösterimi de Kapsayan Birleştirilmiş Ontoloji

Her iki gösterimi de kapsayan birleştirilmiş CMMI Ontolojisinde yer alan kavramlar ve bunlar arasındaki ilişkiler Şekil 21'de görülmektedir.



Şekil 21 - Birleştirilmiş CMMI Ontolojisi

Birleştirilmiş CMMI Ontolojisinde yer alan ilişkiler ile ilgili açıklamalar aşağıda verilmiştir.

is-a ilişkisi: Bu ilişki, sınıflar arasındaki üst sınıf – alt sınıf ilişkisini göstermektedir.

hasLevel ilişkisi: Organizasyonlar, süreçlerinin kurumsallığı açısından değerlendirildiğinde Olgunluk Seviyeleri ile derecelendirilir. Bu ilişki, ontolojide <Organization> ve <Maturity Level> nesneleri arasındaki <hasLevel> ilişkisi ile gösterilmiştir. CMMI'daki süreç alanlarının bir organizasyondaki yeteneklerinin model ile karşılaştırılarak ayrı ayrı değerlendirilmesi sonucunda ise, her bir süreç alanı Yetenek Seviyesi ile derecelendirilir. Bu ilişki ise <Process Area> ve <Capability Level> nesneleri arasındaki <hasLevel> ilişkisi ile ifade edilmiştir.

isLeveledBy ilişkisi: Basamaklı Gösterimde derecelendirmenin Olgunluk Seviyeleri ile yapılması, ontolojide <Staged> ve <Maturity Level> nesneleri arasındaki <isLeveledBy> ilişkisi ile gösterilmiştir. Sürekli Gösterimde ise derecelendirmenin Yetenek Seviyeleri ile yapılması, ontolojide <Continuous> ve <Capability Level> nesneleri arasındaki <isLeveledBy> ilişkisi ile ifade edilmiştir.

hasPrecedence ilişkisi: CMMI'da her seviye, kendinden önceki seviyelerin gereklerinin de karşılanmış olmasını gerektirmektedir. Bu durum, ontolojide, <Level> nesnesi üzerindeki <hasPrecedence> ilişkisi ile gösterilmiştir. Bu ilişki, Sürekli ve Basamaklı Gösterimler için ayrı ayrı oluşturulmuş olan ontolojilerde <Capability Level> ve <Maturity Level> nesneleri üzerinde tanımlanmış iken, birleştirilmiş CMMI Ontolojisinde <Capability Level> ve <Maturity Level> nesnelerinin atası olan <Level> nesnesine taşınmıştır.

consistsOf ilişkisi: CMMI'da seviye 2'den 5'e kadar (2 ve 5 dahil) olmak üzere dört Olgunluk Seviyesi vardır ve modeldeki süreç alanlarının her biri bu seviyelerden yalnız birinde yer almaktadır. Bu ilişki, <ProcessAreaSet> ve <Process Area> nesneleri arasındaki <consistsOf> ilişkisi ile gösterilmiştir.

isMemberOf ilişkisi: CMMI'da süreç alanları, her süreç alanı bir kategoride olmak üzere toplam dört kategoride ele alınmaktadır. Bu durum <Process Area> ve <Process Area Category> nesneleri arasındaki <isMemberOf> ilişkisi ile ifade edilmiştir.

satisfiedBy$_S$ ve satisfiedBy$_C$ ilişkileri: Bu ilişkide yer alan "S" alt simgesi Basamaklı (Staged) Gösterimde yer alan <satisfyBy> ilişkisini, "C" alt simgesi ise Sürekli (Continuous) Gösterimde yer alan <satisfyBy> ilişkisini vurgulamak için kullanılmıştır.

Bir organizasyonun belli bir olgunluk seviyesine sahip olabilmesi için, bu olgunluk seviyesi ile ilişkilendirilmiş olan süreç alanlarının hepsinin başarı ile uygulanıyor olması gerekmektedir. Bu ilişki, <Maturity Level> ve <ProcessAreaSet> nesneleri arasındaki <satisfiedBy$_S$> ilişkisi ile gösterilmiştir. Bir olgunluk seviyesine karşılık gelen bir grup süreç alanının başarı ile uygulanıyor olması için ise, bu gruptaki süreç alanlarının her birinin başarı ile uygulanıyor olması gerekmektedir. Bu ilişki, <ProcessAreaSet> nesnesi ile <Process Area> nesnesi arasındaki <satisfiedBy$_S$> ilişkisi ile verilmiştir.

Hedefler modelin gerekli bileşenleridir ve seviyelendirmeler için bu hedeflere ulaşılıyor olması bir zorunluluktur. Model bileşenleri her iki gösterim için ortak olmakla birlikte, belli bir yetenek ve olgunluk seviyesi için ulaşılması gereken hedefler ve bu hedefleri işaret eden noktalar farklıdır. Basamaklı Gösterimde, bir süreç alanının başarı ile uygulanıyor olması, bu süreç alanının özel hedeflerine ve ulaşılmak istenen seviye ile ilişkili olan genel hedeflerine ulaşılması ile ilişkilendirilmiştir. Bu ilişki <Process Area> nesnesinden <Generic Goal> ve <Specific Goal> nesnelerine doğru olan <satisfiedBy$_S$> ilişkisi ile gösterilmektedir. Sürekli Gösterimde ise, her bir yetenek seviyesi bir genel hedef ile ilişkilendirilmiştir. Bu ilişki <Capability Level> nesnesinden <Generic Goal> nesnesine doğru olan <satisfiedBy$_C$> ilişkisi ile gösterilmiştir. Ayrıca, Sürekli Gösterimde özel hedeflere ulaşılmasını adresleyen nokta Basamaklı Gösterimdekinden farklıdır. Sürekli Gösterimde, bu gereklilik genel hedef 1'de (GG1) ifade edilmiştir. Tüm süreç alanlarının genel hedef 1'lerinde aynı ifade yer almaktadır ve burada özel hedeflere ulaşılması gerektiği belirtilmektedir. Bu durum <GG1> olgusundan <Specific Goal> nesnesine doğru çizilen <satisfiedBy$_C$> ilişkisi ile gösterilmiştir.

achievedBy ilişkisi: Bir hedefe, bu hedefe ait tüm pratikler başarı ile uygulandığında ulaşıldığı modelde belirtilmektedir. Hem genel hem de özel hedefler için geçerli olan bu ilişki; <Generic Goal> ile <Generic Practice> nesneleri arasında ve <Specific Goal> ile <Specific Practice> nesneleri arasında yer alan <achievedBy> ilişkisi ile ifade edilmiştir.

instanceOf ilişkisi: Bu ilişki, bir sınıf ve bu sınıfın bir olgusu arasındaki ilişkiyi göstermektedir.

CMMI Ontolojisinde yer alan sınıfların tamamı, bu sınıfların olgu sayıları ile birlikte ve sınıf-altsınıf hiyerarşisini gösterecek şekilde Çizelge 4'te verilmiştir.

Çizelge 4 - CMMI Ontolojisindeki Sınıflar

- Generic - GenericGoal - GenericGoal_1 (22 olgu) - GenericGoal_2 (22 olgu) - GenericGoal_3 (22 olgu) - GenericGoal_4 (22 olgu) - GenericGoal_5 (22 olgu)
- GenericPractice - GenericPractice_1 - GenericPractice_1.1 (22 olgu) - GenericPractice_2 - GenericPractice_2.1 (22 olgu) - GenericPractice_2.2 (22 olgu) - GenericPractice_2.3 (22 olgu) - GenericPractice_2.4 (22 olgu) - GenericPractice_2.5 (22 olgu) - GenericPractice_2.6 (22 olgu) - GenericPractice_2.7 (22 olgu) - GenericPractice_2.8 (22 olgu) - GenericPractice_2.9 (22 olgu) - GenericPractice_2.10 (22 olgu) - GenericPractice_3 - GenericPractice_3.1 (22 olgu) - GenericPractice_3.2 (22 olgu) - GenericPractice_4 - GenericPractice_4.1 (22 olgu) - GenericPractice_4.2 (22 olgu) - GenericPractice_5 - GenericPractice_5.1 (22 olgu) - GenericPractice_5.2 (22 olgu)

- Goal - *Generic Goal (Generic sınıfının altsınıfı olarak belirtilmiştir.)* - *Specific Goal (Specific sınıfının altsınıfı olarak belirtilmiştir.)*
- Level - CapabilityLevel (6 olgu) - MaturityLevel (5 olgu)
- Organization
- Practice - *GenericPractice (Generic sınıfının altsınıfı olarak belirtilmiştir.)* - *SpecificPractice (Specific sınıfının altsınıfı olarak belirtilmiştir.)*
- ProcessArea (22 olgu)
- ProcessAreaCategory (4 olgu)
- ProcessAreaSet (4 olgu)
- Representation - Continuous - Staged
- Specific - SpecificGoal (48 olgu) - SpecificPractice (165 olgu)

4.2. OCMQ-E

OCMQ-E; EPF'nin, CMMI esaslı süreç iyileştirmeyi ve değerlendirmeyi destekleyecek şekilde, Eclipse eklentileri ile genişletilmesi ile oluşturulan araçtır. Yeni eklenen yetenekler, ontoloji-tabanlı bir yapı oluşturularak gerçeklenmiştir. OCMQ-E ile temelde hedeflenen; CMMI ile kurumun süreç tanımları arasındaki uyumluluğun takip edilebilmesi ve süreç değerlendirme faaliyetinin veri toplama adımının desteklenmesidir. Bu amaca, EPF'ye eklenen iki ana işlev ile ulaşılmıştır. Bunlar; Eşleme ve Sorgulama işlevleridir.

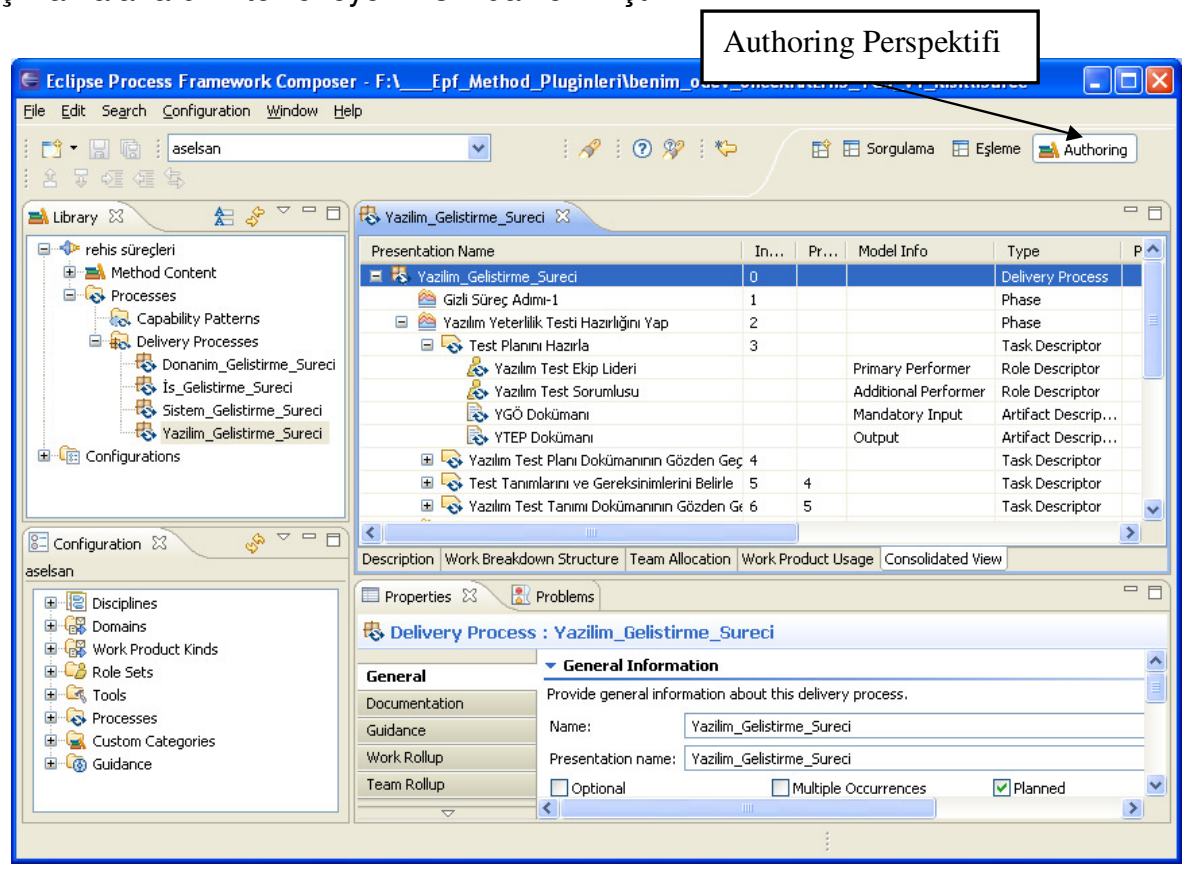
- Eşleme işlevi ile, temel olarak; istenen bir sürecin ontolojisinin oluşturulabilmesi, seçilen bir CMMI bileşeni ile seçilen bir süreç varlığı arasında eşleme kurulabilmesi veya kurulu olan bir eşlemenin

silinebilmesi, eşleme bilgilerinin bir ontolojide saklanması alt işlevleri gerçekleştirilmektedir.

- Sorgulama işlevi ile; CMMI ontolojisi, süreç ontolojileri ve CMMI ile süreç ontolojileri arasında kurulan eşlemelerin yer aldığı ontolojinin sorgulanması ile ilgili alt işlevler gerçekleştirilmektedir.

Bu iki ana işlevin her biri için yeni birer perspektif yaratılarak, EPF'nin varsayılan perspektifi olan *Authoring* etkilenmeyecek şekilde, yeni işlevler kullanıcılara sunulmuştur. Oluşturulan yeni perspektifler; Eşleme ve Sorgulama perspektifleridir.

OCMQ-E'nin orijinal EPF'den gelen Authoring perspektifinin kullanıcı arayüzü görüntüsü Şekil 22'de, Eşleme ve Sorgulama pespektiflerinin görüntüleri ise kısa açıklamalarla birlikte ilerleyen kısımda verilmiştir.



Şekil 22 - EPF Varsayılan Perspektif: Authoring

Eşleme Perspektifi:

Bu perspektif ile, süreç ontolojilerinin yaratılması ve CMMI ile süreçler arasındaki eşlemelerin kurulması ve silinmesi ile ilgili işlemlerin yapılmasına olanak sağlayan

kullanıcı arayüzünün oluşturulması hedeflenmiştir (Şekil 23). Perspektifte; sol üst kısımda Kütüphane Görünümü (Library View), orta üst kısımda Süreç Editörü (Process Editor), sağ üst kısımda CMMI Ontoloji Ağacı Görünümü ve alt kısımda Eşleme Görünümü yer almaktadır. Bunların ilk ikisi (Kütüphane Görünümü ve Süreç Editörü) orijinal EPF bulunan bir görünüm ve bir editördür. Son ikisi ise (CMMI Ontoloji Ağacı Görünümü ve Eşleme Görünümü), tez kapsamında geliştirilen eklentiler ile EPF'ye kazandırılan görünümlerdendir.

CMMI Ontoloji Ağacı Görünümü ile; tez kapsamında geliştirilen CMMI Ontolojisinin kullanıcı arayüzünde görüntülenmesi ve ağaç yapısında görüntülenen CMMI Ontolojisindeki bileşenlerin seçilebilir olması sağlanmıştır.

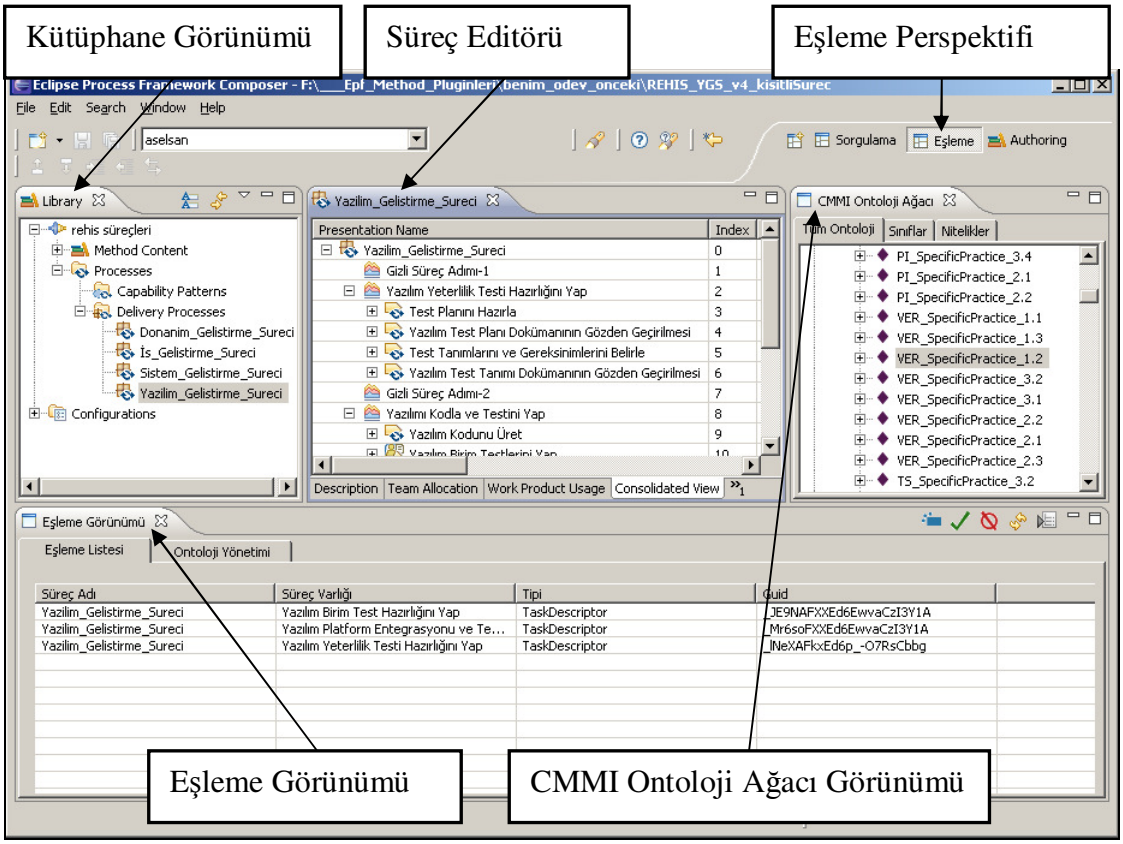
Eşleme Görünümünün *Eşleme Listesi* sekmesi ile sunulan işlevler şunlardır;

- Kütüphane görünümündeki Dağıtım Süreçleri (Delivery Processes) dizininden seçilen ve orijinal EPF yetenekleri kullanılarak oluşturulmuş olan bir sürecin ontolojisinin oluşturulması,
- Süreç editöründen seçilen ontolojisi oluşturulmuş bir sürece ait bir süreç varlığı ile CMMI Ontoloji Ağacı görünümünden seçilen bir CMMI bileşeni arasında eşleme kurulması,
- Eşleme Listesinden seçilen bir eşlemenin silinmesi,
- Seçilen bir CMMI bileşeni ile eşlenmiş olan süreç varlıklarının listelenmesini sağlayacak şekilde Eşleme Listesinin güncellenmesi,
- Eşleme Listesinden seçilen bir süreç varlığının yer aldığı sürecin, Süreç Editörü penceresinde açılması ve ilgili süreç varlığının seçili olarak işaretlenmesi.

Eşleme Görünümünün *Ontoloji Yönetimi* sekmesi ile sunulan işlevler şunlardır;

- OCMQ-E'de yüklü olan ontolojilerin adlarının yer aldığı Ontoloji Listesinin yenilenmesi,
- Ontoloji Listesinden seçilen bir ontolojinin yeniden yüklenmesi,

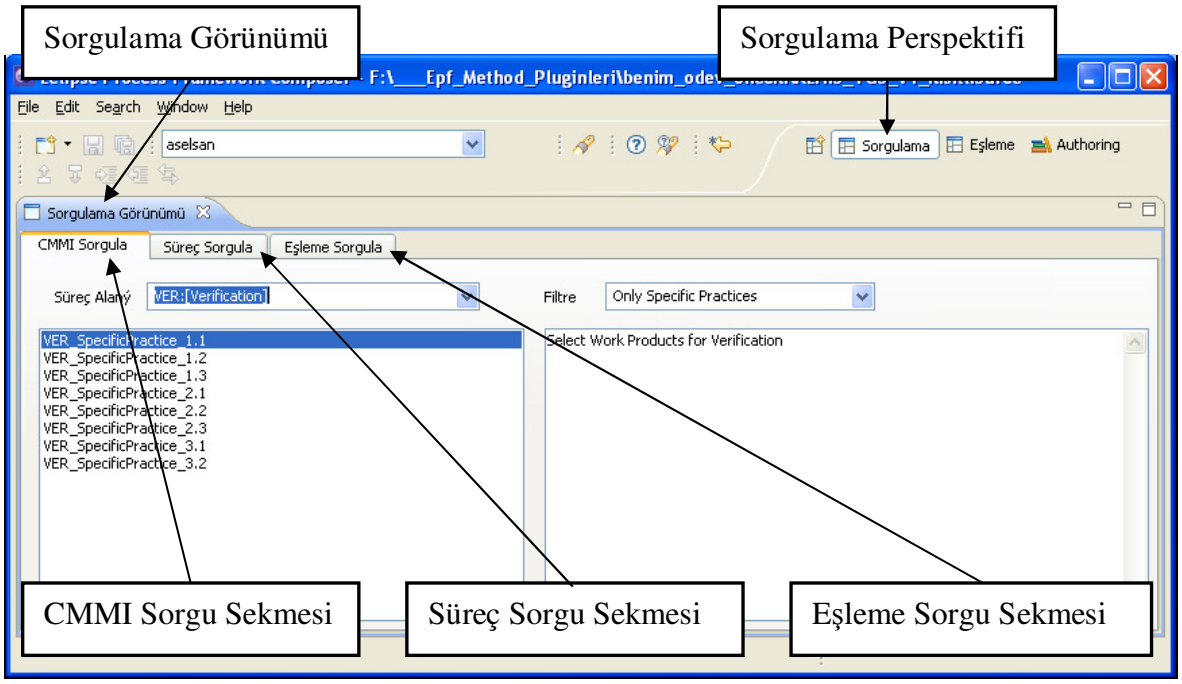
- Ontoloji Listesinden Eşleme Ontolojisi seçildiğinde, bu ontolojinin en baştan herhangi bir eşleme içermeyecek şekilde yaratılabilmesi.



Şekil 23 - OCMQ-E Eşleme Perspektifi

Sorgulama Perspektifi:

Bu perspektif, ontolojilerdeki önemli görülen bazı bilgilerin sorgulamalar yoluyla elde edilerek listelenmesini sağlayan Sorgulama görünümünün yer aldığı perspektiftir. Sorgulamalar üç kategoride ele alınmış ve her kategori ayrı bir sekmede gösterilmiştir. Bunlar; CMMI ile ilgili sorgulamalar, süreç ile ilgili sorgulamalar ve bunların eşlemeleri ile ilgili sorgulamalardır. Şekil 24'te, Sorgulama perspektifinin ekran görüntüsü, bahsedilen Sorgulama görünümü ve sekmeler görülmektedir.



