Model Güdümlü Taktiksel Harp Sahası Yönetim Sistemi Tasarımı

Ethem Fatih Can[1], Özgür Aydın Tekin[1] ve Mahmut Devrim Tokcan[2]
[1]Bilkent Üniversitesi Bilgisayar Müh. Bölümü, Ankara
[2]Orta Doğu Teknik Üniversitesi Enformatik Enstitüsü, Ankara
{efcan, oatekin}@cs.bilkent.edu.tr, e159569@metu.edu.tr

Özetçe

Askeri bir harekatta birimler arası mesajlaşma ve görev dağılımı esastır. Müttefik harekatlarda saha bilgisinin müttefik birimler arasında paylaşılabilmesi hayati öneme sahiptir. Ortak bir dil yapısına dayalı alana özgü diller sayesinde birimler çatışma hakkında güncel bilgiye erişebilir ve farklı dillerin entegrasyonu ile müttefik harekatlar yönetilebilir. Bu çalışmada örnek bir THSYS için üç farklı yaklaşım kullanılmış ve kıyaslanmıştır: (1) Geleneksel dil bilgisi geliştirme yaklaşımı, (2) üst-üst modelden üst model geliştirme ve (3) ayrımlama yoluyla model güdümlü tasarım. Model dönüşümleri ile model güdümlü yaklaşımın üretkenlik, tekrar kullanılabilirlik, taşınabilirlik, müştereklik gibi kalite unsurlarına katkısı gösterilmiştir.

Anahtar Sözcükler *:* Harp Yönetim Sistemleri, Model Güdümlü Yazılım Geliştirme, Alana Özgü Dil Tasarımı, UML Ayrımlama Yöntemi, Üst-Model, Model Dönüşümü

Model Driven Development Approach for Tactical Battlefield Management

Abstract

In a military operation distribution of commands, feedback of controls, splitting the intelligence, and execution of missions are the crucial issues. Besides, in a joint war-fight, exchange of information is the most important issue. These issues are handled by battle management systems. The basic aim of such systems is to provide a common language so that any instrument of the combat will be well-informed. This leads to develop domain specific languages (DSL) that can best capture the concerns of the domain and that also conform to a common language for integration. In this work, we elaborate on a sample tactical battlefield management system (TBMS) utilizing the benefits of the model-driven software development (MDSD) such as productivity, reuse, portability, and interoperability. First a traditional approach based on grammarware is taken. Then two MDSD approaches are considered: from scratch and UML Profiling. Finally, the three approaches are compared for several aspects.

Keywords *:* Battle Management System, MDSD, DSL, UML Profiling, Meta-model from scratch, XText, Model to model transformation, model to text transformation

1 Giriş

Bir savaş sahasında birçok birim yer alır. Her birim, genel bir amaç çerçevesinde kendisine atanmış özel bir görevi yerine getirmeye çalışır. Bu birimler farklı saflardan olabileceği gibi aynı safta da olabilir. Burada, bütün birimlerin aynı safta oldukları ve tek bir genel amaç için savaş sahasında oldukları durum göz önüne alınmıştır.

Savaş sahası için planlanan taktiklerin başarıya ulaşabilmesi ancak birimler arası etkin haberleşme ile mümkündür. Her birim kendi özel görevini yerine getirirken, aynı zamanda, diğer birimleri de çevresel ve taktiksel değişiklikler hakkında bilgilendirmelidir. Bilindiği gibi Kıbrıs barış harekatı sırasında haberleşme sistemlerinin çökmesi sonucu birlikler kendi safındaki birlikleri karşı saftan görüp tahrip etmiştir.

Sahadaki birimlerin sayısı arttıkça emir komuta yapısı da karmaşıklaşır, birimlere özel görevlerin atanması ve komutanlar tarafından birimlerin yönetilmesi zorlaşır. Bu nedenle komutanlar için sahanın genel görünümünü ve özel görevlerin takibini sağlayacak detaylı görünümler sunacak sistemler gereklidir. THSYS bu amaçla geliştirilen,

harp sahası için taktik geliştirme ve gerçek-zamanlı uygulama bilinci sağlayacak bir sistemdir.