Şekil 24 - OCMQ-E Sorgulama Perspektifi

4.2.1. OCMQ-E'nin Kapsamı ve Kullanımı

Modeller ve standartlar, süreç değerlendirmelerinde, hem kurumun süreç tanımlarının hem de bunların projelerdeki uygulanışının ele alınmasını gerektirmektedir. Ancak, süre kısıtından dolayı, tezin kapsamı kurumun süreç tanımları ile CMMI'ın süreç alanları arasındaki uyumun izlenmesi olarak belirlenmiştir.

OCMQ-E'de, süreçler ile CMMI süreç alanları arasındaki eşlemeler elle kurulduğundan, bu eşlemeleri yapacak kişilerin, kurum süreçleri ve CMMI ile ilgili bilgi sahibi olması gerekmektedir. Aracın, kalite güvencesi mühendisleri gibi, bir kurumun süreçleri ve CMMI hakkında bilgi sahibi olan ve bunlar arasındaki uyumluluğu takip etmekten sorumlu kişiler tarafından kullanılması beklenmektedir.

OCMQ-E ile süreç, model ve bunlar arasındaki eşlemelere ait bilgiye sahip olunduğu için, süreç iyileştirmeyi ve süreç değerlendirme etkinliklerinin veri toplama aşamasını desteklemenin mümkün olduğu değerlendirilmektedir. Bu destek, mevcut sorgulama yetenekleri kullanılarak sağlanabileceği gibi, ihtiyaç duyulan başka sorgulamaların eklenmesi yoluyla da sağlanabilecektir. Zira tez kapsamında, sadece önemli görülen bazı sorgulamalara yer verilebilmiştir.

4.2.2. Kullanılan Geliştirme Ortamı

OCMQ-E'yi oluşturmak için eklentiler ile genişletilen EPF, EPF versiyon 1.5'tir. Eclipse eklentileri ise, Eclipse SDK versiyon 3.4.1 kullanılarak geliştirilmiştir.

Çalışma zamanında uygulamanın bir parçası olan ontoloji yükleme işlevleri için Jena 2.5 Java API kütüphanesi, ontoloji yaratma ve sorgulama işlevleri için de Protégé OWL API kütüphanesi versiyon 3.3.1 kullanılmıştır.

OCMQ-E uygulamasının dışında yaratılarak, uygulama içinden sadece okuma amacıyla kullanılan ontolojilerin oluşturulması için, Protégé OWL Editör versiyon 3.3.1 kullanılmıştır.

4.2.3. Gereksinim Analizi

Hedeflenen OCMQ-E aracının oluşturulması için belirlenen gereksinimler alt bölümlerde verilmiştir.

4.2.3.1. Eşleme İle İlgili Gereksinimler

CMMI Ontolojisinin Görüntülenmesi İle İlgili Gereksinimler:

G1: CMMI ontolojisinin EPF'de görüntülenir olması gerekmektedir.

G2: Görüntülenen CMMI ontolojisine ait olgular ayrı ayrı seçilebilir olmalıdır.

Süreç Ontolojisinin Oluşturulması İle İlgili Gereksinimler:

G3: EPF'de oluşturulmuş bir sürecin OWL ontolojisi oluşturulabilmelidir.

G4: EPF'de oluşturulmuş bir sürecin OWL ontolojisi oluşturulurken, SPEM esas alınarak oluşturulmuş Süreç Yapısı Ontolojisi temel alınmalıdır.

Eşleme Kurulması İle İlgili Gereksinimler:

G5: Kullanıcı arayüzünden seçilen bir süreç varlığı, yine kullanıcı arayüzünden seçilen bir CMMI bileşeni ile eşleştirilebilir olmalıdır.

G6: Kullanıcı arayüzünden seçilen bir süreç varlığı, seçilen bir CMMI bileşeni ile eşleştirildiğinde, bu eşleme kullanıcı arayüzünde görüntülenmelidir.

G7: Kullanıcı arayüzünden seçilen bir süreç varlığı ile seçilen bir CMMI bileşeni eşleştirildiğinde, bu eşleme bir OWL dosyasında saklanmalıdır.

Eşleme Silinmesi İle İlgili Gereksinimler:

G8: Bir CMMI bileşeni ile bir süreç varlığı arasında kurulu olan bir eşleme (Bkz. G5) silinebilir olmalıdır.

G9: Kurulu bir eşleme silinirken, eşleme bilgisi kullanıcı arayüzündeki listeden silinmelidir.

G10: Kurulu bir eşleme silinirken, eşleme bilgisi, bu bilginin saklandığı OWL dosyasında silinmelidir.

Eşlemelerin Listelenmesi İle İlgili Gereksinimler:

G11: Bir CMMI bileşeni ile eşleştirilmiş olan süreç varlıkları, bu CMMI bileşeni ile ilişkilendirilmiş olarak kullanıcı arayüzünde listelenebilir olmalıdır.

4.2.3.2. Ontolojilerin Yönetimi İle İlgili Gereksinimler

Ontoloji-tabanlı olan OCMQ-E dahilinde, birçok ontoloji ele alınmaktadır. Bu ontolojilerle ilgili önemli görülen ihtiyaçları karşılayacak gereksinimlere bu kısımda yer verilmiştir.

G12: Yüklü olan ontolojilerin kullanıcı arayüzünden görülebilmesi gerekmektedir. Böylece, ontolojisi oluşturulmuş süreçler de görülebilecektir.

G13: Yüklü olan ontolojileri gösteren listenin yenilenebiliyor olması gerekmektedir.

G14: Tüm ontolojilerin yeniden yüklenebilmesi gerekmektedir.

G15: Eşleme Ontolojisinin en baştan boş bir şekilde yaratılabiliyor olması gerekmektedir.

4.2.3.3. Ontoloji Sorgulamaları İle İlgili Gereksinimler

CMMI Alan Bilgisinin Sorgulanması İle İlgili Gereksinimler:

G16: Bir süreç alanının *özel pratikleri* listelenebilir olmalıdır.

G17: Bir süreç alanının *genel pratikleri* listelenebilir olmalıdır.

G18: Bir süreç alanının *tüm pratikleri* listelenebilir olmalıdır.

G19: Bir süreç alanının *özel hedefleri* listelenebilir olmalıdır.

G20: Bir süreç alanının *genel hedefleri* listelenebilir olmalıdır.

G21: Bir süreç alanının *tüm hedefleri* listelenebilir olmalıdır.

Süreç Bilgisinin Sorgulanması İle İlgili Gereksinimler:

G22: Seçilen bir süreçte yer alan *aktiviteler* listelenebilir olmalıdır.

G23: Seçilen bir süreçte yer alan *görevler* listelenebilir olmalıdır.

G24: Seçilen bir süreçte yer alan *iş ürünleri* listelenebilir olmalıdır.

G25: Seçilen bir süreçte yer alan *roller* listelenebilir olmalıdır.

CMMI-Süreç Eşlemelerinin Sorgulanması İle İlgili Gereksinimler:

Bir CMMI bileşeni ile eşleştirilen süreç varlıklarının listelenmesi ile ilgili gereksinimler*;*

G26: Seçilen bir CMMI süreç alanının, herhangi bir süreç varlığı ile *eşlenen özel pratikleri* listenebilir olmalıdır.

G27: Seçilen bir CMMI süreç alanının, herhangi bir süreç varlığı ile *eşlenen genel pratikleri* listenebilir olmalıdır.

G28: Seçilen bir CMMI süreç alanının, herhangi bir süreç varlığı ile *eşlenen özel hedefleri* listenebilir olmalıdır.

G29: Seçilen bir CMMI süreç alanının, herhangi bir süreç varlığı ile *eşlenen genel hedefleri* listenebilir olmalıdır.

G30: Seçilen bir CMMI süreç alanının, hiçbir süreç varlığı ile *eşlenmemiş olan özel pratikleri* listenebilir olmalıdır.

G31: Seçilen bir CMMI süreç alanının, hiçbir süreç varlığı ile *eşlenmemiş olan genel pratikleri* listenebilir olmalıdır.

G32: Seçilen bir CMMI süreç alanının, hiçbir süreç varlığı ile *eşlenmemiş olan özel hedefleri* listenebilir olmalıdır.

G33: Seçilen bir CMMI süreç alanının, hiçbir süreç varlığı ile *eşlenmemiş olan genel hedefleri* listenebilir olmalıdır.

G34: Seçilen bir CMMI süreç alanının *tüm özel pratikleri* listenebilir olmalıdır.

G35: Seçilen bir CMMI süreç alanının *tüm özel hedefleri* listenebilir olmalıdır.

G36: Seçilen bir CMMI süreç alanının *tüm genel hedefleri* listenebilir olmalıdır.

G37: Seçilen bir CMMI süreç alanının *tüm genel pratikleri* listenebilir olmalıdır.

Bir süreç varlığının eşlendiği CMMI bileşenlerinin listelenmesi ile ilgili gereksinim;

G38: Seçilen bir süreç varlığının eşleştirildiği CMMI bileşenleri listelenebilir olmalıdır.

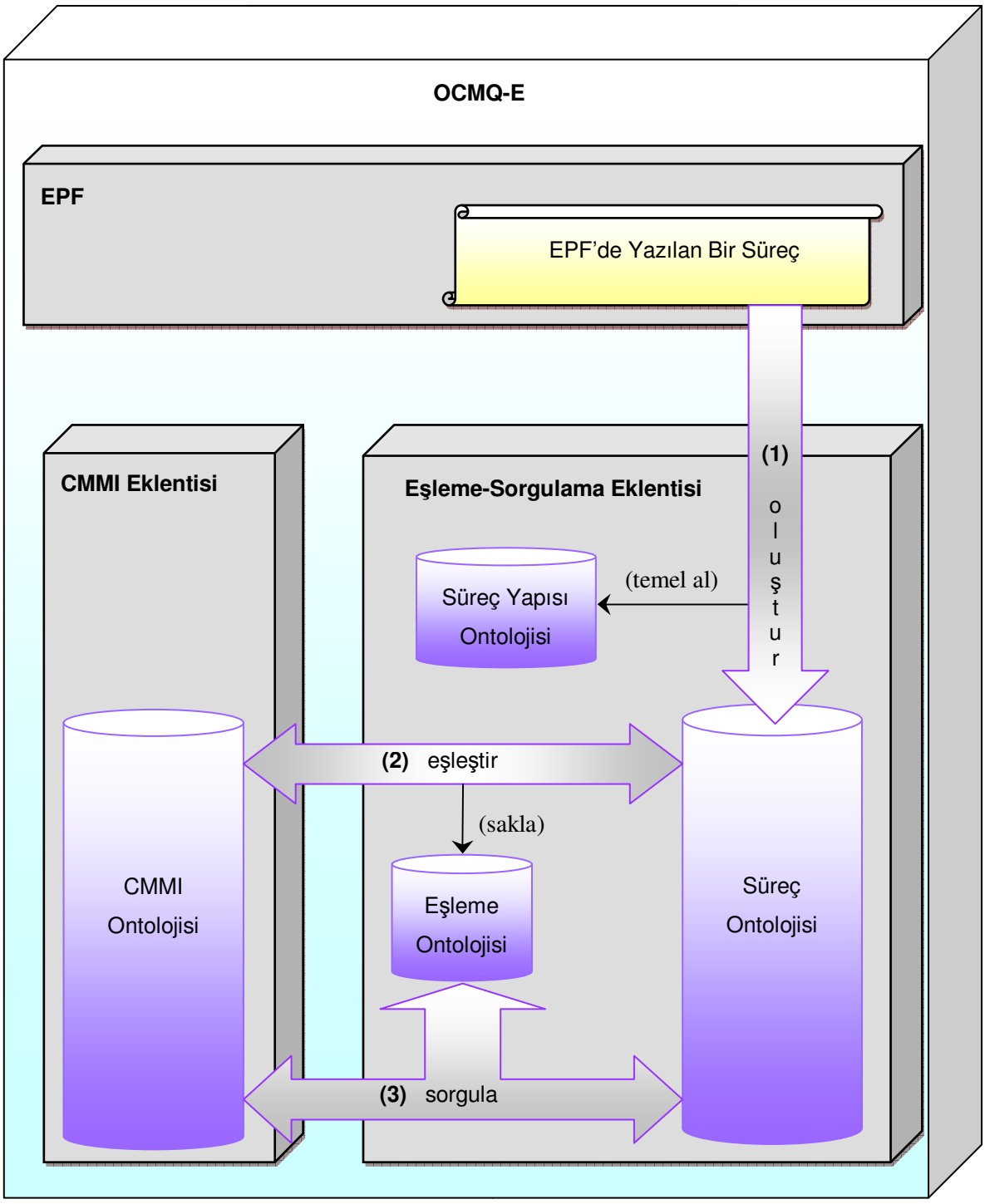
4.2.4. OCMQ-E'nin Mimarisi ve Tasarımı

OCMQ-E mimarisi Şekil 25'te verilmiştir. Şekilden de görülebileceği gibi, OCMQ-E iki Eclipse eklentisinin EPF'ye eklenmesi ile oluşturulmuştur. Bu eklentiler; CMMI Ontolojisi eklentisi ve Eşleme-Sorgulama eklentisidir.

CMMI Ontolojisi eklentisi; (1) bir OWL dosyasında yer alan CMMI Ontolojisinin okunmasını ve ontolojinin EPF kullanıcı arayüzünde gösterilmesini, (2) görüntülenen CMMI bileşenlerinin seçilebilir olmasını sağlayan eklentidir.

Eşleştirme-Sorgulama eklentisi ise; (1) EPF'de yazılan süreçlerin ontolojilerinin oluşturulmasını, (2) oluşturulan süreç ontolojileri ile CMMI ontolojisinin eşleştirilmesini ve eşleme bilgisinin de bir ontolojide saklanmasını, (3) yüklü olan ontolojilerin (CMMI ontolojisi, süreç ontolojileri ve eşleme ontolojisi) sorgulanmasını sağlayan eklentidir.

CMMI ontolojisinin görüntülenmesi ile ilgili yeteneklerin başka uygulamalar tarafından da kullanılabilmesine, diğer bir deyişle yeniden kullanılabilirliğine olanak sağlamak amacıyla, CMMI eklentisi ayrı bir eklenti olarak geliştirilmiştir.



Şekil 25 - OCMQ-E Mimarisi

Tez kapsamında oluşturulan perspektifler, bu perspektiflerde yer alan ve yeni geliştirilen görünümler ve görünümlerin gerçeklendikleri eklentiler Çizelge 5'te verilmiştir.

Çizelge 5 - Yeni Eklenen Perspektifler, Görünümler ve Eklentiler Arasındaki İlişki

Perspektif	*Görünüm*	*Eklenti*
Eşleme Perspektifi	CMMI Ontoloji Ağacı Görünümü	CMMI Ontoloji Eklentisi
	Eşleme Görünümü	Eşleme-Sorgulama Eklentisi
Sorgu Perspektifi	Sorgulama Görünümü	Eşleme-Sorgulama Eklentisi

4.2.4.1. CMMI Eklentisi

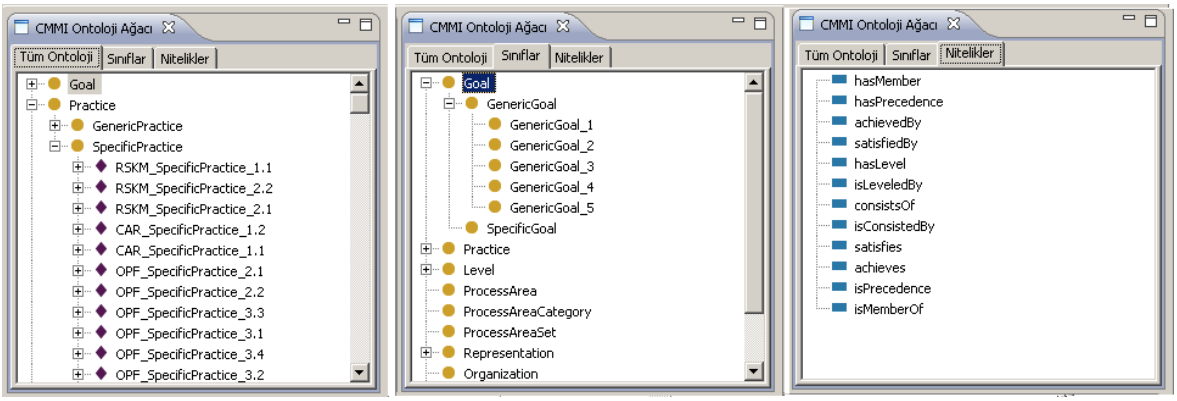
CMMI ontolojisinin EPF'de görüntülenmesi ve ağaç yapısında görüntülenen CMMI bileşenlerinin seçilebilir olması ile ilgili gereksinimler (G1 ve G2) bu eklenti dahilinde gerçekleştirilmiştir. Bir OWL dosyasında yer alan CMMI Ontolojisi, Jena kütüphanesi kullanılarak bu dosyadan okunmaktadır. Okunan ontoloji üç farklı şekilde kullanıcı arayüzünde görüntülenmek istenmiş ve bunların her biri ayrı sekmede görüntülenmiştir.

- Birinci sekmede; CMMI ontolojisindeki sınıflar, nitelikler ve olgular ağaç yapısında görüntülenmektedir. Ontolojideki sınıflar, sınıf–altsınıf ilişkisi içinde, nitelikler ve olgular ise ağaç yapısında ilişkili oldukları sınıfların altında yer alacak şekilde gösterilmektedir.

- İkinci sekmede; sadece sınıflar sınıf-altsınıf ilişkisi içinde ve ağaç yapısında gösterilmektedir.

- Üçüncü sekmede; sadece nitelikler listelenmektedir.

CMMI Ontoloji Ağacı görünümü ve sekmeler Şekil 26'da görülmektedir.

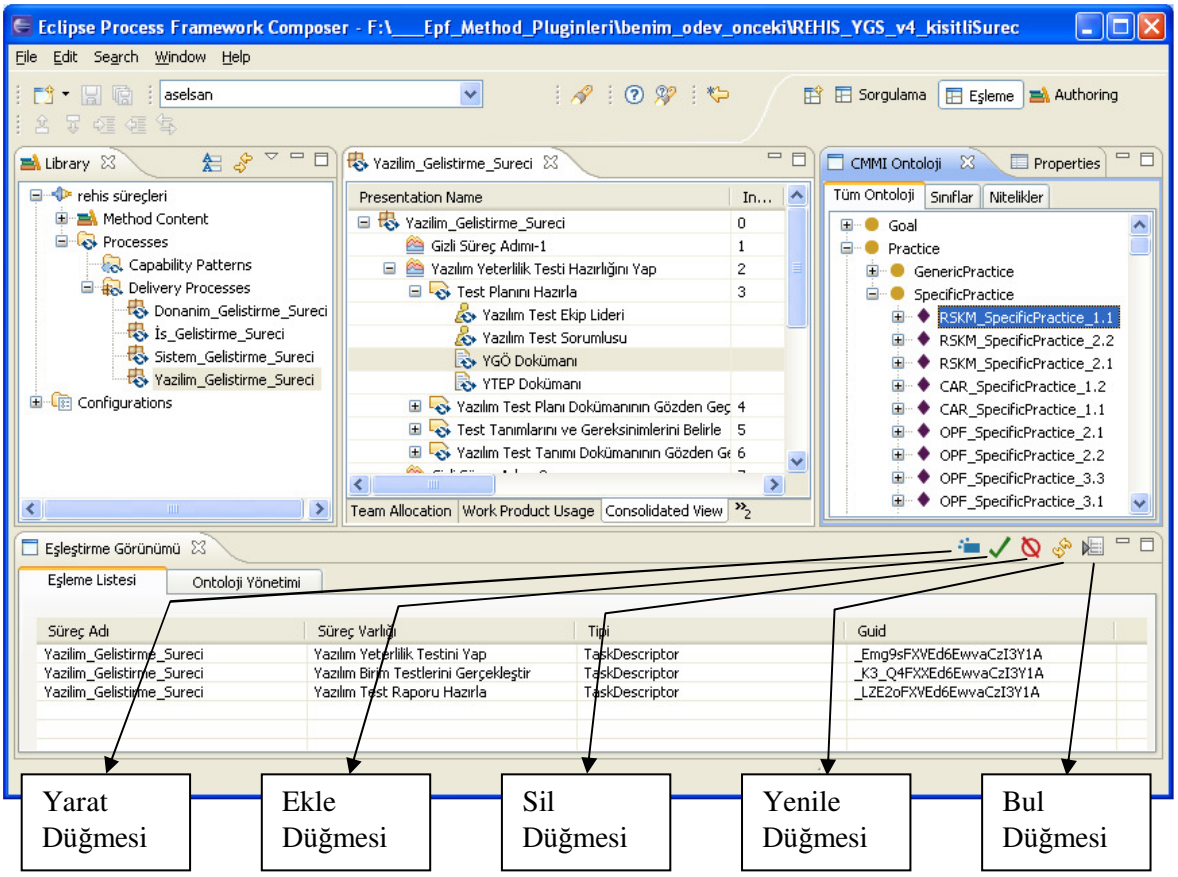


Şekil 26 - CMMI Ontoloji Ağacı Görünümü ve Sekmeler

4.2.4.2. Eşleme-Sorgulama Eklentisi

Bölüm 4.2.3.'te verilen; süreç ontolojisinin oluşturulması, eşleme kurulması, eşleme silinmesi, eşlemelerin listelenmesi, ontolojilerin yönetimi ve sorgulamalar ile ilgili gereksinimler (G3-G38) bu eklenti dahilinde gerçekleştirilmiştir.

Bu eklentideki işlevler; eşleme işlemleri, ontoloji yönetimi ve sorgulamalar olmak üzere üç alt bağlamda ele alınabilir. Eşleme işlemlerinin gerçekleştirildiği kullanıcı arayüzü görüntüsü ve düğmeler Şekil 27'de verilmiştir.



Şekil 27 - Eşleme İşlemleri İçin Kullanıcı Arayüzü Görünümü ve Düğmeler

Düğmelerin işlevleri şöyledir;

- Yarat düğmesi; Süreç ontolojisinin oluşturulmasını sağlar.
- Ekle düğmesi; Eşleme kurulmasını sağlar.
- Sil düğmesi; Eşlemenin silinmesini sağlar.
- Yenile düğmesi; Eşlemelerin listelenmesini sağlar.
- Bul düğmesi: Eşleme listesinden bir süreç varlığı seçilip Bul düğmesine basıldığında, seçilen süreç varlığının yer aldığı sürecin Süreç Editöründe açılmasını ve ilgili süreç varlığının seçili olarak işaretlenmesini sağlar.