Bahsedildiği üzere, mesajlaşma (haberleşme) THSYS'nin en önemli kısmını oluşturmaktadır. Var olan teknolojik altyapıyı kullanarak, sayısal mesajlar, güvenli ve etkin bir şekilde birimler arası iletişimi sağlar. Böylece, askeri taktikler doğru bir şekilde sahaya hızla taşınabilir. THSYS sistemlerinin genel özellikleri şunlardır:

- ön-tanımlı mesaj kalıpları,
- ön-tanımlı yapıda olmayan serbest mesajlaşma desteği,
- mesaj geçerliği için başlangıç ve bitiş tarih-zaman dilimleri belirleme imkanı,
- hava destekli fotoğraf gibi mesaj ekleri,
- saha ve olay görüntüleme amaçlı coğrafik bilgi sistemi,
- mesajlar için sayısal kodlama,
- gelen ve giden mesajların kaydedilmesi.

THSYS için model güdümlü yaklaşım hedefimizin iki ana nedeni vardır. Birincisi, THSYS dünyada yoğun olarak ilgilenilmektedir, birçok farklı örnekleri vardır; fakat henüz istenilen seviyede amaçları karşılar hale gelememiştir. THSYS için geliştirilmiş bilinen bir üst-model yaklaşımı yoktur. İkincisi, taktiksel sahanın coğrafi konumuna bağlı olarak, sahada yer alacak birden fazla müttefik saf için ortak bir dil ya da model bilinmemektedir. Safların kullandıkları farklı sistemler müşterek haberleşmeyi sağlayacak uyum paketlerinden yoksundur. Model güdümlü tasarım bu problemlere çözüm arar. Alana özgü temel unsurlar üst modelde tanımlanır. Böylece bu üst modelden yaratılacak modellerin temel unsurları yansıtması garanti edilir ve alt seviye gerçekleme detayları modelden soyutlanmış olur. Model dönüşümleri ile uygulama kodu modelden otomatik olarak üretilir. Böylece alan modelleri asıl tema olur ve üretkenlik, tekrar kullanılabilirlik, taşınabilirlik, müştereklik artar.

Yazının ilerleyen kısımları şu şekilde düzenlenmiştir. THSYS üzerine süregelen çalışmalar Bölüm 2'de tartışılmıştır. Bölüm 3'te ilgilenilen alan incelenmiş, Bölüm 4'te THSYS için dil bilgisi geliştirilmiştir. Bölüm 5'te THSYS için sıfırdan ve ayrımlama yoluyla geliştirilen üst modeller sunulmuştur. Bölüm 7'de geliştirilen yaklaşımlar karşılaştırılmış, Bölüm 8'de ileriye yönelik yapılması planlanan geliştirmeler sunulmuştur.

2 Benzer Çalışmalar

THSYS üzerine olan ilgi, bu konuda çalışan şirketler göz önüne alındığında, oldukça yoğundur. Çalışmaların odak noktası alt görevlerin otomatik ve daha hızlı yerine getirilmesidir. Haberleşme elemanlarının bütünleşmesi ile emir komuta koordinasyonunun sağlanması hedeflenmektedir.

Thales şirketinin Tactical Battlefield Management System (T-BMS) isimli uygulaması emir komuta mesajlarının görsel dağılımı ve güvenli veri haberleşmesi üzerine kurulmuştur [1].

TROP taktiksel seviyede saha yönetim sistemine bir diğer örnektir. Temel işlevleri, görev hazırlığı, emir, komuta, haberleşme ve veri paylaşımıdır. Uygulamanın hali hazırda bulunan donanım ve yazılım paketleri ile uyumlu olduğu ileri sürülmektedir [2].

SitaWare saha yönetim sistemi, saha komutanlarının ve birlik komutanlarının sahadaki olayları gözlemlemelerine yardımcı olmak amacıyla piyasaya sürülmüştür. Diğer bir amacı da sahadaki alt birimlerin koordinasyonu ve sahanın geneli hakkında bilinçlendirilmesidir [3].

Çatışma yönetim sistemi, taktiksel komuta ve kontrollerin en üst birimden en alt birimlere kadar çift yönlü olarak iletilmesini sağlar [4].

BattleHawk çatışma yönetim sistemi, "Vehicle Platform" ve "Infantry Digital Soldier System" gibi yazılımlarla birlikte, haberleşme, seyrüsefer ve güvenlik bakımından harp sahalarının kolayca yönetilmesini amaçlar [5].

Endüstriyel uygulamalara ek olarak, çeşitli araştırma merkezlerinde de benzer sistemler geliştirilmiştir. Araştırmaların odağı ortak çatışmalar için sefer planlama, yürütme, koordinasyon yönlendirmedir. Örneğin, Amerika Birleşik Devletleri'ndeki "Tactical Battle Management Core Systems (TBMCS)" sahadaki savaş uçaklarının, hava yakıt ikmal tankerlerinin, helikopterlerin, insansız hava araçlarının ve güdümlü füzelerin eş-güdümlü hareketini sağlamak amacıyla geliştirilmiştir. TBMCS başlangıçtaki geniş amaçlı sunucu-istemci uygulaması halinden, daha gelişmiş internet tabanlı ticari bir şirket haline gelmiştir [6].

Araştırmacılar herkesçe anlaşılabilir bir Savaş Yönetim Dili "Battle Management Language (BML)" geliştirmiştir. BML askeri harekattaki

iletişim dilinin standartlaştırılması yönündeki ilk adımlardandır. Bu dille taktiksel sahadaki aktif birimlerin emir komutası sağlanır [7]. "Coalition Battle Management Language (C-BML)" dili "Simulation Interoperability Standards Organization" tarafından geliştirilmiştir ki bu dil hem canlı birimlerin hem de insansız birimlerin benzetim ve gerçek-zamanlı işleyişi için ortak bir dildir [8].

[9] bir grup askerden oluşan bir askeri birim için model önermiştir. Bu modelde dağıtık süreçlerin eş-zamanlı olarak sürdürülebilirliği esas alınmıştır.

[10]'da model güdümlü bir taktiksel hava sahası yönetim sistemi "JSSEO" geliştirilmesi incelenmiştir. JSSEO, OMG'nin "Model Driven Architecture (MDA)" yöntemini kullanarak platformdan bağımsız bir model sunar. JSSEO'nun dayandığı ortak süreç mimarisi müttefiklerin kullandığı farklı sistemlerin bütünleşmesi problemini daha kontrol edilebilir bir hale getirir.

Özetle, endüstriyel saha yönetim sistemleri şirketlerin kendi ihtiyaçlarını esas alarak mevcut yazılımların ve dillerin taktiksel sahada kullanılmak üzere birleştirilerek ortak bir dil oluşturmak çabasına dayalıdır. Fakat bütün saha yönetim modellerini veya dillerini kapsayan, hepsinin temel aldığı bir üst-model hala üretilememiştir. A ve B ülkelerinin kullandığı farklı iki taktiksel yönetim sisteminin ortak bir harekatta tümleşik şekilde kullanılması problemi şirketler bazında tam olarak çözülmüş değildir. Taktiksel saha yönetimi üzerine çalışan bağımsız araştırmacılar ise ortak dil geliştirilmesi üzerine çalışmış ve uzun yıllar dil bilgisi tabanlı yöntemler denenmiştir. Böylece dil bilgisine dayalı birçok araç geliştirilmiş, özel amaçlı dillerin yazılabildiği ve kullanıcı tarafından tanımlanan kurallara göre test edilebilen editörler tasarlanmıştır. Bölüm 4'te, bu araçlardan açık kaynak kodlu Eclipse ortamında çalışabilen XText yazılımı kullanılarak THSYS için geliştirilen dil bilgisi verilmiştir. İlerleyen bölümlerde ise dil bilgisine değil üst-modele dayalı yöntemler verilmiş, dil bilgisi yöntemine göre üstünlükleri tartışılmıştır.

3 Alan Analizi

Askeri birimler hiyerarşik yapıdadır. Bu yapıda emirler üst birimlerden alt birimlere, raporlar da alt birimlerden üst birimlere iletilir. Bu çift yönlü haberleşme yapısı sayesinde emir komuta zincirindeki her birim harekat hakkında yalnızca ana görevin kendisini ilgilendiren kısmını bilir ve yürütür. Bu şekilde bir haberleşme geleneksel radyo, telsiz ağı ile sağlanamaz. Bölüm 1'de de sıralanan temel ihtiyaçları karşılayacak iyi tanımlanmış bir harekat yönetim sistemine ihtiyaç vardır. Bu ihtiyaçlar özetle şunlardır: Emirlerin daha hızlı iletilmesi ve yürütülmesi, böylece ana görevin çabucak tamamlanması, tüm birimlerinin anlayacağı ortak bir dil tanımlanması, görev alanlarının belirlenip görevlerin sorumlulara iletilmesi, görev tatbik durumunun (rapor) otomatik olarak güncellenmesi, alarm durumlarında ilgili birimlerin derhal haberdar edilmesi ve emir komutaların müttefik çatışma durumları için bütünleşmesi.