Süreç Ontolojisinin Oluşturulması:

EPF'de süreçler *Capability Pattern* ve *Delivery Process* olarak adlandırılan nesne tipleri ile ifade edilmektedir. Capability Pattern, bir sürecin bir bölümünün tanımlandığı yapıdır. Bu yapı ile tekrar kullanılabilir süreç modülleri oluşturulabilmektedir. Delivery Process ise, tüm bir sürecin tanımlandığı nesne tipidir. Delivery Process tipinde bir nesne yaratılarak bir kurum süreci tanımlanabileceği gibi, bir projede uygulanan süreç de tanımlanabilmektedir. Bu tezde, Delivery Process olarak tanımlanan bir sürecin ontolojisinin oluşturulması hedeflenmiştir. Süreç Ontolojisi (SO), Süreç Yapısı Ontolojisi (SYO) adı verilen bir ontoloji temel alınarak oluşturulmaktadır. SYO ise Delivery Process göz önünde bulundurularak oluşturulmuş olan bir ontolojidir.

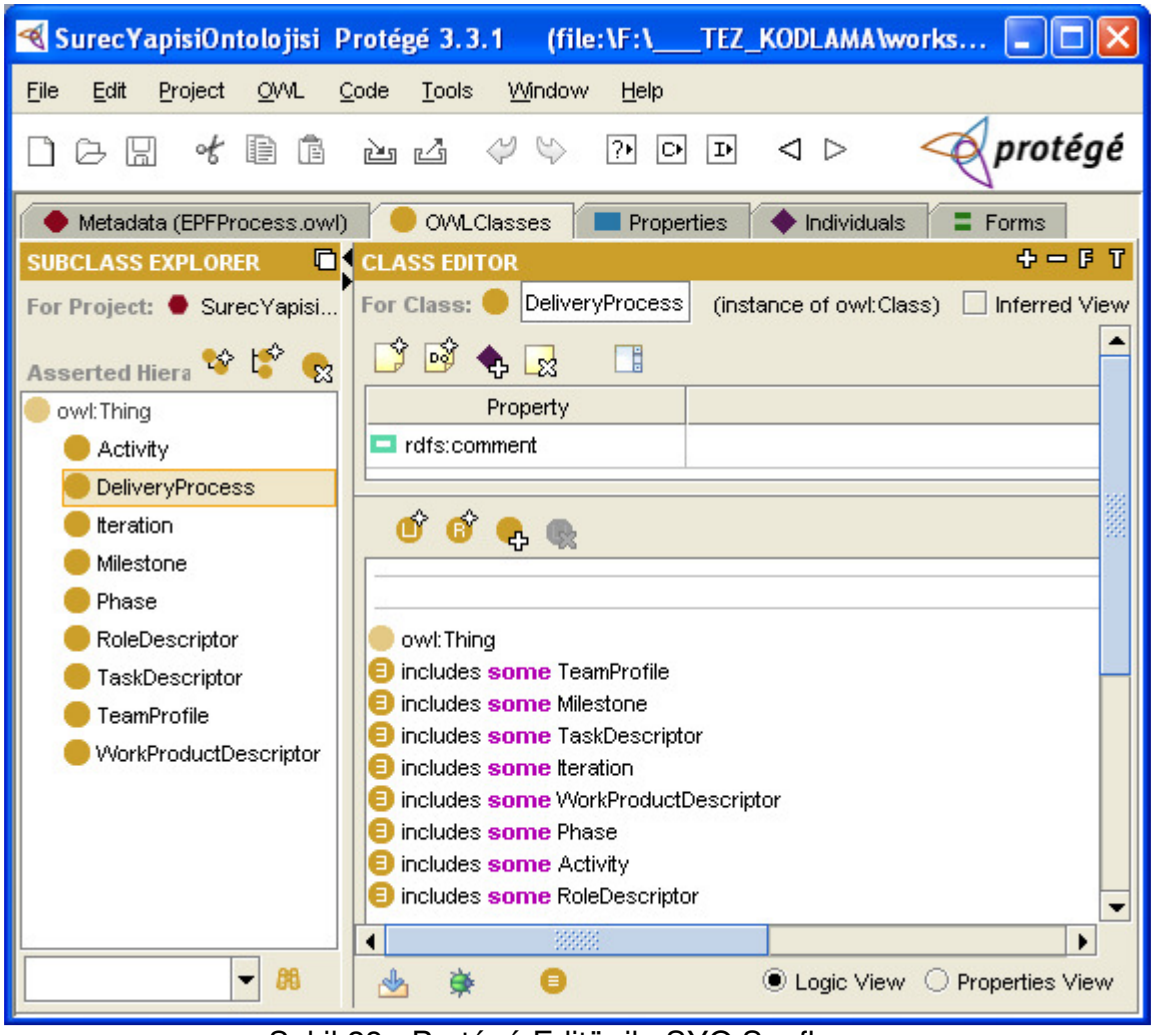
SYO oluşturulurken Delivery Process nesnesinin ağaç yapısı esas alınmıştır. Bu yapıda yer alan nesneler ve bunlar arasındaki ilişkiler şöyledir;

- EPF'de bir Delivery Process nesnesinin altında yer alabilen nesneler şunlardır: Phase, Iteration, Activity, Milestone, Task Descriptor, Work Product Descriptor, Role Descriptor, Team Profile.
- Delivery Process, Capability Pattern, Phase ve Iteration, Activity nesnesiden türetilmişlerdir. Ağaç yapısında, bu nesnelerin altında, Delivery Process nesnesinin altında yer alabilen nesneler yer alabilmektedir.

- Milestone, Task Descriptor, Work Product Descriptor, Role Descriptor nesnelerinin altında herhangi bir nesne tanımlanamamaktadır. Bunlar ağaç yapısında uç nesne olarak yaratılabilmektedirler.

- Team Profile altında ise sadece bir Team Profile nesnesi yaratılabilmektedir.

EPF Delivery Process ile tanımlanan bir sürecin varlıkları arasındaki ağaç yapısındaki ilişkinin ifade edilebilmesi için, SYO'da *includes* ilişkisi yaratılmış ve birbiri altında yer alan nesneler bu ilişki ile gösterilmiştir. Şekil 28'de, SYO'daki sınıflar ve Delivery Process sınıfının diğer sınıflarla olan *includes* ilişkisi görülmektedir.



Şekil 28 - Protégé Editör ile SYO Sınıfları

Kütüphane görünümünde yer alan bir *Delivery Process* olgusu seçildikten sonra *Yarat* düğmesine basıldığında, seçili sürecin ontolojisi yaratılır. Bir sürecin ontolojisi, Süreç Yapısı Ontolojisi (SYO) temel alınarak ve süreç varlıklarının

EPF'deki *evrensel biricik tanımlayıcı* (global unique identifier: guid) değerleri kullanılarak oluşturulmaktadır. Oluşturulan Süreç Ontolojisi (SO) süreç ile aynı adı taşıyan bir dosyada ve OWL biçiminde saklanmaktadır.

Eşleme Kurulması:

OCMQ-E, Süreç Ontolojisi ile CMMI Ontolojisi arasında eşleme kurulacak şekilde tasarlandığı için, bir süreç varlığı ile bir CMMI bileşeni arasında eşlemenin kurulabilmesi için, önce bu süreç varlığının ait olduğu sürecin ontolojisinin yaratılmış olması gerekmektedir. Tasarımın bu şekilde yapılmış olması, süreçlerin de ontolojileri vasıtasıyla sorgulanabilmesine olanak sağlamıştır. Ontolojisi oluşturulmuş bir süreçteki bir süreç varlığının seçilmesi ve CMMI ontoloji ağacından bir bileşen seçilmesi, ardından *Ekle* düğmesine basılması ile eşleme kurulur. Kurulan eşleme, Eşleme görünümünde yer alan Eşleme Listesinde bir satır olarak görünür ve eşleme bilgisi Eşleme Ontolojisi adı verilen bir ontolojide OWL biçiminde saklanır. Eşleme Ontolojisinde, Süreç Ontolojisindeki (SO) nesne ile CMMI Ontolojisindeki nesnenin ilişkilendirilme bilgileri yer almaktadır. Süreç varlıkları ile CMMI bileşenleri, birçoğa-birçok ilişki ile eşleştirildiğinden dolayı, bunlar arasında kurulan bir eşleştirme, birçoğa-birçok ilişkinin iki yönüne ait bilgiyi gösterecek şekilde Eşleme Ontolojisinde saklanmaktadır. Bir eşleştirme kurulurken, eşleştirilen CMMI bileşeni, ilişkilendirildiği süreç varlığının CMMI bileşenleri listesine eklenmekte ve eşleştirilen süreç varlığı da, ilişkilendirildiği CMMI bileşeninin süreç varlıkları listesine eklenmektedir. Bu ilişkilendirme bilgileri, eşleme sorgulamalarında kullanılmaktadır.

Eşleme Silinmesi:

Kurulu bir eşlemeyi silmek için, Eşleme Listesinden silinmesi istenen eşleme ile ilgili satır seçilir ve ardından *Sil* düğmesine basılır. Bir eşleme silinirken, eşleme ontolojisinde (EO) yer alan bu ilişkiye ait bilgiler, bir CMMI bileşeni ile ilişkilendirilen süreç varlıkları arasından ve bir süreç varlığı ile ilişkilendirilen CMMI bileşenleri arasından silinmektedir.

Eşlemelerin Listelenmesi:

Bu işlevde yalnızca Eşleme Ontolojisi (EO) kullanılmaktadır. Bir CMMI bileşeni ile hangi süreç varlıklarının eşleştirildiği bilgisi sorgulama yoluyla EO'dan alınmaktadır. Eşleştirilen süreç varlıkları, EO'da, evrensel biricik tanımlayıcı bilgileri ile saklanmaktadır. Bundan dolayı, bir CMMI bileşeni ile eşleştirilen süreç varlıkları listelenirken, öncelikle, sorgulama ile, ilgili süreç varlıklarının biricik tanımlayıcı bilgileri elde edilmekte, daha sonra ise bu bilgiler kullanılarak ilgili süreç varlıklarının *adı, türü* gibi detaylı bilgilere EPF'den ulaşılmaktadır. Son olarak, ulaşılan bu bilgiler, seçilen CMMI bileşeni ile eşleştirilen süreç varlıklarının listelenmesi bağlamında, bir liste halinde kullanıcı arayüzünde gösterilmektedir.

Ontolojilerin Yönetimi:

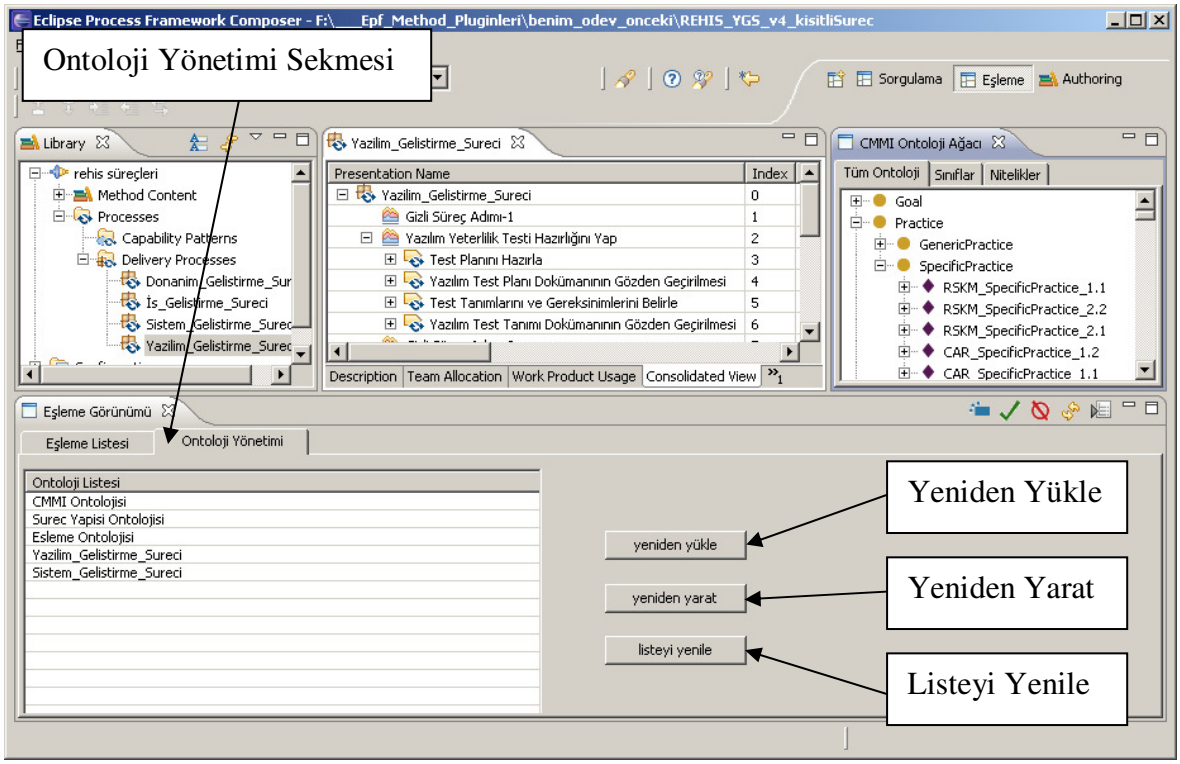
Ontoloji yönetimi ile ilgili gereksinimler, Şekil 29'da görülmekte olan Ontoloji Yönetimi sekmesi dahilinde gerçeklenmiştir. Bu sekmede, o an OCMQ-E'de yüklü olan ontolojilerin listesi görülmektedir. OCMQ-E açılırken, Şekil 30'da bir örneği görülmekte olan OntolojiListesi.txt dosyasından, yüklenmesi gereken ontolojilerin adları okunmakta ve bu ontolojiler dizinden okunarak yüklemektedir. OCMQ-E'nin kullanımı sırasında bir sürecin ontolojisi oluşturulduğunda, eğer OntolojiListesi.txt dosyasında adı olmayan bir süreç ise (ilk kez yaratılıyorsa), sürecin adı dosyaya eklenir. Böylece, aracın açılışında yüklenmesi gereken ontolojiler takip edilir.

Şekil 29'daki düğmelerin işlevleri özetle şöyledir;

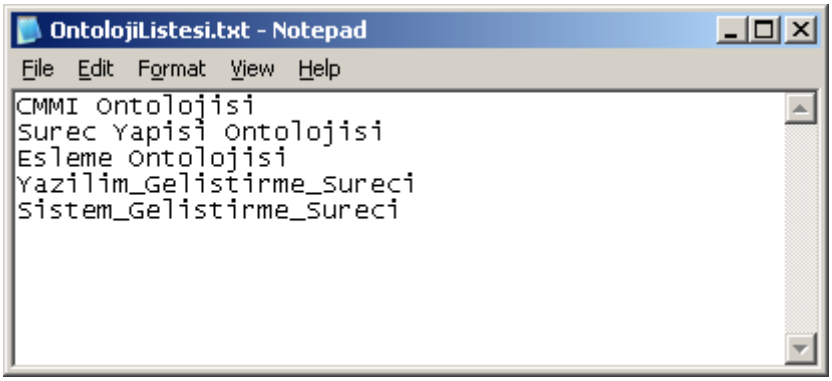
- Yeniden Yükle düğmesi; ontoloji listesinden seçilen herhangi bir ontolojinin yüklenmesini sağlar.

- Yeniden Yarat düğmesi; sadece Eşleme Ontolojisi seçili olduğunda aktif olan bir düğmedir ve bu ontolojinin en baştan herhangi bir eşleme içermeyecek şekilde yeniden oluşturulmasını sağlar.

- Listeyi Yenile düğmesi; OntolojiListesi.txt dosyasının yeniden okunarak ekrandaki Ontoloji Listesinin güncellenmesini sağlamaktadır.



Şekil 29 - Ontolojilerin Yönetimi



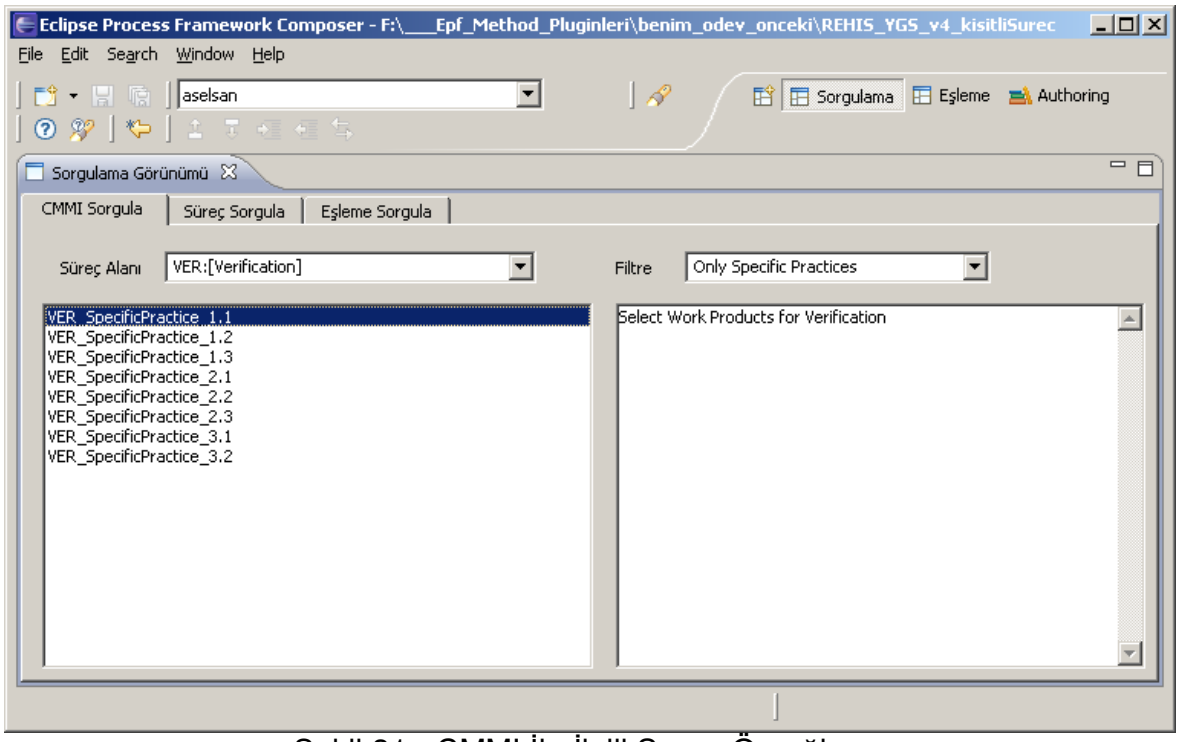
Şekil 30 - OntolojiListesi.txt Dosyasının Görünümü

Sorgulamalar:

Tez kapsamında, süreç ile ilgili farklı bilgiler içeren üç ontolojinin sorgulanması örneklenmek istenmiş ve bu amaçla bazı sorgulamalar geliştirilmiştir. Bu ontolojiler; CMMI Ontolojisi, Süreç Ontolojileri ve Eşleme Ontolojisidir.

- *CMMI Alan Bilgisi İle İlgili Sorgulamalar:* Seçilen bir süreç alanı için; tüm pratikler, özel pratikler, genel pratikler, tüm hedefler, özel hedefler ve genel hedefler listelenmektedir. Listelenen bir pratik ya da hedef tıklandığında ise sağ tarafta yer alan alanda seçilen pratik ya da hedef ile ilgili kısa açıklama görülebilmektedir. Yalnız CMMI Ontolojisinin kullanıldığı bu sorgu ile, CMMI alan bilgisinin sorgulanması örneklenmiştir.

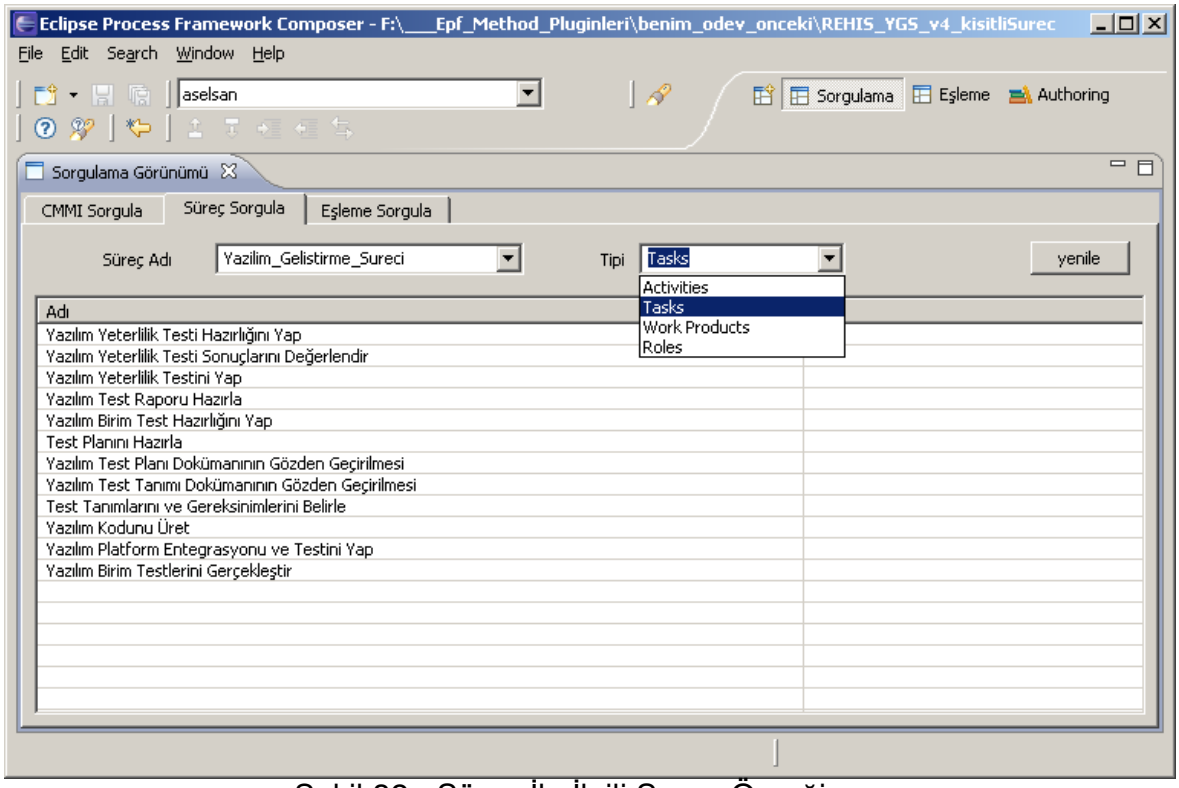
Şekil 31'de, CMMI alan bilgisi ile ilgili sorgulamaya bir örnek olarak, CMMI'daki Verification (Doğrulama) süreç alanının özel pratiklerinin listelendiği ve özel pratik 1.1 ile ilgili kısa açıklamanın yer aldığı ekran görüntüsü verilmiştir.



Şekil 31 - CMMI İle İlgili Sorgu Örneği

- *Süreç İle İlgili Sorgulamalar:* Ontolojisi oluşturulan süreçler arasından seçilen bir süreç için, bu süreçte yer alan aktiviteler, görevler, roller ve iş ürünleri listelenebilmektedir. Yalnız seçilen sürece ait Süreç Ontolojisinin (SO) kullanıldığı bu sorgu ile, süreç ile ilgili sorgulama örneklenmiştir.

Şekil 32'de, süreç ile ilgili sorgulamaya bir örnek olarak, Yazılım Geliştirme Sürecindeki görevlerin listelendiği ekran görüntüsü verilmiştir.



Şekil 32 - Süreç İle İlgili Sorgu Örneği

- *Eşlemeler İle İlgili Sorgulamalar:* CMMI ile süreçler arasındaki eşlemeler birçoğa-birçok eşlemelerdir. Bir CMMI bileşeni ile birçok süreç varlığı eşlenebildiği gibi, bir süreç varlığı da birçok CMMI bileşeni ile kurulan eşlemede yer alabilmektedir. Bundan dolayı, eşlemelerin sorgulanması birçoğa-birçok olan bu ilişkinin iki yönünü de ele alacak şekilde gerçekleştirilmiştir. Bunlar;

(1) Bir CMMI bileşeni ile eşleşen süreç varlıklarının listelenmesi

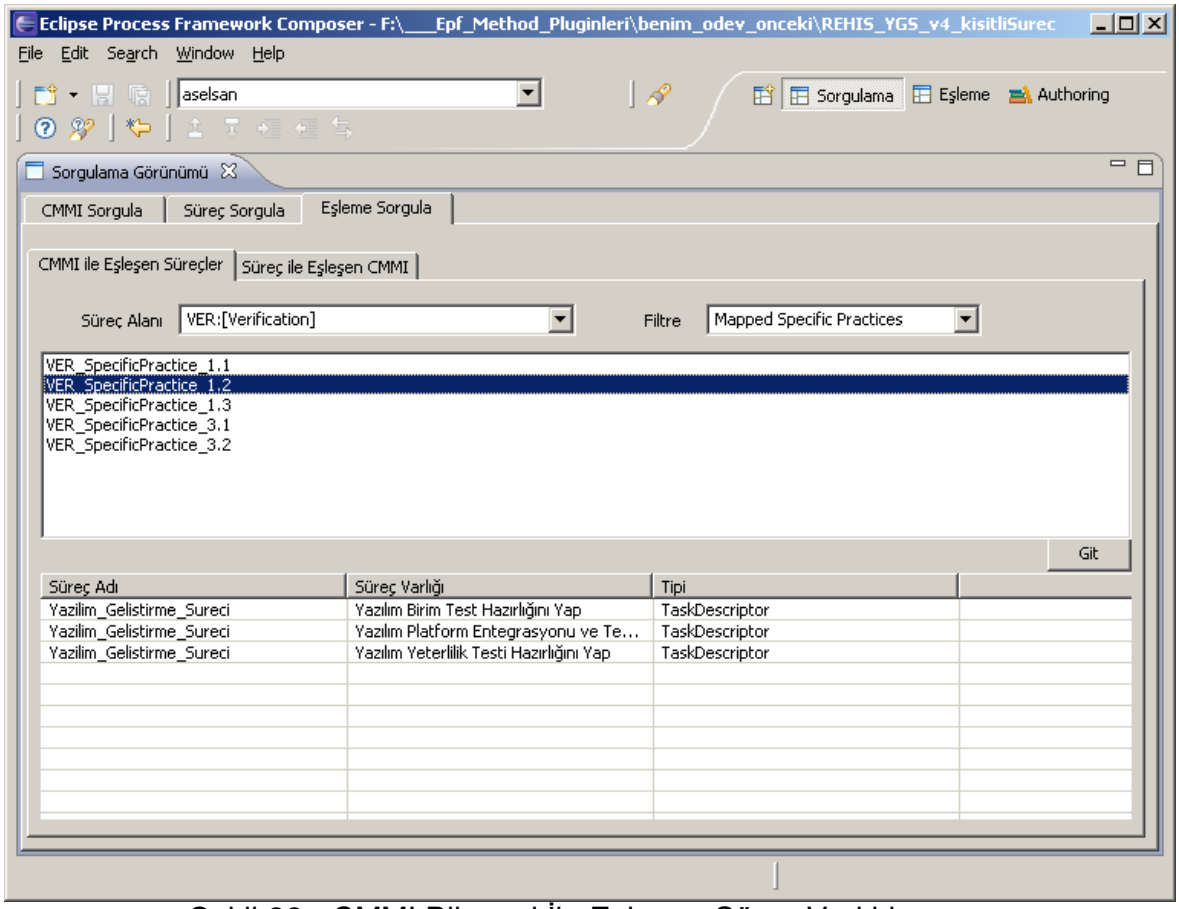
(2) Bir süreç varlığı ile eşleşen CMMI bileşenleri listelenmesi

(1) Bir CMMI bileşeni ile eşleşen süreç varlıklarının listelenmesi: Seçilen bir CMMI süreç alanı için; bir süreç varlığı ile eşleşen, hiçbir süreç varlığı ile eşleşmemiş olan ve tüm; özel ve genel pratikler, özel ve genel hedefler listelenebilmektedir. Eşleme Ontolojisinin (EO) kullanıldığı bu sorgular, Çizelge 6'daki satır ve sütûnların kesişimine karşılık gelmektedir.

Çizelge 6 - CMMI-Süreç Eşleme Sorgulamaları

	Herhangi Bir Süreç Varlığı İle Eşlenen	Hiçbir Süreç Varlığı İle Eşlenmeyen	Tüm
Özel Pratik	√	√	√
Genel Pratik	√	√	√
Özel Hedef	√	√	√
Genel Hedef	√	√	√

Şekil 33'te, bu tip sorgulamaya örnek olarak, CMMI'daki Verification (Doğrulama) süreç alanının, herhangi bir süreç varlığı ile eşleşen özel pratiklerinin listesi ve özel pratik 1.2 ile eşlenen süreç varlıklarının listesi görülmektedir.

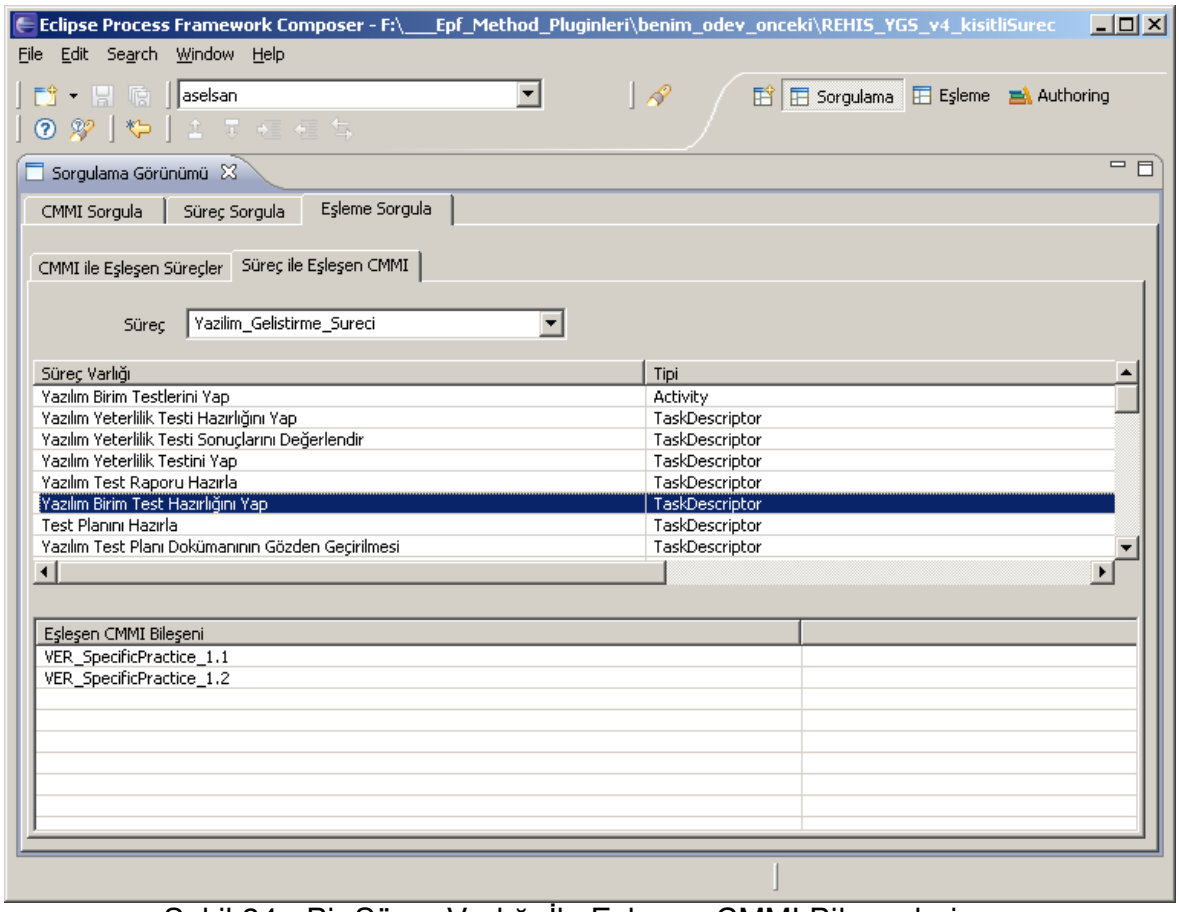


Şekil 33 - CMMI Bileşeni İle Eşlenen Süreç Varlıkları

(2) Bir süreç varlığı ile eşleşen CMMI bileşenlerinin listelenmesi: Ontolojisi oluşturulan süreçler arasından seçilen bir sürece ait süreç varlıkları listelenmekte,

bu varlıklar arasından biri seçildiğinde ise, seçilen varlık ile eşleştirilen CMMI bileşenleri listelenmektedir. Bu sorgulama Eşleme Ontolojisi (EO) üzerinde yapılmaktadır.

Şekil 34'te, bu tip sorgulamaya örnek olarak, Yazılım Geliştirme Sürecindeki Birim Test Hazırlığını Yap görevinin eşleştirildiği CMMI bileşenlerinin listesi görülmektedir.

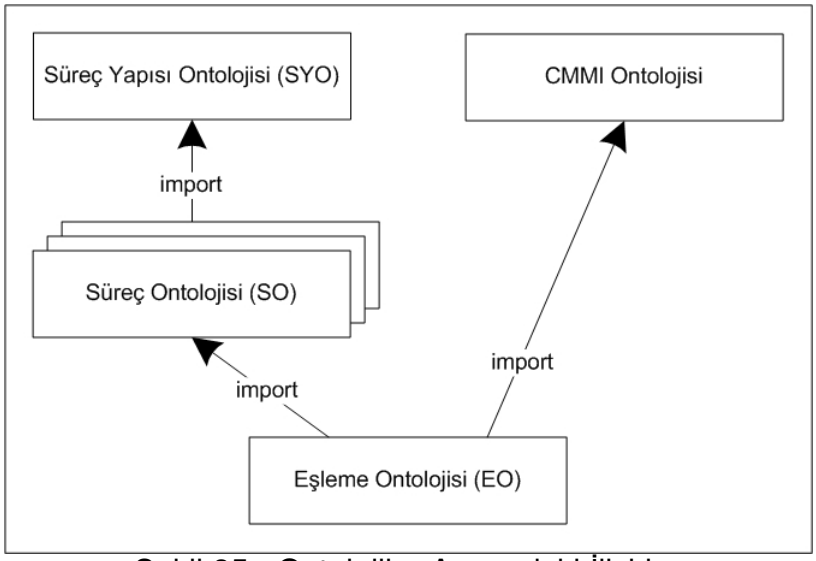


Şekil 34 - Bir Süreç Varlığı İle Eşlenen CMMI Bileşenleri

4.2.4.3. Ontolojiler ve Aralarındaki İlişkiler

Bu bölümde, tez kapsamında oluşturulan ontolojiler ve bunlar arasındaki ilişkiler açıklanmaktadır. Bu ontolojiler; CMMI Ontolojisi, Süreç Yapısı Ontolojisi (SYO), her süreç için ayrı oluşturulan Süreç Ontolojileri (SO'lar) ve Eşleme Ontolojisidir (EO). İlerleyen bölümde, SYO, SO ve EO ile ilgili bilgi verilmiştir. CMMI Ontolojisi ile ilgili detaylı açıklamlar Bölüm 4.1.'de verildiğinden dolayı burada tekrarlanmamıştır.

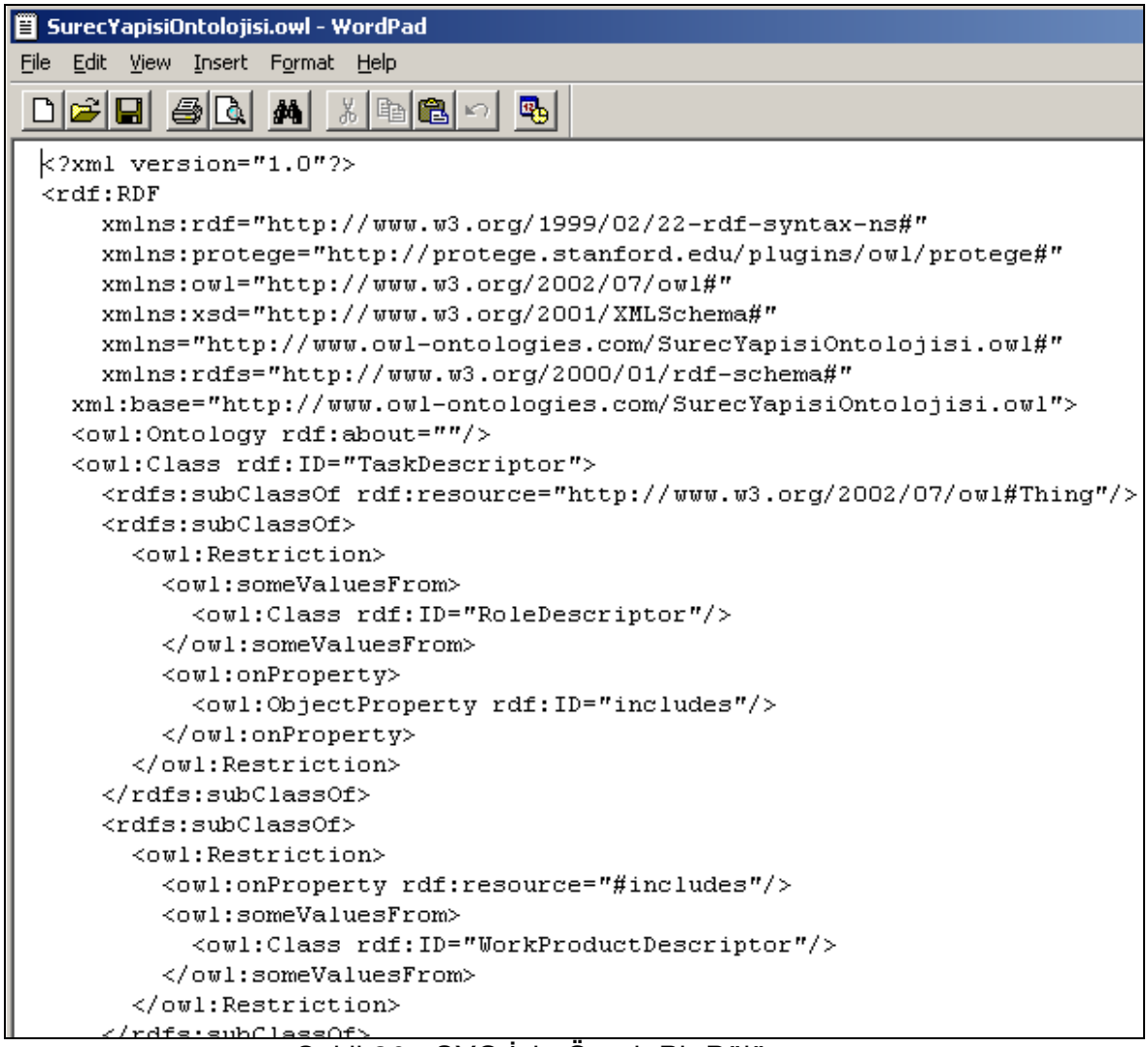
Ontolojiler arasındaki ilişkiler Şekil 35'te görülmektedir.



Şekil 35 - Ontolojiler Arasındaki İlişki

SYO: EPF'de bir süreç tanımlanırken kullanılan sınıfları ve bu sınıflar arasındaki ağaç yapısını ifade etmek üzere tanımlanan *includes* ilişkisini içeren bir ontolojidir. Bu ontolojide yer alan sınıflar şunlardır; Activity, Delivery Process, Iteration, Milestone, Phase, Role Descriptor, Task Descriptor, Team Profile ve Work Product Descriptor. Bu sınıflar ve ilişkileri, SPEM'deki *Yöntemle Birlikte Süreç* (Process With Methods) meta-model paketinde yer alan sınıflar ve bunların ilişkilerinin alt kümesine karşılık gelmektedir.

Şekil 36'da bir bölümü görülmekte olan SürecYapisiOntolojisi.owl dosyasının tamamı Ek-1'de verilmiştir.



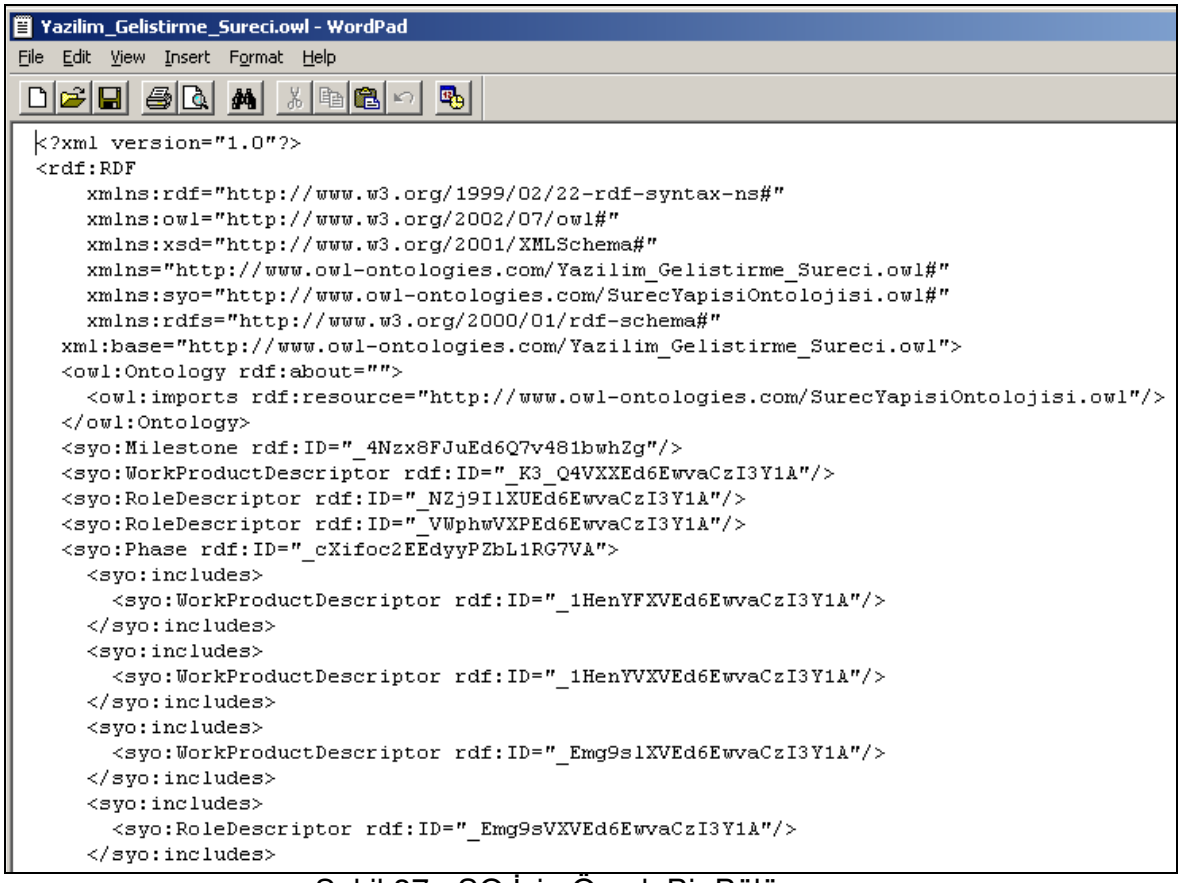
Şekil 36 - SYO İçin Örnek Bir Bölüm

Bu ontolojinin uygulama içinden değişmesi gerekliliği yoktur. Diğer bir deyişle durağan bir ontolojidir. Bundan dolayı, Protégé OWL Editör v3.3.1 ile bir kez yaratıldıktan sonra uygulama içinden salt okunur amaçla kullanılmıştır.

SO: EPF'de yazılan bir sürecin ontolojisidir. EPF'de birçok süreç yazılabileceğinden, her sürecin ontolojisi, süreç adı ile aynı adı taşıyan ayrı bir OWL dosyasında saklanmaktadır. Bir sürecin ontolojisi, SYO temel alınarak oluşturulur. SYO'da, süreç oluşturmak için gerekli nesneler ve bu nesneler arasındaki ilişkiler yer alırken, SO'da, EPF'de yazılmış bir sürece ait süreç varlıklarının evrensel biricik tanımlayıcıları bilgisi yer almaktadır. Bu olgular, SYO'daki sınıf tipleri ve ilişkileri esas alınarak yaratılmaktadır. Örneğin; süreçte 'Birim testleri yap' aktivitesinin olduğunu varsayalım. Bu aktivite için, SYO'daki Activity nesnesi esas alınarak bir nesne ve 'Birim testleri yap' aktivitesinin evrensel

biricik tanımlayıcısı da bu nesnenin bir olgusu olarak yaratılır ve bu olgu SO'da saklanır.

Şekil 37'de, Yazılım Geliştirme Sürecine ait olan Yazilim_Gelistirme_Sureci.owl dosyasının bir bölümü görülmektedir.