Bizim harp sahası modelimizde, sahada ana unsur olarak bir çatışma (combat) belirli bir zamanda olur. Bir ana görev (mission) ve bu ana görevin özel parçaları (task) vardır. Çatışmada sıfır ya da daha fazla sayıda birlik yer alır. Her birliğe bir özel görev atanır. Her birliğin bir komutanı (commander) ve sıfır ya da daha fazla sayıda askeri (trooper) vardır. Bir birliğe teçhizat (equipment) olarak araç (vehicle), silah (weapon) ve mühimmat (ammunition) tahsis edilebilir. Birliğin kullandığı bir iletişim sistemi (message management system) vardır ve birlikler arası haberleşme bu sistem ile sağlanır. İleti (message) bir emir (order) veya rapor (report) olabilir. İletiler çeşitli yapılarda olabilir. Alan elemanlarına dair terimler sözlüğü Çizelge-1'de sunulmuştur.

Çizelge-1: Terimler sözlüğü

Terim	Açıklaması
Çatışma (combat)	Bir grup birliği içeren askeri çatışma.
Birlik (troop)	Bir grup asker ya da askeri birim.
Ana görev (mission)	Çatışmadaki birliklerinin genel görevi.
Özel görev (task)	Ana görevin özel bir parçası.
Teçhizat (equipment)	Birliğe tahsis edilen kaynak.
Personel (personnel)	Bir komutan veya bir grup asker.
İletişim sistemi (Message Management System)	Bir birliğe ait giden, gelen iletilerin yapısını belirleyen, yönlendiren ve kaydeden sistem.
Konum (Location)	Dünya üzerinde bir noktayı belirten enlem, boylam çifti.
İleti (Message)	Özel yapıda emir, komuta mesajları.
Emir (Order)	Alt birimlere görev bildiren ileti yapısı.
Rapor (Report)	Alt birimlerin üst birimlere gönderdiği ileti yapısı.

4 Dilbilgisi Tanımlama

Bu bölümde, alan olarak seçtiğimiz THSYS için tanımladığımız dil bilgisi sunulmuştur. Amaç, bütün birimlerce anlaşılabilecek ortak bir dil geliştirmektir ki emirler hızlı bir şekilde yürütülebilsin ve saha içi bilgi akışı hızlı, güvenli ve doğru bir şekilde sağlansın. Önerilen dil bilgisi BNF (Backus-Naur-Form) üst dil yapısında tasarlanmıştır. Geliştirilen dilin kısa bir bölümü aşağıda örnek olarak sunulmuştur.

<Combat> ::= 'Combat' <Name>
<operationDate> <Mission> <Troops>
<Troops>::=<Troop>|<Troops>
<Troop>::= 'Troop' <Name> <Location> <Task> 'Commander' <Personnel> 'Trooper' <Personnels> 'Equipment' <Equipments> <MsgManSys> | <Troops>

Önerilen dil bilgisi BNF olarak tanımlandıktan sonra Eclipse XText eklentisi kullanılarak dil kuralları XText ile yazılmış, çeşitli ek kısıtlamalar eklenmiş ve otomatik olarak dil editörü üretilmiştir. Bu editör ile kullanıcı kendi uygulamasını rahatlıkla yazabilir ve dil bilgisine uymayan imla hataları anında kullanıcıya bildirilir. Kullanılan XText kodunun bir kısmı aşağıda örnek olarak sunulmuştur.

```
Combat :"Combat" name=ID
            operationDate = DateTime
            mission = Mission  (troops += Troop)* ;
Troop :    "Troop" name=ID
            location = Location tasks += Task+
            "Trooper" trooper+=Personnel+
            "Commander" commander=Personnel
            ("Equipment" equipments +=
            (Vehicle | Weapon | Ammunition))*
            msgManSys = MsgManSys
            (subTroops += Troop)*;
```

XText ile ek kısıtlar iki farklı dilde tanımlanabilir. Birincisi "Object Constraint Language (OCL)" türevi bir dil olan "Check Language", diğeri ise Java. Bizim tercihimiz Java olmuştur. Aşağıda, kısıtı gerçekleyecek olan Java kodunun Eclipse XText ortamında nasıl etkinleştirileceğini açıklayan örnek sunulmuştur.

```
fragment=validation.JavaValidatorFragment{…
            composedCheck = "MyDslJavaValidator"}
```

Dil editöründe kullanıcıyı hatalı girdilere karşı kısıtlayacak olan kod, "MyDslJavaValidator.java" dosyasına yazılmış ve kısıtların nasıl girildiğine dair örnek aşağıda sunulmuştur.

```
public void checkLatitude(Latitude latitude) {
        if( latitude.getDegrees() <0
        || latitude.getDegrees() >90){…
```

5 THSYS için Üst Model Tanımlama

Bu bölümde, THSYS için Bölüm 3'te belirtilen ihtiyaçları karşılamak üzere yeni üst modeller tasarlanmıştır. İki farklı yöntem kullanılmıştır. İlk olarak sıfırdan EMOF üst-üst modeline dayanan bir üst model, ikinci olarak da var olan UML üst modelinin ayrımlama yöntemi (profiling) tasarlanması yöntemleri ele alınmıştır. Her iki üst model için soyut sözdizimi modeli, somut sözdizimi modeli ve anlamsal kurallar tanımlanmıştır. Üst modellere uyumlu örnek modeller sunulmuştur.