Şekil 37 - SO İçin Örnek Bir Bölüm

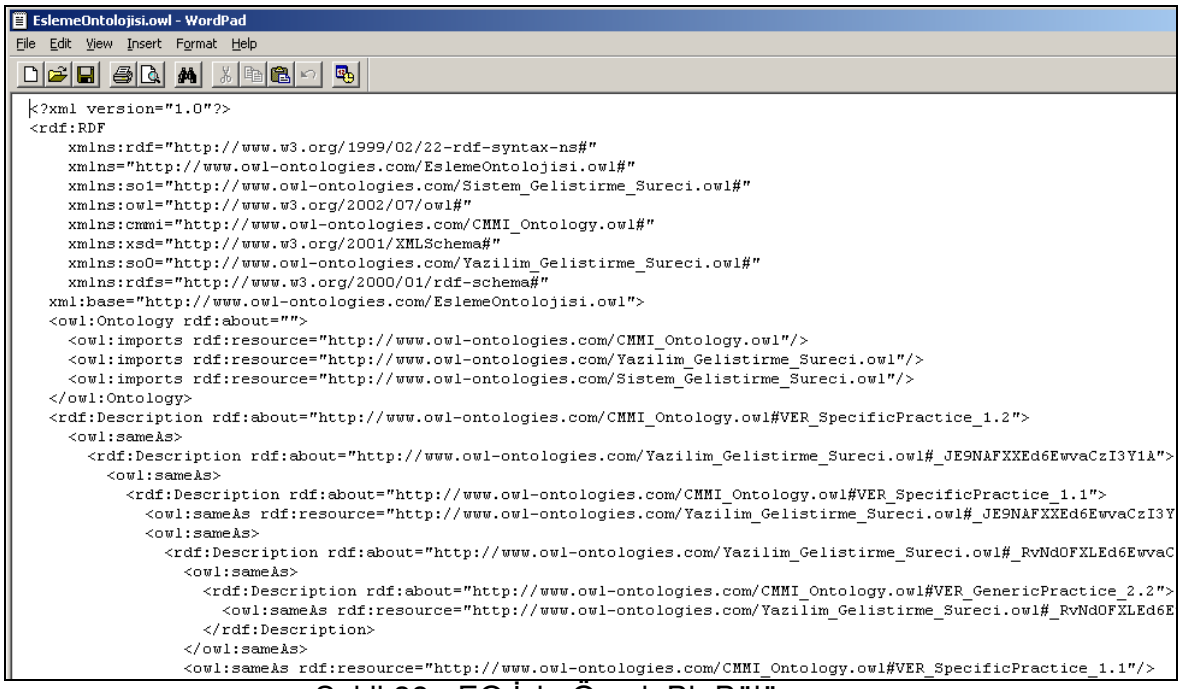
SO, uygulama içinden yaratılan, okunur-yazılır bir ontolojidir. Ontolojisi oluşturulan her süreç, süreç adı ile aynı adı taşıyan bir OWL dosyasında saklanır. Bunun sonucu olarak, sistemde aynı anda birden fazla SO olabilmektedir.

EO: CMMI ontolojisini ve SO'ları kullanan bu ontoloji, CMMI bileşenleri ile süreç varlıklarının eşleştirilmesine ilişkin bilgilerin yer aldığı ontolojidir. Bu eşleme birçoğa birçok (n-m) bir eşlemedir. Bir CMMI bileşeni birçok süreç varlığı ile eşleştirilebileceği gibi, bir süreç varlığı da birçok CMMI bileşeni ile kurulan eşlemede yer alabilmektedir.

Eşlemelerin CMMI pratikleri ile süreçteki görevler, aktiviteler veya iş ürünleri arasında yapılması beklenmekle birlikte, oluşturulan altyapı herhangi bir CMMI

bileşeni ile herhangi bir süreç varlığı arasında eşleme kurmaya olanak verecek şekilde esnek tasarlanmıştır.

Şekil 38'de, EslemeOntolojisi.owl dosyasının bir bölümü görülmektedir.



Şekil 38 - EO İçin Örnek Bir Bölüm

Eşlemelerin kurulması ve silinmesi gerekliliğinden dolayı, bu ontoloji okunur-yazılır bir ontolojidir. Sistemde bir tane EO ontolojisi olur ve tüm eşlemeler bu ontolojide yer alır.

4.2.5. OCMQ-E'nin Operasyonel Kullanımı

Bu bölümde, OCMQ-E'nin kullanımı ile ilgili özet bilgi verilmiştir.

Yeni Süreç Yazma ya da Mevcut Süreçte Değişiklik Yapma:

OCMQ-E kullanılarak yeni bir sürecin yazılması istendiğinde, Authoring perspektifi seçilerek işlemler yapılır. Authoring perspektifi orijinal EPF'de olan bir perspektiftir ve bu perspektifte gerçekleştirilen işlevler, OCMQ-E'ye EPF'den gelen yetenekler kullanılarak gerçekleştirilmektedir. Mevcut bir süreçte değişiklik yapılması istendiğinde ise, Authoring perspektifi ile birlikte Sorgulama perspektifinin de kullanılması faydalı olabilir. Süreçte değişiklik yapılırken, Sorgulama perspektifinde yer alan ve *bir süreç varlığı ile eşlenen CMMI bileşenlerinin listelendiği Eşleme sorgusu* ile, değiştirilecek ya da silinecek süreç varlığı ile eşlenen CMMI

bileşen(ler)inin olup olmadığı, varsa bu değişikliğin mevcut CMMI uyumunu nasıl etkileyeceği göz önünde bulundurularak işleme devam edilmesi faydalı olabilir.

CMMI ile Süreçlerin Eşlenmesi:

(1) CMMI ile eşlenecek süreç, ontolojisi oluşturulmuş bir süreç ise bu adım atlanarak sonraki adıma geçilir. Eğer değilse, eşlenecek süreç, Kütüphane görünümündeki Delivery Processes dalının altından bulunur ve seçili olarak işaretlenir. Daha sonra, Eşleme görünümündeki *Yarat* düğmesine basılarak seçili olarak işaretlenen sürecin ontolojisi oluşturulur. Bu sürecin daha önce ontolojisinin oluşturulup oluşturulmadığından veya yüklü olup olmadığından emin olmak istendiğinde, Eşleme görünümündeki Ontoloji Yönetimi sekmesinde yer alan Ontoloji Listesine bakılarak kontrol edilebilir.

(2) Eşleme görünümdeki Eşleme Listesinde, bir CMMI bileşeni ile eşlenmiş olan süreç varlıkları listelenmektedir. Bu liste elle yenilenen bir listedir. Bundan dolayı, her eşleme işleminden önce, eşlenecek CMMI bileşeni CMMI ağacında seçili olarak işaretlendikten sonra Eşleme görünümündeki *Yenile* düğmesine basılarak Eşleme Listesinin güncellenmesinde ve listenin kontrol edilmesinde fayda vardır.

(3) Eşlenecek süreç, Kütüphane görünümündeki Delivery Processes dalının altından bulunarak Süreç editöründe görüntülenir ve editörün altında yer alan sekmelerden biri olan *Consolidated View* açılır. Eşlenecek süreç varlığı, editörde yer alan listeden seçilir. Bu süreç varlığı ile eşlenecek olan CMMI bileşeni de CMMI Ontoloji Ağacı görünümünün ilk sekmesi olan *Tüm Ontoloji* sekmesinden seçilir. Genellikle CMMI pratikleri ile süreç varlıkları eşlenmektedir ve pratiklere Tüm Ontoloji sekmesindeki *Practices* dalının altından ulaşılabilir. Eşlenecek süreç varlığı ve CMMI bileşeni, belirtilen şekilde kullanıcı arayüzünde seçili olarak işaretlendikten sonra, Eşleme görünümündeki *Ekle* düğmesine basılarak eşleme kurulur. Eşleme kurulduğunda, ilgili süreç varlığının, Eşleme görünümündeki Eşleme Listesine yeni bir satır olarak eklendiği görülür.

(4) Bir eşlemenin yanlış ya da geçersiz olduğu düşünülerek silinmesi istendiğinde, ilgili eşleme satırı Eşleme Listesinde seçili olarak işaretlendikten sonra Eşleme görünümündeki *Sil* düğmesine basılarak silinir. Bu silme işlemi ile, ilgili eşleme,

hem kullanıcı arayüzündeki Eşleme Listesinden hem de Eşleme Ontolojisinden silinir. Bir eşleme silinmeden önce Eşleme Listesinin yenilenmesinde fayda olabilir.

(5) Kullanım sırasında, Eşleme Listesinde yer alan bir süreç varlığı ile ilgili daha detaylı bilgi edinmek istendiğinde, listeden ilgili süreç varlığı seçildikten sonra Eşleme görünümündeki *Git* düğmesine basılır. Bu düğmeye basılması, ilgili süreç varlığının ait olduğu sürecin Süreç editöründe görüntülenmesini ve bu süreç varlığının editörde seçili olarak işaretlenmesini sağlar.

Sorgulamalar:

Sorgulamalar ihtiyaç duyuldukça kullanılabilirler. Sorgulamalara genellikle hangi durumlarda ihtiyaç duyulacağı ile ilgili olarak şunlar söylenebilir;

(1) CMMI sorgulaması, bir süreç alanının sahip olduğu pratiklerin ve hedeflerin bir liste halinde görülmesi ya da seçilen bir pratiğin ya da hedefin ne olduğu ile ilgili çok özet bilgi edinmek (daha çok hatırlama) amacıyla kullanılabilir. Genellikle, bir eşleme yapılacağı zaman, eşlenecek bir pratik ya da hedef ile ilgili, bunun hangi pratik ya da hedef olduğundan emin olmak için kullanılabilir. Bu sorgu, CMMI pratikleri ve hedefleri ile ilgili detaylı bilgi edinmeyi sağlamamaktadır.

(2) Süreç sorgulaması, süreçte yer alan; aktiviteler, görevler, iş ürünleri ve roller bir liste halinde görülmek istendiğinde kullanılabilir.

(3) Eşleme sorgulamaları:

- *CMMI ile eşlenen süreç varlıklarının listelenmesini sağlayan sorgulama*; süreç değerlendirme etkinliğindeki verilerin toplanması aşamasını desteklemek için ve süreç iyileştirme etkinliğinde, iyileştirme yapılabilecek noktaları belirlemede yol gösterici olarak kullanılabilir.

- *Bir süreç varlığı ile eşlenen CMMI bileşenlerinin listelenmesi ile ilgili sorgu*; bir süreçte değişiklik yaparken, bu değişikliğin süreç ile CMMI arasındaki uyumu nasıl etkileyeceğini görebilmek amacıyla kullanılabilir.

5. ÖRNEK ÇALIŞMA: ASELSAN A.Ş. REHİS Grubu Yazılım Geliştirme Süreci

Türkiye'nin en büyük savunma sanayi kuruluşlarından biri olan Aselsan A.Ş., bir Türk Silahlı Kuvvetlerini Güçlendirme Vakfı kuruluşudur. Aselsan'da, yazılım ve sistem geliştirme, 4 Grup Başkanlığı çatısı altında gerçekleştirilmektedir ve bugünlerde bu gruplarda CMMI-Dev v1.2 için 3.Seviye Olgunluk Belgesi alma çalışmaları yürütülmektedir. Gruplardan biri olan REHİS (Radar, Elektronik Harp ve İstihbarat Sistemleri) Grup Başkanlığı'nda, farklı disiplinlerden çalışanların bir araya gelmesi ile oluşturulmuş bir ekip tarafından, MS Excel tabloları kullanılarak, her CMMI pratiğinin karşısına, bu pratiği karşılayacak süreç adım(lar)ı ve ilgili iş ürünü/ürünleri yazılarak, CMMI değerlendirmesi için eksiklerin belirlenmesi ve değerlendirme etkinliğinin bir aşaması olan verilerin toplanması aşamasına hazırlık yapılmaktadır. MS Excel uygulamasının, hem kurumlar hem de tetkikçiler tarafından bu amaçla yaygın olarak kullanıldığı gözlemlenmiştir.

Excel tabloları bilginin tablo yapısı ile düzenlenmesine olanak vermekle birlikte, bu yapı bilgi üzerinde istenilen biçim ve koşullarda kolay sorgulama yapılabilmesine imkân vermemektedir. Aranan bilgiye, uygulamadaki arama (search) yeteneği kullanılarak ulaşılmak istendiğinde ise, bunun zor ve hataya açık bir ortam olacağı düşünülmektedir. Örneğin, Excel kullanılan bir ortamda, bir süreç varlığının hangi CMMI pratikleri ile eşlendiğinin öğrenilmek istendiğini varsayalım. Eşleme ile ilgili bilgi Excel tablolarına serbest metin biçiminde yazıldığında ve her CMMI pratiğinin karşısına bununla eşleşen süreç varlıklarının adları yazılarak eşlemeler kurulduğunda, bir süreç varlığının adı yanlış yazılırsa ya da aynı süreç varlığı farklı pratiklerin karşısında farklı kelimelerle ifade edilirse, bu süreç varlığının hangi CMMI pratikleri ile eşlendiği sorusuna arama yeteneği kullanılarak cevap arandığında ulaşılan bilgi doğru olmayacaktır. Tablo yapısı, pratiklerden süreç varlıklarına ulaşmada kolaylık sağlasa da, tersi bir sorunun cevabı için kolaylık sağlamamaktadır. Yalnız bu örnekteki durum değil, kurum için çok değerli olan süreç bilgisinin birçok farklı açıdan ele alınması ihtiyacı (örneğin; belli bir kurum sürecinin hangi CMMI bileşenleri ile eşlendiğinin görülmek istenmesi gibi) oluşabilir. Bundan dolayı bu bilgilerin sorgulanabilmesi önemlidir.

Ayrıca, büyük emekler verilerek toplanan bu değerli bilgi güncellenerek idame ettirilirse, sadece planlanan süreç değerlendirme etkinliğinde değil, süreç iyileştirme faaliyetlerinde ve sonraki süreç değerlendirme etkinliklerinde de

kullanılabilir. Bu durumda, sonraki süreç değerlendirme etkinlikleri için aynı emeğin ve zamanın tekrar harcanmasına gerek duyulmayabilir.

Excel tablolarına yazılan eşleme bilgileri OCMQ-E'ye aktarıldığında, eşleme bilgisi ontoloji biçiminde saklandığından dolayı, eşlemeler üzerlerinde sorgulamalar yapılabilmektedir. Böylece, Excel'deki arama işlevi kullanılarak değil, sorgulamalar yoluyla istenilen bilgilere ulaşılabilmektedir. Excel'deki bilgilerin toplanmasının amacı CMMI ile ilgili eksiklerin tespit edilmesi ve hedeflenen süreç değerlendirmeye hazırlık iken, bilgiler araca aktarıldığı taktirde, yapılacak süreç iyileştirmelerinde yol gösterici olarak da kullanılabilecektir. Hem süreçler hem de süreçlerin CMMI ile uyumu aynı ortamda bulunduğundan ve bir süreç varlığı ile eşlenen CMMI bileşenleri kolayca görülebildiğinden dolayı, süreçte güncelleme yapılıken, güncellenecek süreç varlığı ile eşlenen CMMI bileşenleri listelenerek CMMI uyumunun bundan nasıl etkileneceği görülebilmektedir. Ayrıca, hiçbir süreç varlığı ile eşlenmeyen CMMI pratiklerinin kolayca görülebilir olması, eksiklerin tespit edilmesini desteklemekte ve süreç iyileştirme için yönlendirici olabilmektedir.

5.1. Yazılım Geliştirme Süreci (YGS)

OCMQ-E'nin kullanımının örneklenmesi için REHİS Grubu'ndaki süreçlerden biri olan Yazılım Geliştirme Süreci (YGS) [20], bu çalışma için bir örnek süreç tanımı olarak ele alınmıştır. YGS'de yer alan bazı süreç adımları Çizelge 7'de verilmiştir. Kurumsal bilginin gizliliği çerçevesinde, süreç bilgisinin tamamının raporda yer almaması gereğinden dolayı sürecin yalnızca kısıtlı bir kısmı raporda verilebilmiştir.

Çizelge 7 - YGS'nin Bazı Adımları

1	…	
2	Yazılım Yeterlilik Testi Hazırlığını Yap	
	2.1	Test Planını Hazırla
	2.2	Yazılım Test Planı Dokümanının Gözden Geçirilmesi
	2.3	Test Tanımlarını ve Gereksinimlerini Belirle
	2.4	Yazılım Test Tanımları Dokümanının Gözden Geçirilmesi

3	…		
4	Yazılımı Kodla ve Testini Yap		
	4.1	Yazılım Kodunu Üret	
	4.2	Yazılım Birim Testlerini Yap	
		4.2.1	Yazılım Birim Testi Hazırlığını Yap
		4.2.2	Yazılım Platform Entegrasyonu ve Testini Yap
		4.2.3	Yazılım Birim Testlerini Gerçekleştir
	4.3	Yazılım Kodunun Tamamlanması	
5	Yazılım Yeterliliğini Kontrol Et		
	5.1	Yazılım Yeterlilik Test Hazırlığını Yap	
	5.2	Yazılım Yeterlilik Testini Yap	
	5.3	Yazılım Test Raporu Hazırla	
	5.4	Yazılım Yeterlilik Test Sonuçlarını Değerlendir	
…	…		

5.2. YGS'nin OCMQ-E'de İfade Edilmesi

YGS OCMQ-E'nin orijinal EPF yetenekleri kullanılarak ifade edilmiştir.

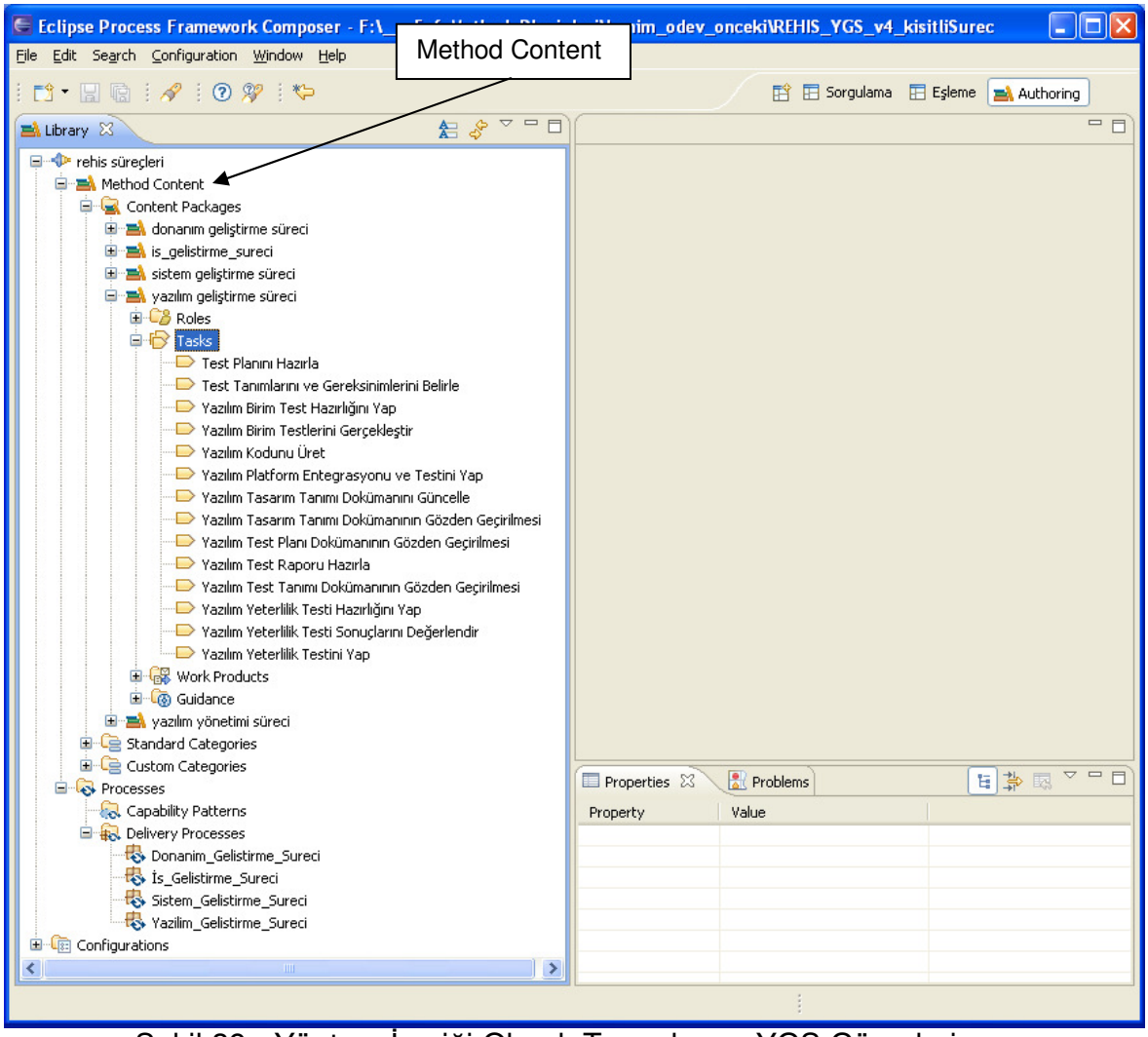
Bir süreç EPF'de farklı şekillerde ifade edilebilmektedir. Bu farklılık hem süreç varlıklarının türü ile hem de ele alındığı yer ile ilgili olabilir. Örneğin süreçteki iş adımının bir aktivite olarak ya da bir görev olarak tanımlaması ile ilgili çok kesin tanımlar yoktur. Süreç varlığının yeri ile ilgili olarak da, süreç varlıkları bir sürecin içinde tanımlanabileceği gibi süreçten bağımsız olarak bir yöntem içeriği olarak da tanımlanabilmektedir. Bir modül olarak tanımlanarak birden fazla süreçte kullanılmak istenen ya da süreç varlıkları kütüphanesinde bulunması istenen varlıklar Yöntem İçeriği nesnesi altında tanımlanır. Sadece belli bir sürece özgü olan nesneler ise, kütüphaneye konulmadan doğrudan o sürecin bir varlığı olarak tanımlanabilmektedir.

Çizelge 7'de hiyerarşik yapıda görülen YGS süreç adımlarından, hiyerarşinin en alt seviyesinde yer alan süreç adımlarının, orijinal EPF'de yer alan ve Şekil 39'da

görülmekte olan *Method Content* (Yöntem İçeriği) nesnesi altında *Task Description* (Görev Tanımı) nesnesi olarak tanımlanması uygun görülmüştür. Yöntem İçeriği, SPEM'deki Yöntem İçeriği paketinin gerçekleştirimidir ve tez kapsamında bu paketin doğrudan ontolojisi oluşturulmamaktadır. Ancak, süreçlerin ontolojileri oluşturulurken temel alınan SYO, Yöntemle Birlikte Süreç paketi esas alınarak oluşturulduğundan ve bu paket de Yöntem İçeriği paketi ile Süreç Yapısı paketini birleştiren bir paket olduğundan dolayı, SYO'da Yöntem İçeriği paketinden de nesneler yer almaktadır.

Görev türündeki nesneler, ağaç yapısında ifade edilen bir sürece eklendiklerinde, altlarında başka bir nesnenin tanımlanmasına izin vermeyen, diğer bir deyişle ancak bir uç nesne olabilen nesnelerdir. Ayrıca görevler; girdi iş ürünü, çıktı iş ürünü ve roller ile ilişkilendirilebilmektedirler. Bundan dolayı, YGS'de yer alan dokümanlar Yöntem İçeriğinde *Work Product Description* (İş Ürünü Tanımı) ve roller de *Role Description* (Rol Tanımı) olarak tanımlandıktan sonra, YGS'de belirtildiği şekilde, görevler ile ilişkilendirilmişlerdir. Böylece, yöntem içeriği ile ilgili tanımlamalar yapılarak süreç tanımlanmadan önceki hazırlıklar tamamlanmıştır.