5.1 Sıfırdan Üst Model Tanımlama

THSYS için sıfırdan bir üst model EMF (Eclipse Modeling Framework) kullanılarak tanımlanmıştır. Bu üst model Eclipse için geliştirilen Java tabanlı bir MOF olan Ecore üst üst modeline dayanır ve Ecore model olarak geçer [11]. Bir sonraki bölümde bu üst modelin temelini oluşturan soyut sözdizimi modeli verilmiştir.

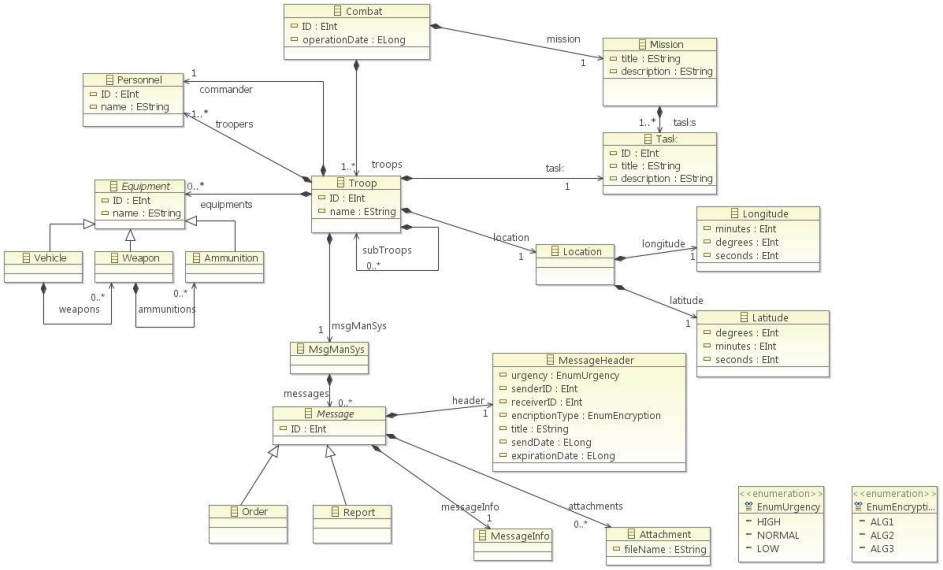
5.1.1 Soyut Sözdizimi

Ecore kullanılarak geliştirilen soyut sözdizimi modeli Şekil 1'de verilmiştir. Modelin tasarımında kullanılan gösterim UML'e benzer ve kendini açıklayıcı şekildedir. Okuyucunun temel UML gösterimine hakim olduğu düşünülerek modelde kullanılan gösterimle ilgili daha fazla açıklama yapılmaya ihtiyaç görülmemiştir.

Çizelge-2'de üst model elemanları ve ilintili oldukları MOF elemanları verilmiştir.

Çizelge-2: Üst modelin MOF ile bağlantısı

TBMS Element	MOF Element
TBMS::Combat	MOF::Class
TBMS::Troop	MOF::Class
TBMS::Personnel	MOF::Class
TBMS::MsgManSys	MOF::Class
TBMS::Attachment	MOF::Class
TBMS::MessageInfo	MOF::Class
TBMS::MessageHeader	MOF::Class
TBMS::Equipment	MOF::Interface
TBMS::Message	MOF::Interface
TBMS::Location	MOF::Interface
TBMS::EnumUrgency	MOF::DataType
TBMS::EnumEncryptionType	MOF::DataType



Şekil-1: Sıfırdan oluşturulan üst model

5.1.2 Somut Sözdizimi

Önceki bölümde oluşturulan Ecore modelinin kullanıcıya sunumunda kullanılacak olan somut gösterim modelini oluşturmak için Eclipse'in GMF (Graphical Modeling Framework) aracı kullanılmıştır. Bu aracın kullanılmadığı durumlar için EMF öntanımlı bir ağaç yapısı gösterim modeli sunar. GMF ile hazırladığımız somut sözdizimi bir örnek model ile Şekil-2'de gösterilmiştir. Hazırladığımız somut sözdizimi gerçek hayatta kullanılan simgelerden oluşmaktadır. Böylelikle kullanılan somut gösterim kendi kendini açıklar niteliktedir. Simgelerin model elemanı karşılıkları Çizelge-3'te verilmiştir. Örnek modele ilişkin ağaç gösterimi Bölüm 5.1.4'te verilmiştir.

5.1.3 Statik Anlamsal Kurallar

Şu ana kadar THSYS'de kullanılan elemanların ve bu elemanların birbirleri ile olan ilişkilerinin yer aldığı soyut sözdizimi modeli ve bu modelin kullanıcıya sunulacak olan somut sözdizimi modeli tanımlanmış oldu. Soyut modelde elemanlar arası ilişkiler sayısal bağlamda kısıtlanmaktadır. Örneğin her birliğin bir adet özel görevi olmak zorundadır veya bir birliğin birçok aracı olabilir. Bu bölümde üst modelimiz için soyut modelde getirilemeyen kısıtlara yönelik bazı statik anlamsal kurallar tanımlanmıştır. Eclipse ortamı bu kuralların iki farklı şekilde tanımlanmasına imkan sağlar. Birincisi geleneksel yöntem olan OCL (Object Constraint Language) kullanımı, ikincisi de özel Java fonksiyonlarının soyut model ile birleştirilmesi yöntemidir. Bu çalışmada ikinci yöntem tercih edilmiştir.

```
/* valueCheck constraint of 'Location'.
* @generated NOT
*/
public boolean validateLocation_valueCheck(
Location location, DiagnosticChain diagnostics,
Map<Object, Object> context){
  boolean latd, longd, llmin, llsec;
  latd=location.getLatitude().getDegrees()>-90
  && location.getLatitude().getDegrees()<90;
  ...
  if (!latd || !longd || !llmin || !llsec) {
    if (diagnostics != null) {
      diagnostics.add(
            createDiagnostic( Diagnostic.ERROR,
             DIAGNOSTIC_SOURCE,0,
            "_UI_GenericConstraint_diagnostic",
            new Object[] { "valueCheck",
            getObjectLabel(location, context) },
            new Object[] { location },context));   }
    return false;   }
  return true;  }
```

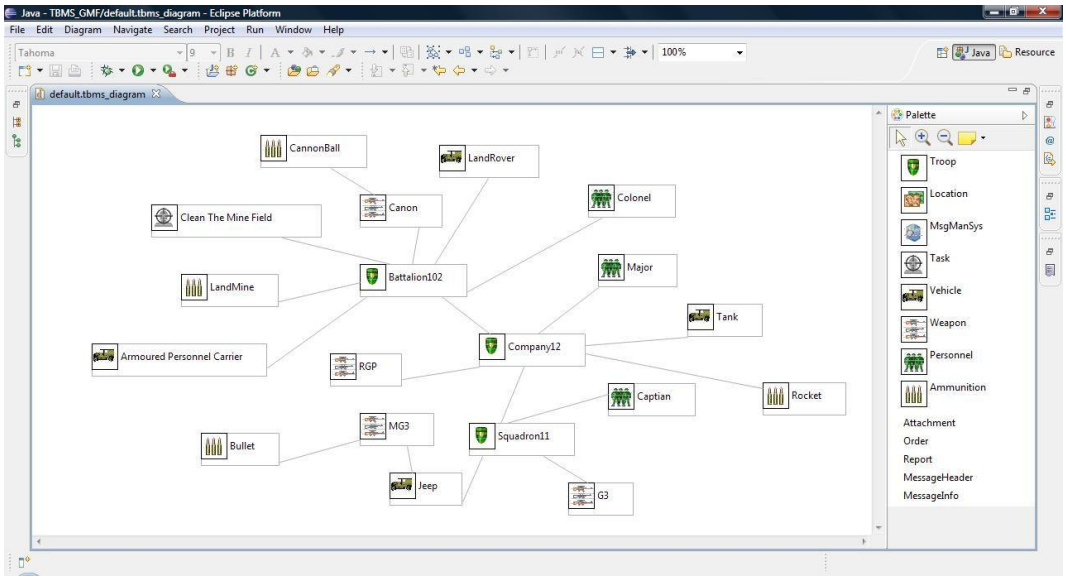
Çizelge-4'te üst modelimize eklenen statik kurallar verilmiştir. Modelin bu kurallara uyum testi Java fonksiyonları olarak gerçeklenmiş, örnek bir Java fonksiyonu yukarıda verilmiştir. Kuralların üst modele bağlantısı Ecore'un sunduğu "EAnnotation" ile yapılmıştır. Böylece Java fonksiyonlarımız model yaratılma esnasında otomatik olarak model editör tarafından çağrılmaktadır.