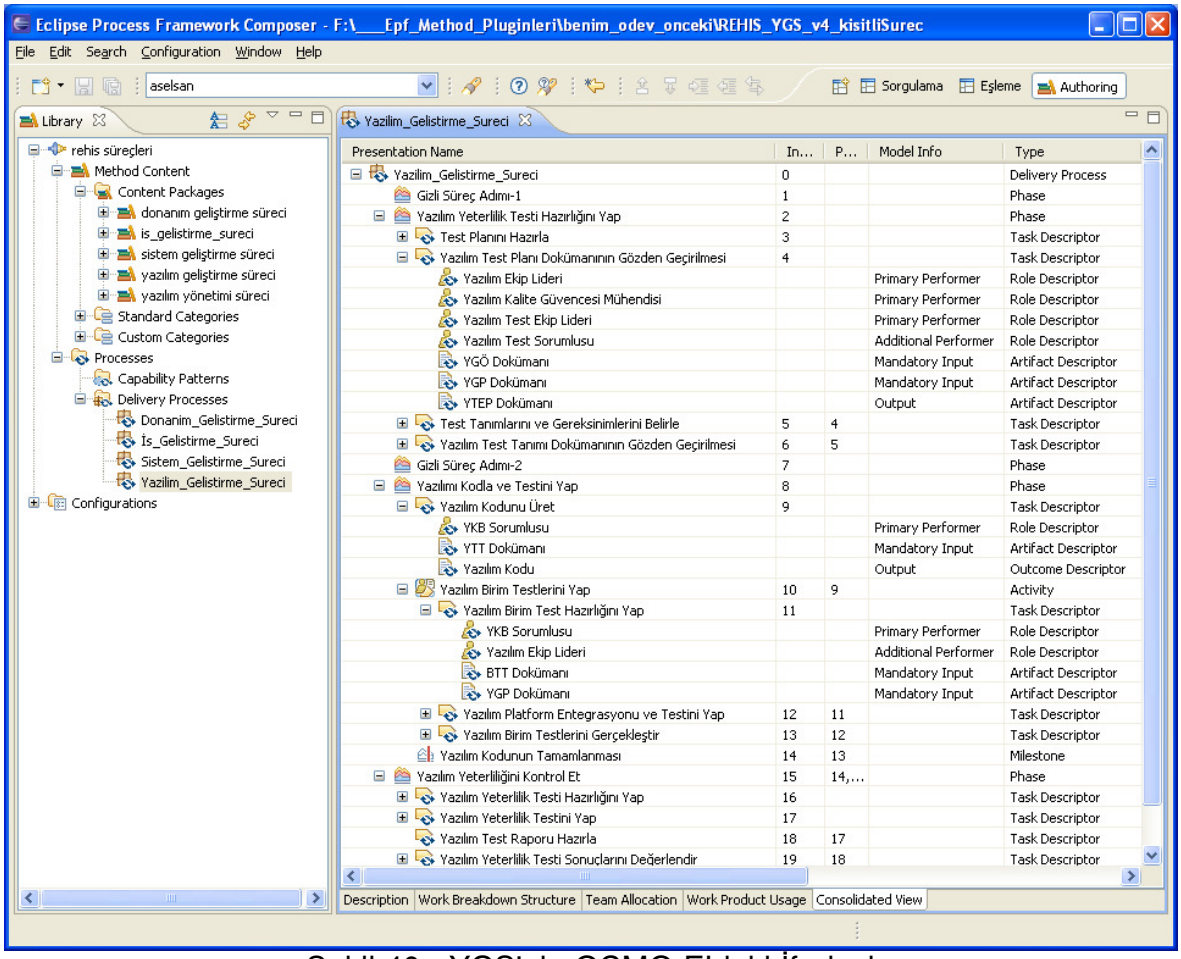
Şekil 39'da Yöntem İçeriği olarak tanımlanan YGS görevleri görülmektedir. Yöntem İçeriği olarak tanımlanmış olan roller ve iş ürünleri ile ilgili açıklama ve şekil, kurumsal bilginin gizliliği çerçevesinde verilmemiştir.



Şekil 39 - Yöntem İçeriği Olarak Tanımlanan YGS Görevleri

Sürecin tanımlanması aşamasında, EPF'deki Processes (Süreçler) dizini altında yer alan Delivery Processes (Dağıtım Süreçleri) dizini altında 'Yazılım Geliştirme Süreci' adı ile bir süreç yaratılmıştır. Çizelge 7'de hiyerarşinin en üst seviyesinde yer alan (1, 2, 3 vb. numaralı) süreç adımları *Phase* (Faz) türünde nesneler ile ifade edilmiştir. YGS şelale yaşam döngüsü modeline göre tanımlanmış bir süreç olduğundan dolayı bu ana adımların faz olarak ifade edilmesi uygun bulunmuştur. Çizelge 7'deki diğer süreç adımları ise (en üst ve en alt seviye olmayan adımlar), Activity (Aktivite) türündeki nesneler ile ifade edilmiştir. Aktivite nesnesi alt nesneler içerebilen bir nesne türüdür ve bu özelliği ile sürecin ağaç yapısındaki kırınımının ifade edilebilmesine olanak vermiştir.

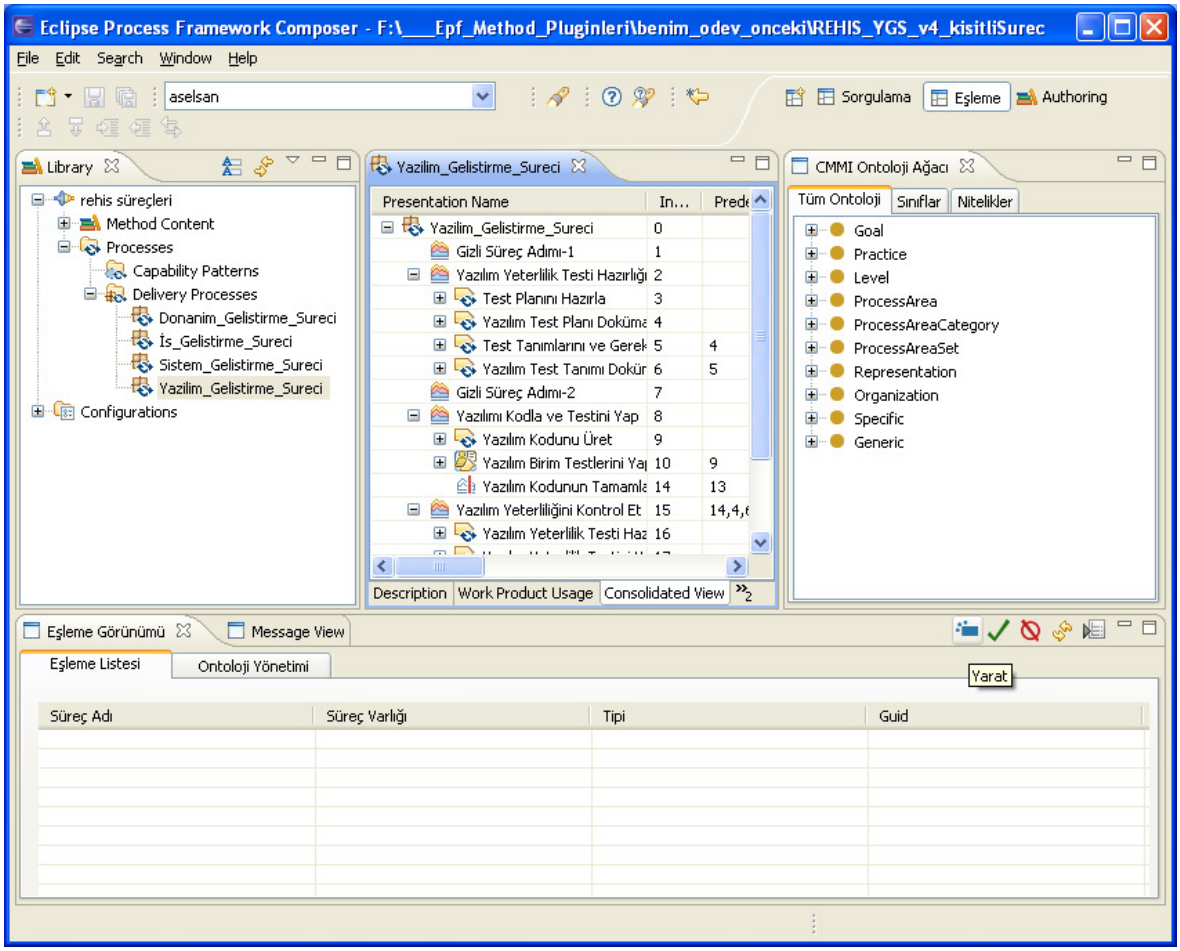
Şekil 40'ta OCMQ-E'de bir Dağıtım Süreci olarak ifade edilen YGS görülmektedir.



Şekil 40 - YGS'nin OCMQ-E'deki İfadesi

5.3. YGS'nin Ontolojisinin Yaratılması

YGS, OCMQ-E'nin orijinal EPF yetenekleri kullanılarak Bölüm 5.2.'de anlatıldığı şekilde ifade edildikten sonra, ontolojisi oluşturulmuştur. YGS'nin ontolojisi, OCMQ-E'deki Eşleme perspektifi ekranda görünürken ve Kütüphane (Library) görünümünde Yazılım Geliştirme Süreci seçili iken Eşleme görünümündeki *Yarat* düğmesine basılarak oluşturulmuştur. Buna ilişkin ekran görüntüsü Şekil 41'de, oluşan Yazilim_Gelistirme_Sureci.owl dosyasının içeriği ise Ek-2'de verilmiştir.



Şekil 41 - YGS'nin Ontolojisinin Oluşturulması

5.4. YGS ile CMMI Pratiklerinin Eşleştirilmesi

Çizelge 7'de yer alan YGS süreç adımlarının, CMMI-Dev 3. Olgunluk Seviyesinde yer alan süreç alanlarının pratikleri ile eşlenmesine ait bilgi Çizelge 8'de verilmiştir.

Çizelge 8 - YGS ile CMMI Pratiklerinin Eşlenmesi

YGS Süreç Adımı			*CMMI Süreç Alanı*	*CMMI Pratiği*
1	…		…	…
2	Yazılım Yeterlilik Testi Hazırlığını Yap			
	2.1	Test Planını Hazırla	VER	SP 1.1 Select Work Products for Verification
			VER	SP 1.3 Establish Verification Procedures and Criteria

			VER	GP 2.2
	2.2	Yazılım Test Planı Dokümanının Gözden Geçirilmesi	VER	GP 2.7
	2.3	Test Tanımlarını ve Gereksinimlerini Belirle	VER	SP 1.3 Establish Verification Procedures and Criteria
	2.4	Yazılım Test Tanımları Dokümanının Gözden Geçirilmesi	VER	GP 2.7
3	…	…	…	...
4	Yazılımı Kodla ve Testini Yap			
	4.1	Yazılım Kodunu Üret	TS	SP 3.1 Implement the Design
	4.2	Yazılım Birim Testlerini Yap		
	4.2.1	Yazılım Birim Testi Hazırlığını Yap	VER	SP 1.1 Select Work Products for Verification
			VER	SP 1.2 Establish the Verification Environment
	4.2.2	Yazılım Platform Entegrasyonu ve Testini Yap	VER	SP 1.2 Establish the Verification Environment
	4.2.3	Yazılım Birim Testlerini Gerçekleştir	VER	SP 3.1 Perform Verification
	4.3	Yazılım Kodunun Tamamlanması		
5	Yazılım Yeterliliğini Kontrol Et			
	5.1	Yazılım Yeterlilik Test Hazırlığını Yap	VER	SP 1.2 Establish the Verification Environment
	5.2	Yazılım Yeterlilik Testini Yap	VER	SP 3.1 Perform Verification
	5.3	Yazılım Test Raporu Hazırla	VER	SP 3.1 Perform Verification
	5.4	Yazılım Yeterlilik Test Sonuçlarını Değerlendir	PI	SP 3.1 Confirm Readiness of Product Components for Integration
			VER	SP 3.2 Analyze Verification Results
…	…	…	…	…

Çizelge 8'de yer alan eşleme bilgileri, OCMQ-E'deki Eşleme perspektifi kullanılarak araç ortamına aktarılmıştır. Eşlemeler, OCMQ-E'deki Süreç editöründen ilgili süreç varlığı ve CMMI Ontoloji Ağacı görünümünden de ilgili CMMI pratiği seçildikten sonra, Eşleme görünümünde yer alan *Ekle* düğmesine basılarak kurulmuştur. Kurulan eşlemelerin saklandığı EslemeOntolojisi.owl dosyası Ek-3'de verilmiştir.

YGS ile CMMI pratikleri eşleştirilirken, eşleştirmenin YGS tarafı için, süreçte yer alan fazlar, aktiviteler ve görevler kullanılmıştır. Ancak OCMQ-E, istendiğinde iş ürünlerinin de CMMI pratikleri ile eşleştirilebilmesine olanak veren yeteneğe sahiptir.

5.5. Sorgulamalar

Daha önceki bölümlerde de belirtildiği gibi, OCMQ-E'de üç tip sorgulama yapılabilmektedir. Bunlar;

- CMMI ile ilgili sorgulamalar,

- Ontolojisi oluşturulan bir süreç ile ilgili sorgulamalar,

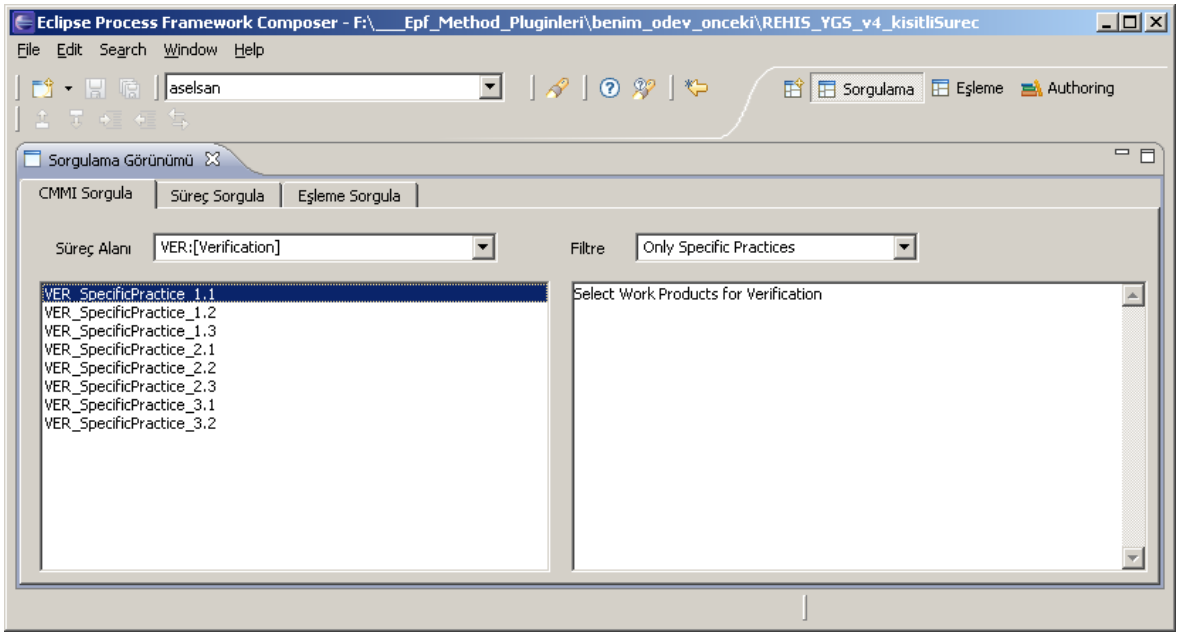
- Süreç – CMMI eşlemeleri ile ilgili sorgulamalardır.

İlerleyen kısımda süreç sorguları bağlamında YGS ile ilgili sorgulara ve Süreç-CMMI eşlemeleri ile ilgili sorgular bağlamında ise YGS-CMMI eşlemeleri ile ilgili sorgulara değinilmiştir.

5.5.1. CMMI Sorguları

CMMI ile ilgili sorgular, bir süreç alanında yer alan hedeflerin ve pratiklerin listelenmesini, seçilen bir hedef ya da pratik için de kısa bir açıklamanın ekranda görüntülenmesini sağlamaktadır. Bu sorgulama ile, eşlemeler kurulurken, ihtiyaç duyuldukça, pratikler ve hedefler ile ilgili kısa açıklamalara bakılmış, her defasında CMMI referans modeline bakmak zorunda kalınmamıştır. Açıklamalar CMMI Ontolojisinden çekilmektedir ve ontolojiye girilen bilgi detaylandırıldığında, ekranda görüntülenen bilgi de daha detaylı olacaktır.

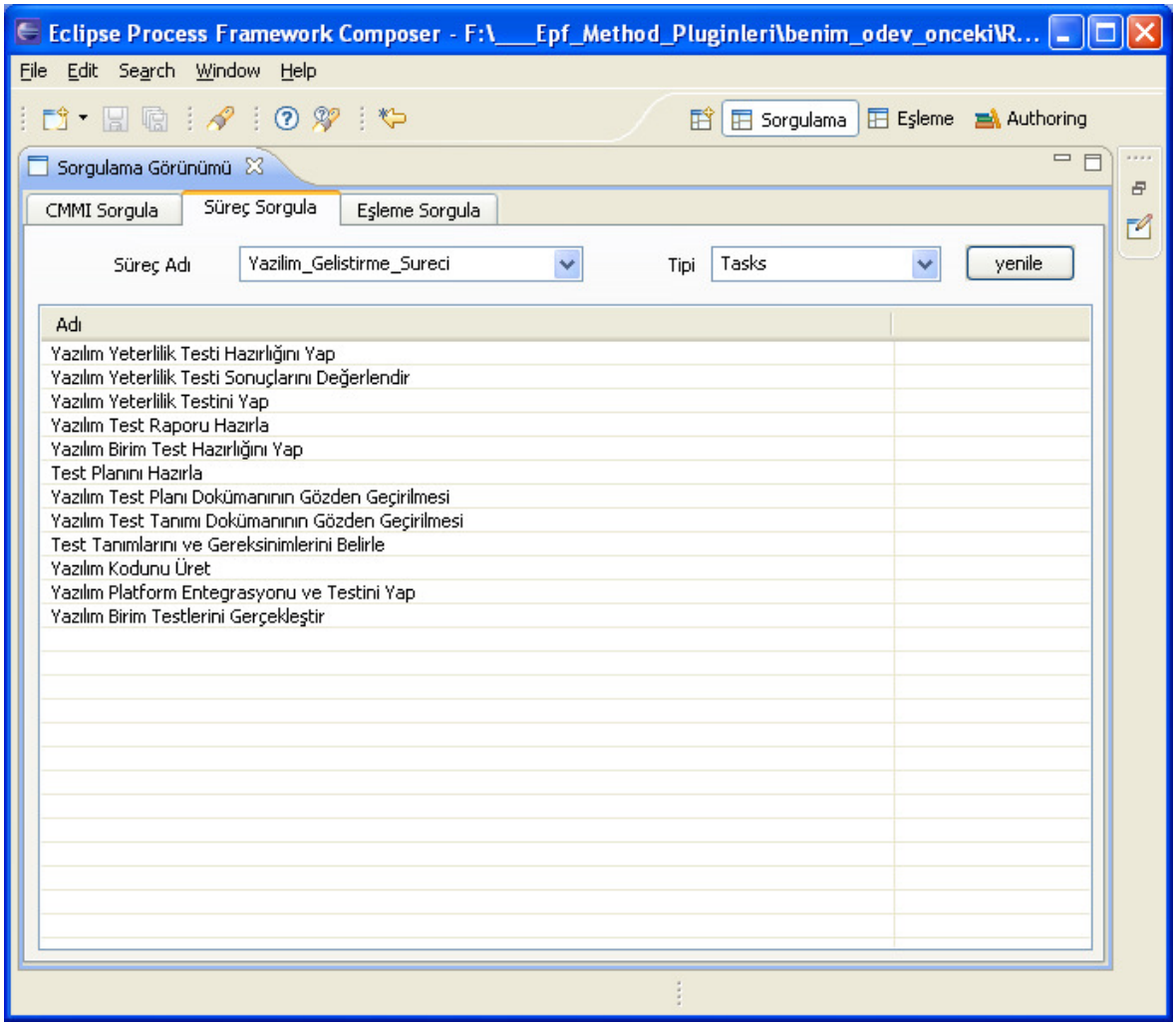
CMMI sorgusu ile ilgili örnek bir ekran görüntüsü Şekil 42'de verilmiştir.



Şekil 42 - CMMI Sorgusu

5.5.2. YGS Sorguları

Ontolojisi oluşturulan bir süreç ile ilgili sorgulamalar kapsamında, bu süreçte yer alan aktiviteler (activities), görevler (tasks), iş ürünleri (work products) ve roller (roles) listelenmektedir. Bu sorgulamaya örnek olarak, YGS'de yer alan görevlerin listelenmesine ilişkin ekran görüntüsü Şekil 43'te verilmiştir.



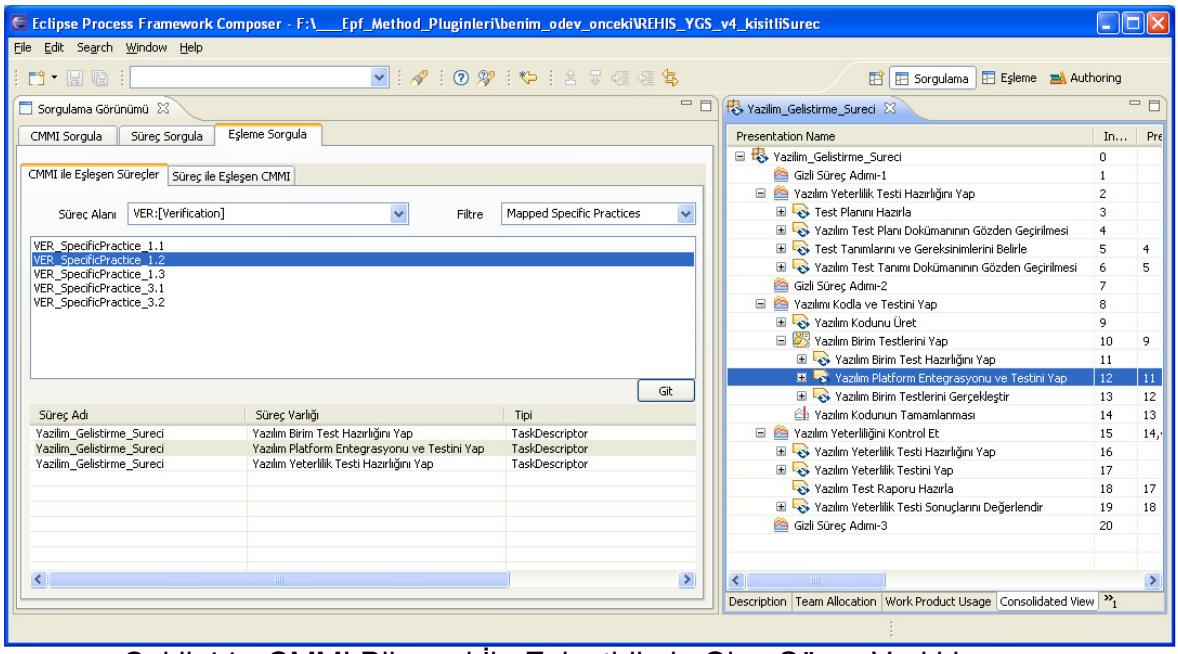
Şekil 43 - Süreç Sorgusu: YGS'deki Görevler

5.5.3. YGS-CMMI Eşlemesi İle İlgili Sorgular

Süreç-CMMI eşlemesi ile ilgili sorgulamalar kapsamında, iki farklı açıdan sorgulama yapılabilmektedir. Bunlardan biri; CMMI bileşeni ile eşleşen süreç varlıklarının listelenmesi, diğeri de bir süreç varlığı ile eşleşen CMMI bileşenlerinin listelenmesidir.