Çizelge-3: üst modelin somut sözdizimi elemanları

Icon	Metaclass	Icon	Metaclass
	Combat		Mission
	Task		Troop
	Location		Latitude
	Longitude		Ammunition
	Vehicle		Weapon
	MessageHeader		MessageInfo
	Order		Report

Çizelge-4: üst model kısıtları

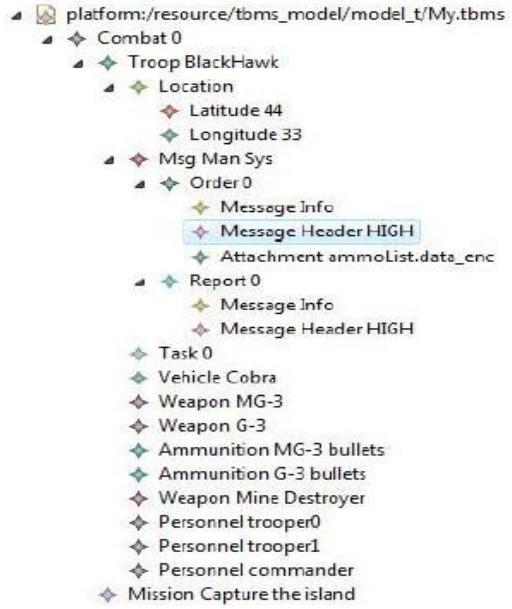
THSYS elemanı	Kısıt
Troop → name	Birliğe bir isim atanması zorunludur.
Personnel → name	Personele bir isim atanması zorunludur.
MessageHeader → sendDate	İleti gönderme zamanı, iletinin gerçerliğinin bitişinden önce olmalıdır.
Equipment → name	Techizata bir isim atanması zorunludur
Latitude→degree	Enlem derecesi $[-90, 90]$ aralığında olabilir.
Latitude→minutes	Enlem dakikası $[0, 60)$ aralığında olabilir.
Latitude→seconds	Enlem saniyesi $[0, 60)$ aralığında olabilir.
Longitude→degree	Boylam derecesi $[-180, 180]$ aralığında olabilir.



Şekil-2: özel somut sözdizimi ile hazırlanan model

5.1.4 Örnek Model

Sıfırdan yaratılan üst modele ilişkin örnek model ağaç yapısında ve XML düzeninde olmak üzere iki ayrı somut sözdizimi modeli ile gösterilmiştir. Daha önce tanımladığımız özel somut modeli gösterimi için bkz. Şekil-2. Şekil-3'te örnek modelin ağaç yapısı gösterimi sunulmuştur. Örnek model tamamen yapay bir senaryo üzerine kurulmuştur. Küçük bir çatışma belirli bir bölgede ve tarihte geçer. Birtakım teçhizatlara sahip birlik ana bir görevin (mission) bir parçasını (task) üstlenir. Çatışma sırasında birimler arasında çeşitli emir ve rapor mesajları iletilir.



Şekil-3: Sıfırdan tanımlanan üst model için model ağaç yapısı gösterimi

5.2 Genişletme ile Üst Model Tanımlama

Hafif genişletme yöntemi var olan bir üst modele biçim eklenerek yapılır. Bu bölümde UML üst modeline amacımıza uygun biçimler tanımlayarak tasarladığımız THSYS üst modeli anlatılmıştır. Biçimlerin kendi değişkenleri, diğer deyişle etiketlenmiş değişkenleri olabilir. Bu çalışmada UML sınıf üst modeli Eclipse ortamında EMF 2.5 ve UML2.2 araçları kullanılarak biçimlendirilmiş ve amaca uygun olarak bazı etiketlenmiş değişkenler tanımlanmıştır.