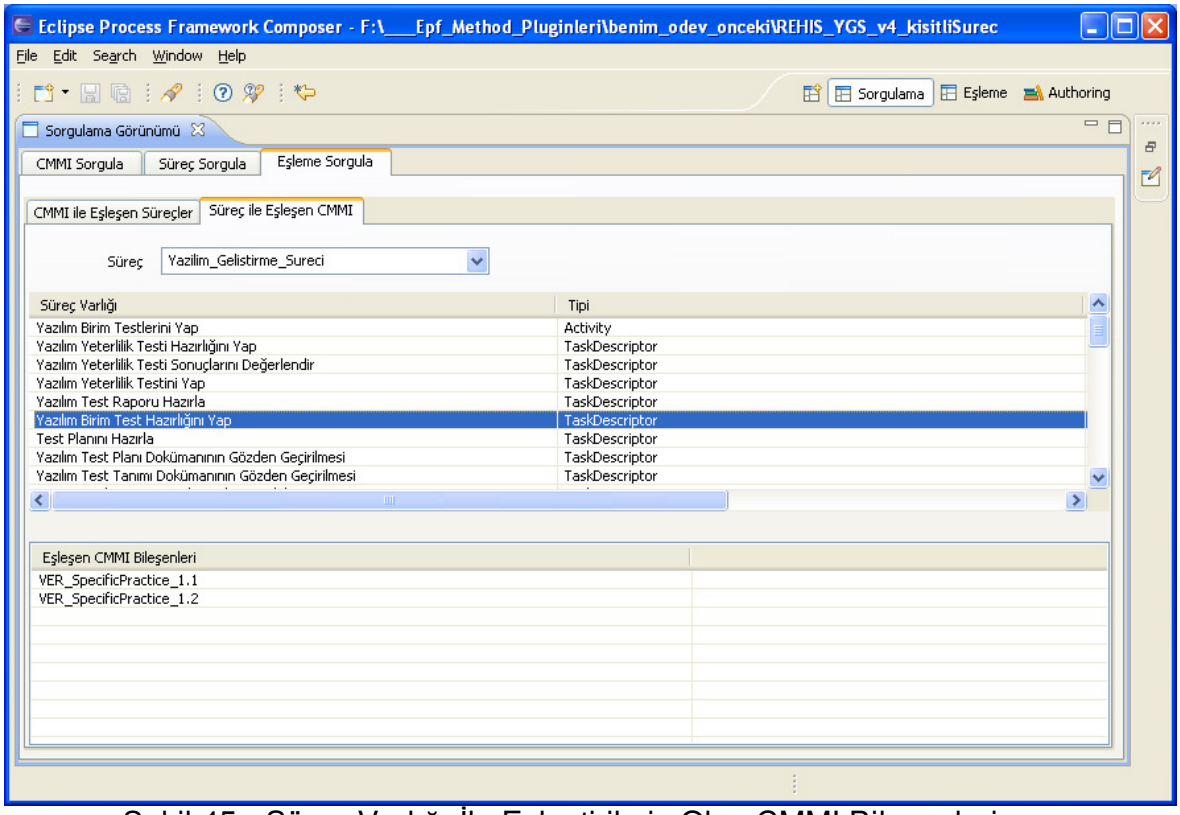
CMMI bileşeni ile eşleşen süreç varlıklarının listelendiği sorgulama için; seçilen bir CMMI süreç alanının kurum süreçleri ile eşleşen ve eşleşmeyen, özel ve genel, pratikleri ve hedefleri listelenebilmektedir. Eşleşen bir pratik veya hedef seçildiğinde ise, eşleştirilmiş olan süreç varlıkları listelenmektedir. CMMI'daki Verification (Doğrulama) süreç alanının YGS ile eşleşen süreç varlıklarının ve özel pratik 1.2 ile eşlenen YGS süreç varlıklarının listelendiği ekran görüntüsü Şekil 44'te görülmektedir. Süreç varlıklarının yer aldığı listeden bir süreç varlığı seçildikten sonra *Git* düğmesine basıldığında ise, bu süreç adımının süreçte yer

aldığı satır uygulama tarafından işaretlenmektedir. Belli bir süreç alanının pratikleri ile eşleşen süreç varlıkları, süreç değerlendirme aktivitesinin verilerin toplanması aşamasında kullanılabilir. Belli bir süreç alanındaki hiçbir süreç varlığı ile eşleştirilmemiş olan pratiklerin listelenmesi ise süreç iyileştirme etkinliklerinde yönlendirici bir girdi olarak ele alınabilir.



Şekil 44 - CMMI Bileşeni İle Eşleştirilmiş Olan Süreç Varlıkları

Bir süreç varlığı ile eşleşen CMMI bileşenlerinin listelendiği sorgulama için; *Süreç* açılan kutusundan (combo box) Yazılım Geliştirme Süreci seçildikten sonra, bu süreçte yer alan tüm süreç varlıkları listelenmiştir. Şekil 45'te de görülen bu listeden, YGS'de yer alan *Yazılım Birim Test Hazırlığını Yap* süreç adımı seçildiğinde ise, pencerenin alt tarafında yer alan *Eşleşen CMMI Bileşenleri* listesinde, CMMI'daki Verification (Doğrulama) süreç alanının 1.1 ve 1.2 özel pratiklerinin (specific practice) listelendiği görülmüştür. Bu sorgulama ile, Yazılım Birim Test Hazırlığı süreç adımında yapılacak bir değişiklikten Doğrulama süreç alanındaki 1.1 ve 1.2 numaralı özel pratiklerin etkilendiği kolayca görülebilmektedir. Bu yetenek özellikle süreç iyileştirme etkinlikleri sırasında CMMI ile uyumun bozulmamasını takip etmek için kullanılabilir.



Şekil 45 - Süreç Varlığı İle Eşleştirilmiş Olan CMMI Bileşenleri

6. SONUÇ

Bir yazılım ürününün geliştirilmesinde ve idamesinde kullanılan süreçlerin, bu yazılımın kalitesi üzerindeki etkisinin kabulünden hareketle, kurumsal süreçlerin iyileştirilmesi ve değerlendirilmesi ile ilgili modeller ve standartlar ortaya konulmuştur. Bir kurumun önemli varlıklarından ve bilgi birikimlerinden biri olan süreçlerin, bu modeller ve standartlar ile uyumluluğunun izlenmesi, değerlendirilmesi gibi aktiviteler önemlidir. Bu aktiviteleri destekleyecek araçlar kullanmak ise, süreç odaklı yönetimi benimseyen kurumlar için neredeyse bir zorunluluk olarak ortaya çıkmaktadır.

Bu araçlarda ontolojilerin kullanılması bazı faydalar sağlayacaktır. Ontolojiler süreç referans modellerinin/standartlarının makineler tarafından anlaşılmasını ve paylaşılmasını sağlayacaklardır. Ayrıca, kurum süreçlerinin de ontolojiler ile ifade edilmesi; biçimsel ifade, uyumluluk, dönüşümlülük, paylaşım gibi faydalar sağlayacaktır. Bunun da ötesinde, süreç referans modeli/standardı ontolojileri ile kurum süreçlerinin ontolojileri arasında eşlemeler kurulmasının ve bu eşlemelerin sorgulanmasının, süreç iyileştirmede ve süreç değerlendirme faaliyetlerinin veri toplama aşamasında faydalı olacağı düşünülmektedir.

Bu düşüncelerden hareketle, tez kapsamında, CMMI-Dev için bir ontoloji geliştirilmiş ve bu ontolojiyi de kullanan ontoloji-tabanlı bir araç, mevcut bir süreç yönetimi aracı olan EPF'ye ek yetenekler kazandırılarak oluşturulmuştur. Ek yetenekler, Eclipse eklentilerinin geliştirilmesi ile sağlanmış ve oluşturulan yeni araç OCMQ-E olarak adlandırılmıştır.

OCMQ-E ile özetle şunlar gerçekleştirilmiştir;

- CMMI ontolojisi, EPF ile bütünleştirilmiştir.
- Orijinal EPF'nin süreç yazma yetenekleri kullanılarak yazılan süreçlerin ontolojileri oluşturulabilmektedir.
- Ontolojisi oluşturulan süreçler ile CMMI süreç alanları arasında eşlemeler kurulabilmekte ve bu eşleme bilgileri bir ontolojide saklanmaktadır.

- CMMI ontolojisi sorgulanarak, CMMI'daki bir süreç alanında yer alan hedefler ve pratikler listelenebilmekte, bir pratik ya da hedef seçildiğinde ise, bununla ilgili özet bilgiye ulaşılabilmektedir.

- Ontolojisi oluşturulan süreçler sorgulanabilmekte ve bu sorgulama sonucunda, bir süreçte yer alan; aktiviteler, görevler, iş ürünleri ve roller listelenebilmektedir.

- CMMI-süreç eşlemelerini içeren ontolojinin sorgulanması ile, bir CMMI bileşeni ile eşlenen süreç varlıkları ve bir süreç varlığı ile eşlenen CMMI bileşenleri listelenebilmektedir.

OCMQ-E'nin kullanımı ve işlevselliği, Aselsan A.Ş. REHİS Grubu Yazılım Geliştirme Süreci ile örneklenmiştir. Bu örnekleme sonucunda şunlar söylenebilir;

- CMMI ile süreçler arasında eşleme kurulurken ve bu eşlemeleri yönetirken, bu amaç için geliştirilmiş bir araç kullanmanın Excel tablolarından daha hızlı ve kolay bir kullanım ortamı sağladığı gözlenmiştir. Araç kullanıldığında, her seferinde, süreç ile ilgili bilgi ihtiyacı olduğunda süreç dokümanlarına, CMMI ile ilgili bilgi ihtiyacı olduğunda da CMMI dokümanına bakılmasına çoğu zaman gerek kalmamaktadır. Başlangıçta süreçlerin doğru ve detaylı bir şekilde ifade edilmesi için emek ve zaman harcanması gerekmekle birlikte, bu iş bir kez yapıldıktan sonra, eşlemelerin yönetimi için kolay kullanımlı bir ortam sağlamış olmaktadır.

- OCMQ-E'de yer alan ontolojilerin sorgulanması yoluyla, süreç iyileştirme ve değerlendirme için önemli olduğu düşünülen bilgilere kolay, hızlı ve doğru bir şekilde erişildiği gözlenmiştir.

- Süreçlerin yönetiminin ve CMMI–süreç eşlemelerinin aynı araç kullanılarak yapılması, süreç iyileştirme ve değerlendirme için bütünleşik bir ortam sağlamaktadır. Süreçlerde değişiklik yaparken, CMMI ile uyumun bundan nasıl etkileneceği, CMMI–süreç eşlemeleri yapılırken de süreçlerdeki eksikler ve hatalar görülebilmektedir. Ayrıca, sağlanan bu bütünleşik ortam, aynı bilginin farklı ortamlarda tekrarlanarak yer

almasını, böylece de verinin tutarsızlığı ihtimalini engellemesi açısından da önemli görülmektedir.

Aracın getirdiği kolaylıklar ise şöyle özetlenebilir;

- Süreç ile CMMI arasında eşleme kurmak için ve kurulan eşlemeleri yönetmek için kolay bir kullanım ortamı sunulmaktadır.
- Süreç ile CMMI arasında eşlemeler kurulduğunda;
  - Bir CMMI bileşeni ile eşlenen süreç varlıkları listelenebilmektedir.
  - Bir süreç varlığı ile eşlenen CMMI bileşenleri listelenebilmektedir.
- Araç ortamından ayrılmaya gerek kalmadan, CMMI pratikleri ve hedefleri ile ilgili hatırlama amaçlı bilgi edinilebilmektedir.

KAYNAKLAR

[1] Dean Leffingwell & Don Widrig. Managing Software Requirements – A Unified Approach. Addison Wesley, 2000.

[2] Daniel Galin, Software Quality Assurance: From Theory To Implementation, Pearson 2004.

[3] ISO. ISO/IEC 12207, Systems And Software Engineering – Software Life Cycle Processes. IEEE Std 12207-2008, Second Edition, 01-02-2008.

[4] Ian Sommerville, Software Engineering, Addison Wesley, 7th Edition, 2004.

[5] W. Humprey. A Discipline for Software Engineering. Addison-Wesley, 1995.

[6] Software & Systems Process Engineering Metamodel Specification (SPEM), version 2.0, OMG Document Number: formal/2008-04-01, Object Management Group (OMG), 2008.

[7] ISO, ISO/IEC 15504, Information Technology - Process Assessment, 2003-2008.

[8] CMMI Product Team, CMU/SEI. CMMI for Development. CMMI-SE/SW V1.2, CMU/SEI-2006-TR-008, ESC-TR-2006-008, August 2006. http://www.sei.cmu.edu/reports/06tr008.pdf
(ziyaret tarihi: 14.09.2009)

[9] N.R. Tague. The Quality Toolbox, 2nd Ed., ASQ Quality Press, pp. 390-392, ISBN 978-0-87389-639-9, 2004.

[10] ISO, ISO 9001:2000 Quality Management Systems -- Requirement, 2000.

[11] Eclipse Process Framework Project (EPF).
http://www.eclipse.org/epf/
(ziyaret tarihi: 14.09.2009)

[12] Eclipse Foundation.
http://www.eclipse.org/
(ziyaret tarihi: 14.09.2009)

[13] OMG, Object Management Group.
http://www.omg.org/
(ziyaret tarihi: 14.09.2009)

[14] W3C. Web Ontology Language Reference OWL, 2004.
http://www.w3.org/2004/OWL/
(ziyaret tarihi: 14.09.2009)

[15] Carnegie Mellon, Software Engineering Institute.
http://www.sei.cmu.edu/
(ziyaret tarihi: 14.09.2009)

[16] CMMI Product Team, CMU/SEI. CMMI for Services. CMMI-SVC V1.2, CMU/SEI-2009-TR-001, ESC-TR-2009-01, February 2009.

[17] CMMI Product Team, CMU/SEI. CMMI for Acquisition. CMMI-ACQ V1.2, CMU/SEI-2007-TR-017, ESC-TR-2007-017, November 2007.

[18] Stanford Center for Biomedical Informatics Research, Protégé-OWL Ontology Editor, version : 3.3.1. http://protege.stanford.edu/overview/protege-owl.html (ziyaret tarihi: 14.09.2009)

[19] Aselsan A.Ş. http://www.aselsan.com.tr/ (ziyaret tarihi: 14.09.2009)

[20] Aselsan A.Ş. Mikrodalga ve Sistem Teknolojileri Grubu, Yazılım Geliştirme Süreci, MSSD-SG-20, Kasım 2005, Revizyon A.

[21] ISO/IEC, 15504-9, Information technology — Software process assessment —Part 9:Vocabulary, First edition, 1998-08-15.

[22] ISO. International Organization for Standardization http://www.iso.org/iso/home.htm (ziyaret tarihi: 14.09.2009)

[23] ISO, ISO/IEC 15288: Systems And Software Engineering – System Life Cycle Processes, 2008.

[24] J. Gremba, C. Myers. The IDEAL model: a practical guide for improvement. Software Engineering Institute Bridge, 1997.

[25] H.K.N. Leung, L. Liao, Y. Qu. Automated support of software quality improvement. International Journal of Quality & Reliability Management, Vol. 24 No. 3, pp. 230-243, 2007.

[26] SCAMPI Upgrade Team. Standard CMMI Appraisal Method for Process Improvement (SCAMPI) A, Version 1.2: Method Definition Document, Handbook, CMU/SEI-2006-HB-002, August 2006.

[27] SCAMPI Upgrade Team. Appraisal Requirements for CMMI, Version 1.2 (ARC, V1.2), Technical Report, CMU/SEI-2006-TR-011, August 2006.

[28] P.H. Feiler, W.S. Humphrey. Software Process Development and Enactment: Concepts and Definitions. Proceedings of the Second International Conference on Software Process, February 1993.

[29] B. Curtis, M.I. Kellner, J. Over. Process Modeling. Communications of the ACM, Vol.35, No.9, September 1992.

[30] L.J. Osterweil. Software Processes Are Software Too, Revisited:An Invited Talk on the Most Influential Paper of ICSE 9 *. Proceedings of the 1997 (19th) International Conference on Software Engineering, pages: 540 – 548, May 1997.

[31] M.I. Kellner, G.A. Hansen. Software Process Modeling, Technical Report, CMU/SEI-88-TR-009, May 1988.

[32] OMG, Meta-Object Facility Version 2.
http://www.omg.org/mof/
(ziyaret tarihi: 14.09.2009)

[33] B. Chandrasekaran, J.R. Josephson, V.R. Benjamins. What Are Ontologies, and Why Do We Need Them? IEEE Intelligent Systems Vol. 14, Issue 1, pages 20-26, 1999.

[34] T.R. Gruber. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. International Journal Human-Computer Studies 43, pages :907-928, 1995.

[35] J.R.G. Pulido, M.A.G. Ruiz, R. Herrera, E. Cabello, S. Legrand, D. Elliman. Ontology languages for the semantic web: A never completely updated review. Knowledge-Based Systems, Volume 19, Issue 7, pages : 489–497, November, 2006.

[36] W3C. Resource Description Framework (RDF): Concepts and Abstract Syntax, W3C Recommendation, 10 February 2004.

[37] W3C. RDF/XML Syntax Specification (Revised), W3C Recommendation, 10 February 2004.

[38] Y. Ding, D. Fensel, M. Klein, B. Omelayenko. The SemanticWeb: Yet Another Hip? Data and Knowledge Engineering, 2002.

[39] The DARPA Agent Markup Language
http://www.daml.org/
(ziyaret tarihi: 14.09.2009)

[40] D. Fensel. The semantic Web and its languages. IEEE, November/December 2002

[41] F. Van Harmelen, I. Horrocks.Questions and answers on OIL: the Ontology Inference Layer for the semantic web. IEEE Intelligent Systems volume 15, number 6, pages 69-72, December 2000.

[42] D. Fensel, I. Horrocks, F. Van Harmelen, S. Decker, M. Erdmann, M. Klein. OIL in a Nutshell.

[43] A. Gómez-Pérez, O. Corcho. Ontology languages for semantic web. IEEE Intelligent Systems, 2002.

[44] J. de Bruijn. Using Ontologies - Enabling Knowledge Sharing and Reuse on the Semantic Web. Digital Enterprise Research Institute Technical Report DERI-2003-10-29 October 2003

[45] W3C. Owl Web Ontology Language Reference. Technical Report, February 2004.

[46] W3C. OWL Web Ontology Language Overview. W3C Recommendation, 10 February 2004.

[47] OntoWeb Consortium. Deliverable 1.3: A survey on ontology tools. OntoWeb Ontology-based information exchange for knowledge management and electronic commerce IST-2000-29243. May, 2002.

[48] M. Denny. Ontology Tools Survey, Revisited. Technical report, XML.com, July 2004.

[49] A. Gómez-Pérez, M. Fernández-López, O. Corcho. Ontological Engineering. AI Magazine, 1991.

[50] Marek Obitko. Ontologies and Semantic Web. http://www.obitko.com/tutorials/ontologies-semantic-web/ (ziyaret tarihi: 06.07.2009)

[51] V. Iosif, P. Mika, On-To-Knowledge: EnerSearch Virtual Organization, Case Study: Evaluation Document, Deliverable 29, September, 2002.

[52] Z. Zhang. Ontology Query Languages For The Semantic Web: A Performance Evaluation. A Master of Science Degree Thesis, The University Of Georgia, Athens, 2005.

[53] C. Aniszczyk, D. Gallardo. Get Started With The Eclipse Platform, IBM, July 2007.

[54] Object Technology International, Inc. Eclipse Platform Technical Overview. February 2003 (updated for 2.1; originally published July 2001)

[55] A. Bolour. Notes on the Eclipse Plug-in Architecture, Eclipse Corner Article, 2003.

[56] Eclipse SDK, version 3.4.1, Workbench User Guide.

[57] N. Edgar, K. Haaland, J. Li, K. Peter. Eclipse User Interface Guidelines, Version 2.1, Last Updated: February 2004.

[58] D. Springgay, Creating an Eclipse View, Object Technology International, Inc. November, 2001.

[59] P. Haumer. Introducing the Eclipse Process Framework. EclipseCon, 2006.

[60] Protégé
http://protege.stanford.edu/
(ziyaret tarihi: 14.09.2009)

[61] Jena
http://jena.sourceforge.net/
(ziyaret tarihi: 14.09.2009)

[62] G.H. Soydan, M.M. Kokar. An OWL Ontology for Representing the CMMI-SW Model. In ISWC 2006 Workshop on Semantic Web Enabled Software Engineering, 2006.

[63] A.A. Sharifloo, Y. Motazedi, M. Shamsfard, R. Dehkharghani. An Ontology for CMMI-ACQ Model. In 3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA), April 2008.

[64] Suggested Upper Merged Ontology (SUMO).
http://www.ontologyportal.org/
(ziyaret tarihi : 14.09.2009)

[65] Knowledge Interchange Format, draft proposed American National Standard (dpANS), NCITS.T2/98-004, Ed.

[66] S. Rungratri, S. Usanavasin. Project Assets Ontology (PAO) to Support Gap Analysis for Organization Process Improvement Based on CMMI v.1.2. (Book) Making Globally Distributed Software Development a Success Story, Pages: 76-87, Springer Berlin / Heidelberg, May 2008.

[67] L. Liao, Y. Qu, H. Leung. A Software Process Ontology and Its Application. In ISWC 2005 Workshop on Semantic Web Enabled Software Engineering, 2005.

[68] J. G. Doheny, I. M. Filby. A Framework and Tool for Modelling and Assessing Software Development Processes. In The European Software Control and Metrics Conference, Wilmslow, May 1996.

[69] F. García, M. Piattini, F. Ruiz, G. Canfora, C. A. Visaggio. FMESP: Framework for the Modeling and Evaluation of Software Processes. Journal of Systems Architecture: the EUROMICRO Journal, Volume 52 , Issue 11, Pages 627 – 639, November 2006.

[70] C.-S. Lee, M.-H. Wang. Ontology-based Computational Intelligent Multi-agent and Its Application to CMMI Assessment. Springer Science+Business Media, LLC 2007.

[71] C.-S. Lee, M.-H. Wang, J.-J. Chen, C.-Y. Hsu. Ontology-based Intelligent Decision Support Agent for CMMI Project Monitoring and Control. International Journal of Approximate Reasoning, volume 48, pages : 62–76, April 2008.

[72] SPI (Software Process Improvements) Partners.
http://www.spipartners.nl/english/tools/index.html
(ziyaret tarihi: 14.09.2009)

[73] Appraisal Assistant, Griffith University, Software Quality Institute.
http://www.sqi.gu.edu.au/AppraisalAssistant/about.html
(ziyaret tarihi: 14.09.2009)

[74] http://www.chemuturi.com/cmmipaldtls.html
(ziyaret tarihi: 14.09.2009)

[75] http://www.chemuturi.com/
(ziyaret tarihi: 14.09.2009)

[76] http://www.cmm-quest.com/
(ziyaret tarihi: 14.09.2009)

[77] HM&S IT-Consulting
http://www.hms.org/english/default.htm
(ziyaret tarihi: 14.09.2009)

[78] http://www.spice121.com/
(ziyaret tarihi: 14.09.2009)

[79] http://www.spicelite.com
(ziyaret tarihi: 14.09.2009)

[80] Model Wizard
http://isd-inc.com/tools.modelWizard/
(ziyaret tarihi: 14.09.2009)

[81] http://isd-inc.com/
(ziyaret tarihi: 14.09.2009)

[82] Integrated System Diagnostics, Inc. Model Wizard User Guide – V1, 2005.

[83] Appraisal Wizard
http://isd-inc.com/tools.appraisalWizard/
(ziyaret tarihi: 14.09.2009)

[84] Appraisal Wizard V7 User Guide V7.0, Integrated System Diagnostics, Inc.

[85] http://www.man-info-systems.com/index_files/FreeTools.htm
(ziyaret tarihi: 14.09.2009)

[86] Management Information Systems bvba
http://www.man-info-systems.com/
(ziyaret tarihi: 14.09.2009)

Ek-1: Süreç Yapısı Ontolojisi (SYO): SurecYapisiOntolojisi.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:protege="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns="http://www.owl-ontologies.com/SurecYapisiOntolojisi.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/SurecYapisiOntolojisi.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="TaskDescriptor">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:someValuesFrom>
          <owl:Class rdf:ID="RoleDescriptor"/>
        </owl:someValuesFrom>
        <owl:onProperty>
          <owl:ObjectProperty rdf:ID="includes"/>
        </owl:onProperty>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#includes"/>
        <owl:someValuesFrom>
          <owl:Class rdf:ID="WorkProductDescriptor"/>
        </owl:someValuesFrom>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Iteration">
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:someValuesFrom>
          <owl:Class rdf:ID="Phase"/>
        </owl:someValuesFrom>
        <owl:onProperty rdf:resource="#includes"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:someValuesFrom rdf:resource="#Iteration"/>
        <owl:onProperty rdf:resource="#includes"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:someValuesFrom>
          <owl:Class rdf:ID="Milestone"/>
        </owl:someValuesFrom>
        <owl:onProperty rdf:resource="#includes"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#includes"/>
        <owl:someValuesFrom>
```

```
        <owl:Class rdf:ID="Activity"/>
      </owl:someValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom>
        <owl:Class rdf:ID="TeamProfile"/>
      </owl:someValuesFrom>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#includes"/>
      <owl:someValuesFrom rdf:resource="#RoleDescriptor"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Activity">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#Activity"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#includes"/>
      <owl:someValuesFrom>
        <owl:Class rdf:about="#Phase"/>
      </owl:someValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#Milestone"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#TaskDescriptor"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#TeamProfile">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#includes"/>
      <owl:someValuesFrom rdf:resource="#TeamProfile"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Phase">
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <rdfs:subClassOf>
    <owl:Restriction>
```

```
      <owl:someValuesFrom rdf:resource="#Activity"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#Phase"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#Iteration"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#includes"/>
      <owl:someValuesFrom rdf:resource="#Milestone"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#includes"/>
      <owl:someValuesFrom rdf:resource="#RoleDescriptor"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#includes"/>
      <owl:someValuesFrom rdf:resource="#TeamProfile"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="DeliveryProcess">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#includes"/>
      <owl:someValuesFrom rdf:resource="#RoleDescriptor"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#Phase"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#TaskDescriptor"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#Activity"/>
      <owl:onProperty rdf:resource="#includes"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
```

```
      <owl:Restriction>
        <owl:onProperty rdf:resource="#includes"/>
        <owl:someValuesFrom rdf:resource="#WorkProductDescriptor"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#includes"/>
        <owl:someValuesFrom rdf:resource="#TeamProfile"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#includes"/>
        <owl:someValuesFrom rdf:resource="#Iteration"/>
      </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#includes"/>
        <owl:someValuesFrom rdf:resource="#Milestone"/>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:DataRange>
    <owl:oneOf rdf:parseType="Resource">
      <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >System</rdf:first>
      <rdf:rest rdf:parseType="Resource">
        <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Software</rdf:first>
        <rdf:rest rdf:parseType="Resource">
          <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Hardware</rdf:first>
          <rdf:rest rdf:parseType="Resource">
            <rdf:rest rdf:parseType="Resource">
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#nil"/>
              <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
              >Undecided</rdf:first>
            </rdf:rest>
            <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >General</rdf:first>
          </rdf:rest>
        </rdf:rest>
      </rdf:rest>
    </owl:oneOf>
  </owl:DataRange>
  <owl:DataRange>
    <owl:oneOf rdf:parseType="Resource">
      <rdf:rest rdf:parseType="Resource">
        <rdf:rest rdf:parseType="Resource">
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#nil"/>
          <rdf:first
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >Undecided</rdf:first>
```

```
        </rdf:rest>
        <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >Indirect</rdf:first>
      </rdf:rest>
      <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
      >Direct</rdf:first>
    </owl:oneOf>
  </owl:DataRange>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3.1, Build 430)
http://protege.stanford.edu -->
```

Ek-2: Bir Süreç Ontolojisi (SO) Örneği: Yazilim_Gelistirme_Sureci.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns="http://www.owl-ontologies.com/Yazilim_Gelistirme_Sureci.owl#"
    xmlns:syo="http://www.owl-ontologies.com/SurecYapisiOntolojisi.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/Yazilim_Gelistirme_Sureci.owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://www.owl-
ontologies.com/SurecYapisiOntolojisi.owl"/>
  </owl:Ontology>
  <syo:Milestone rdf:ID="_4Nzx8FJuEd6Q7v481bwhZg"/>
  <syo:WorkProductDescriptor rdf:ID="_K3_Q4VXXEd6EwvaCzI3Y1A"/>
  <syo:RoleDescriptor rdf:ID="_NZj9IlXUEd6EwvaCzI3Y1A"/>
  <syo:RoleDescriptor rdf:ID="_VWphwVXPEd6EwvaCzI3Y1A"/>
  <syo:Phase rdf:ID="_cXifoc2EEdyyPZbL1RG7VA">
    <syo:includes>
      <syo:WorkProductDescriptor rdf:ID="_1HenYFXVEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:WorkProductDescriptor rdf:ID="_1HenYVXVEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:WorkProductDescriptor rdf:ID="_Emg9slXVEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:RoleDescriptor rdf:ID="_Emg9sVXVEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:WorkProductDescriptor rdf:ID="_IINYQlXUEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:RoleDescriptor rdf:ID="_NZj9IVXUEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:RoleDescriptor rdf:ID="_NZj9IFXUEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:RoleDescriptor rdf:ID="_1HVdcVXVEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:WorkProductDescriptor rdf:ID="_IINYQVXUEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:TaskDescriptor rdf:ID="_lNeXAFkxEd6p_-O7RsCbbg"/>
    </syo:includes>
    <syo:includes>
      <syo:TaskDescriptor rdf:ID="_1HVdcFXVEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:TaskDescriptor rdf:ID="_Emg9sFXVEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes>
      <syo:RoleDescriptor rdf:ID="_1HVdclXVEd6EwvaCzI3Y1A"/>
    </syo:includes>
    <syo:includes rdf:resource="#_NZj9IlXUEd6EwvaCzI3Y1A"/>
    <syo:includes>
```

```
    <syo:TaskDescriptor rdf:ID="_LZE2oFXVEd6EwvaCzI3Y1A"/>
  </syo:includes>
</syo:Phase>
<syo:TaskDescriptor rdf:ID="_JE9NAFXXEd6EwvaCzI3Y1A"/>
<syo:RoleDescriptor rdf:ID="_RvNd0VXLEd6EwvaCzI3Y1A"/>
<syo:TaskDescriptor rdf:ID="_NDrTgHeQEd6YcMgURT3pTA"/>
<syo:Phase rdf:ID="_2N6E4WyHEd6fJaL3SylIUw"/>
<syo:WorkProductDescriptor rdf:ID="_JE9NBFXXEd6EwvaCzI3Y1A"/>
<syo:Phase rdf:ID="_AI5HUWyIEd6fJaL3SylIUw"/>
<syo:WorkProductDescriptor rdf:ID="_RvNd1FXLEd6EwvaCzI3Y1A"/>
<syo:TaskDescriptor rdf:ID="_P59A8FkvEd6p_-O7RsCbbg"/>
<syo:DeliveryProcess rdf:ID="_OI9hEc2EEdyyPZbL1RG7VA">
  <syo:includes>
    <syo:Phase rdf:ID="_2KFFkXePEd6YcMgURT3pTA"/>
  </syo:includes>
  <syo:includes>
    <syo:Phase rdf:ID="_X6o2gc2EEdyyPZbL1RG7VA">
      <syo:includes rdf:resource="#_RvNd1FXLEd6EwvaCzI3Y1A"/>
      <syo:includes rdf:resource="#_P59A8FkvEd6p_-O7RsCbbg"/>
      <syo:includes rdf:resource="#_RvNd0VXLEd6EwvaCzI3Y1A"/>
      <syo:includes>
        <syo:TaskDescriptor rdf:ID="_RjKacFkvEd6p_-O7RsCbbg"/>
      </syo:includes>
      <syo:includes>
        <syo:WorkProductDescriptor rdf:ID="_P8VmkFkvEd6p_-O7RsCbbg"/>
      </syo:includes>
      <syo:includes rdf:resource="#_NDrTgHeQEd6YcMgURT3pTA"/>
      <syo:includes>
        <syo:RoleDescriptor rdf:ID="_P7c1wFkvEd6p_-O7RsCbbg"/>
      </syo:includes>
      <syo:includes>
        <syo:WorkProductDescriptor rdf:ID="_RvNd01XLEd6EwvaCzI3Y1A"/>
      </syo:includes>
      <syo:includes>
        <syo:RoleDescriptor rdf:ID="_RkyyIFkvEd6p_-O7RsCbbg"/>
      </syo:includes>
      <syo:includes>
        <syo:TaskDescriptor rdf:ID="_SYx_QFXLEd6EwvaCzI3Y1A"/>
      </syo:includes>
      <syo:includes>
        <syo:RoleDescriptor rdf:ID="_P7c1wVkvEd6p_-O7RsCbbg"/>
      </syo:includes>
      <syo:includes>
        <syo:RoleDescriptor rdf:ID="_RvNd0lXLEd6EwvaCzI3Y1A"/>
      </syo:includes>
      <syo:includes>
        <syo:WorkProductDescriptor rdf:ID="_SYx_QVXLEd6EwvaCzI3Y1A"/>
      </syo:includes>
    </syo:Phase>
  </syo:includes>
  <syo:includes>
    <syo:Phase rdf:ID="_0efGcXePEd6YcMgURT3pTA"/>
  </syo:includes>
  <syo:includes>
    <syo:Phase rdf:ID="_alQCoc2EEdyyPZbL1RG7VA">
      <syo:includes>
        <syo:TaskDescriptor rdf:ID="_VWphwFXPEd6EwvaCzI3Y1A"/>
      </syo:includes>
      <syo:includes>
        <syo:Activity rdf:ID="_0KfFgc2GEdyyPZbL1RG7VA">
          <syo:includes rdf:resource="#_JE9NAFXXEd6EwvaCzI3Y1A"/>
```

```
            <syo:includes>
              <syo:RoleDescriptor rdf:ID="_JE9NAVXXEd6EwvaCzI3Y1A"/>
            </syo:includes>
            <syo:includes>
              <syo:RoleDescriptor rdf:ID="_JE9NAlXXEd6EwvaCzI3Y1A"/>
            </syo:includes>
            <syo:includes>
              <syo:WorkProductDescriptor
rdf:ID="_JE9NA1XXEd6EwvaCzI3Y1A"/>
            </syo:includes>
            <syo:includes rdf:resource="#_JE9NBFXXEd6EwvaCzI3Y1A"/>
            <syo:includes>
              <syo:TaskDescriptor rdf:ID="_Mr6soFXXEd6EwvaCzI3Y1A"/>
            </syo:includes>
            <syo:includes>
              <syo:TaskDescriptor rdf:ID="_K3_Q4FXXEd6EwvaCzI3Y1A"/>
            </syo:includes>
            <syo:includes rdf:resource="#_K3_Q4VXXEd6EwvaCzI3Y1A"/>
          </syo:Activity>
        </syo:includes>
        <syo:includes rdf:resource="#_4Nzx8FJuEd6Q7v481bwhZg"/>
        <syo:includes rdf:resource="#_VWphwVXPEd6EwvaCzI3Y1A"/>
        <syo:includes>
          <syo:WorkProductDescriptor rdf:ID="_VWphwlXPEd6EwvaCzI3Y1A"/>
        </syo:includes>
        <syo:includes>
          <syo:WorkProductDescriptor rdf:ID="_VWphw1XPEd6EwvaCzI3Y1A"/>
        </syo:includes>
      </syo:Phase>
    </syo:includes>
    <syo:includes rdf:resource="#_cXifoc2EEdyyPZbL1RG7VA"/>
    <syo:includes>
      <syo:Phase rdf:ID="__vA_gXePEd6YcMgURT3pTA"/>
    </syo:includes>
  </syo:DeliveryProcess>
  <syo:Phase rdf:ID="_texKgWyHEd6fJaL3SylIUw"/>
  <syo:Phase rdf:ID="_MX1wcWyIEd6fJaL3SylIUw"/>
  <syo:DeliveryProcess rdf:ID="_fvW1sWjNEd63Y4cADO2mng">
    <syo:includes rdf:resource="#_MX1wcWyIEd6fJaL3SylIUw"/>
    <syo:includes>
      <syo:Phase rdf:ID="_G0SoUWyIEd6fJaL3SylIUw"/>
    </syo:includes>
    <syo:includes rdf:resource="#_AI5HUWyIEd6fJaL3SylIUw"/>
    <syo:includes>
      <syo:Phase rdf:ID="_5S6uAWyHEd6fJaL3SylIUw"/>
    </syo:includes>
    <syo:includes rdf:resource="#_2N6E4WyHEd6fJaL3SylIUw"/>
    <syo:includes rdf:resource="#_texKgWyHEd6fJaL3SylIUw"/>
    <syo:includes>
      <syo:Phase rdf:ID="_kKYOAWyHEd6fJaL3SylIUw"/>
    </syo:includes>
  </syo:DeliveryProcess>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3.1, Build 430)
http://protege.stanford.edu -->
```

Ek-3: Eşleme Ontolojisi (EO) Örneği: CMMI – YGS Eşlemesine İlişkin EslemeOntolojisi.owl

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://www.owl-ontologies.com/EslemeOntolojisi.owl#"
    xmlns:so1="http://www.owl-
ontologies.com/Sistem_Gelistirme_Sureci.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:cmmi="http://www.owl-ontologies.com/CMMI_Ontology.owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:so0="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/EslemeOntolojisi.owl">
  <owl:Ontology rdf:about="">
    <owl:imports rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl"/>
    <owl:imports rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl"/>
    <owl:imports rdf:resource="http://www.owl-
ontologies.com/Sistem_Gelistirme_Sureci.owl"/>
  </owl:Ontology>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_1.2">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_JE9NAFXXEd6EwvaCzI3Y1A">
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_1.1">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_JE9NAFXXEd6EwvaCzI3Y1A"/>
            <owl:sameAs>
              <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_RvNd0FXLEd6EwvaCzI3Y1A">
                <owl:sameAs>
                  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_GenericPractice_2.2">
                    <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_RvNd0FXLEd6EwvaCzI3Y1A"/>
                  </rdf:Description>
                </owl:sameAs>
                <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_1.1"/>
                <owl:sameAs>
                  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_1.3">
                    <owl:sameAs>
                      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_SYx_QFXLEd6EwvaCzI3Y1A">
                        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_1.3"/>
                      </rdf:Description>
                    </owl:sameAs>
                    <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_RvNd0FXLEd6EwvaCzI3Y1A"/>
                  </rdf:Description>
                </owl:sameAs>
              </rdf:Description>
```

```
            </owl:sameAs>
          </rdf:Description>
        </owl:sameAs>
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_1.2"/>
      </rdf:Description>
    </owl:sameAs>
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_Mr6soFXXEd6EwvaCzI3Y1A">
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_1.2"/>
      </rdf:Description>
    </owl:sameAs>
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_lNeXAFkxEd6p_-O7RsCbbg">
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_1.2"/>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#RD_SpecificPractice_2.3">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_lMKbMFXKEd6EwvaCzI3Y1A">
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#RD_SpecificPractice_3.4">
            <owl:sameAs>
              <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_l7sasFXKEd6EwvaCzI3Y1A">
                <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#RD_SpecificPractice_3.4"/>
              </rdf:Description>
            </owl:sameAs>
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_lMKbMFXKEd6EwvaCzI3Y1A"/>
          </rdf:Description>
        </owl:sameAs>
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#RD_SpecificPractice_2.3"/>
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.2">
            <owl:sameAs>
              <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_frbHwFXQEd6EwvaCzI3Y1A">
                <owl:sameAs>
                  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_3.2">
                    <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_frbHwFXQEd6EwvaCzI3Y1A"/>
                    <owl:sameAs>
                      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_KgmbkFTaEd6x0fvaF_9EBw">
                        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_3.2"/>
                        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.2"/>
```

```
                    </rdf:Description>
                  </owl:sameAs>
                  <owl:sameAs>
                    <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_xfkv4FkzEd6p_-O7RsCbbg">
                      <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_3.2"/>
                      <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.2"/>
                    </rdf:Description>
                  </owl:sameAs>
                  <owl:sameAs>
                    <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_wFuBAFkzEd6p_-O7RsCbbg">
                      <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.2"/>
                      <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_3.2"/>
                    </rdf:Description>
                  </owl:sameAs>
                  <owl:sameAs>
                    <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_Eps5oFTaEd6x0fvaF_9EBw">
                      <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.2"/>
                      <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_3.2"/>
                    </rdf:Description>
                  </owl:sameAs>
                  <owl:sameAs>
                    <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_OQEIUFXJEd6NJ9WDIw662w">
                      <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.2"/>
                      <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_3.2"/>
                    </rdf:Description>
                  </owl:sameAs>
                </rdf:Description>
              </owl:sameAs>
              <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.2"/>
            </rdf:Description>
          </owl:sameAs>
          <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_KgmbkFTaEd6x0fvaF_9EBw"/>
          <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_xfkv4FkzEd6p_-O7RsCbbg"/>
          <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_lMKbMFXKEd6EwvaCzI3Y1A"/>
          <owl:sameAs>
            <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_ZQFp4FkvEd6p_-O7RsCbbg">
              <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.2"/>
            </rdf:Description>
          </owl:sameAs>
          <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_wFuBAFkzEd6p_-O7RsCbbg"/>
          <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_Eps5oFTaEd6x0fvaF_9EBw"/>
```

```
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_OQEIUFXJEd6NJ9WDIw662w"/>
          </rdf:Description>
        </owl:sameAs>
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#RD_SpecificPractice_3.3">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_lMKbMFXKEd6EwvaCzI3Y1A"/>
          </rdf:Description>
        </owl:sameAs>
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#RD_SpecificPractice_1.2">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_lMKbMFXKEd6EwvaCzI3Y1A"/>
          </rdf:Description>
        </owl:sameAs>
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#RD_SpecificPractice_2.2">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_lMKbMFXKEd6EwvaCzI3Y1A"/>
          </rdf:Description>
        </owl:sameAs>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_1.2">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_hewYgFXOEd6EwvaCzI3Y1A">
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_1.2"/>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#REQM_SpecificPractice_1.1">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_DFK8UFkuEd6p_-O7RsCbbg">
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#REQM_SpecificPractice_1.5">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_DFK8UFkuEd6p_-O7RsCbbg"/>
          </rdf:Description>
        </owl:sameAs>
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#REQM_SpecificPractice_1.1"/>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_P59A8FkvEd6p_-O7RsCbbg">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_GenericPractice_2.7">
```

```
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_P59A8FkvEd6p_-O7RsCbbg"/>
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_RjKacFkvEd6p_-O7RsCbbg">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_GenericPractice_2.7"/>
          </rdf:Description>
        </owl:sameAs>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_8OtM8FXOEd6EwvaCzI3Y1A">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.1">
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_8OtM8FXOEd6EwvaCzI3Y1A"/>
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_O2_MIFXOEd6EwvaCzI3Y1A">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_2.1"/>
          </rdf:Description>
        </owl:sameAs>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_otBDYFIHEd664eeY6t5yEw">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#REQM_SpecificPractice_1.2">
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_otBDYFIHEd664eeY6t5yEw"/>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#PI_SpecificPractice_3.1">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_1HVdcFXVEd6EwvaCzI3Y1A">
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#PI_SpecificPractice_3.1"/>
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_3.2">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_1HVdcFXVEd6EwvaCzI3Y1A"/>
          </rdf:Description>
        </owl:sameAs>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_K3_Q4FXXEd6EwvaCzI3Y1A">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_3.1">
```

```
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_Emg9sFXVEd6EwvaCzI3Y1A">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_3.1"/>
          </rdf:Description>
        </owl:sameAs>
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_K3_Q4FXXEd6EwvaCzI3Y1A"/>
        <owl:sameAs>
          <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_LZE2oFXVEd6EwvaCzI3Y1A">
            <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#VER_SpecificPractice_3.1"/>
          </rdf:Description>
        </owl:sameAs>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#PI_SpecificPractice_3.4">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_AzF0cVTaEd6x0fvaF_9EBw">
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#PI_SpecificPractice_3.4"/>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_3.1">
    <owl:sameAs>
      <rdf:Description rdf:about="http://www.owl-
ontologies.com/Yazilim_Gelistirme_Sureci.owl#_VWphwFXPEd6EwvaCzI3Y1A">
        <owl:sameAs rdf:resource="http://www.owl-
ontologies.com/CMMI_Ontology.owl#TS_SpecificPractice_3.1"/>
      </rdf:Description>
    </owl:sameAs>
  </rdf:Description>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3.1, Build 430)
http://protege.stanford.edu -->
```

ÖZGEÇMİŞ

Adı Soyadı : Sema Gazel

Doğum Yeri : Ankara

Doğum Yılı : 1975

Eğitim ve Akademik Durumu :

Lise 1989-1992 Bursa Atatürk Lisesi

Lisans 1992-1997 Hacettepe Üniversitesi - Bilgisayar Mühendisliği Bölümü

Yabancı Dil : İngilizce

İş Tecrübesi : Aselsan A.Ş. 1997-halen