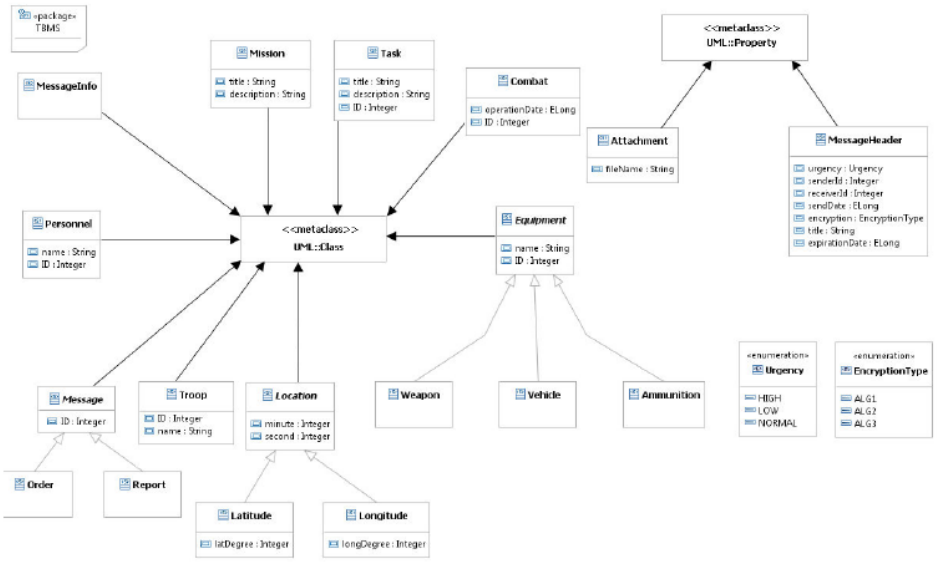
5.2.1 Soyut Sözdizimi

THSYS için genişletme yöntemiyle tasarladığımız üst model Şekil-4'te gösterilmiştir. Daha önce sıfırdan üst model tanımlanması bölümünde anlatılan ihtiyaçlar aynen dikkate alınmıştır.

Üst modelimizin elemanları ve karşı düşen UML üst model elemanları Çizelge-5'te verilmiştir. Kullandığımız sınıf elemanlarının değişkenleri ise "UML::Property" üst model elemanına bağlıdır.

Çizelge-5: üst model kısıtları

Stereotype	Metaclass
Combat	UML::Class
Task	UML::Class
Mission	UML::Class
MessageInfo	UML::Class
Personnel	UML::Class
Message	UML::Class
Troop	UML::Class
Location	UML::Class
Equipment	UML::Class



Şekil-4: Var olan UML üst modelinden genişletme yoluyla elde edilen THSYS üst modeli

5.2.2 Soyut Sözdizimi

Ele alınan genişletme yöntemi UML üst modeline dayandığından burada kullanılan somut gösterimler UML somut gösterimiyle birebir örtüşür. UML somut gösterimi kullanıcılar tarafından iyi bilindiği düşünülerek yeni bir somut gösterim modeli tanımlanmamıştır.

5.2.3 Statik Anlamsal Kurallar

Genişletilmiş üst modelimize dair tanımlanan kurallar önceki bölümlerde anlatılan sıfırdan üst model tasarlanmasında kullanılan kurallar ile aynıdır.

5.2.4 Örnek Model

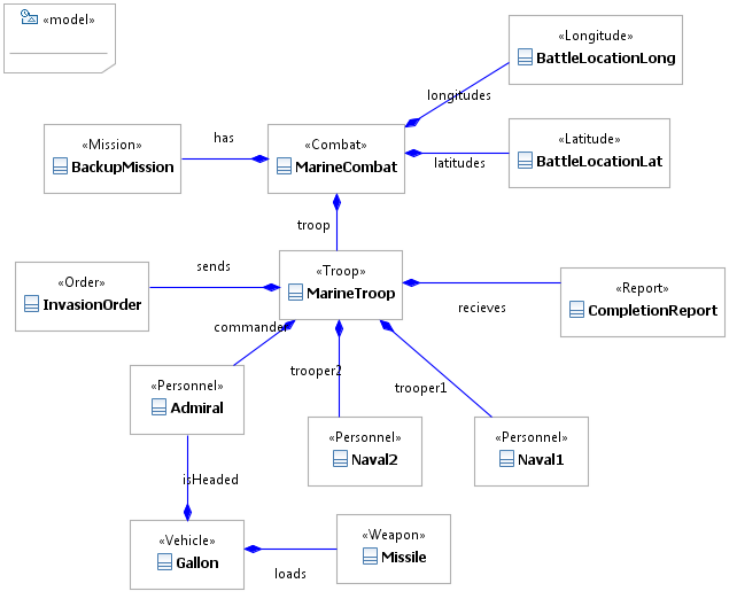
Şekil-5'te genişletilmiş üst modelimiz ile uyumlu olacak şekilde tasarlanmış örnek bir model sunulmuştur. Bu modelin somut gösterimi UML gösterimi ile aynıdır. Örnek model Bölüm 5.1.4'te verilen modeldekine benzer bir senaryo içerir ancak askeri birlik olarak deniz kuvvetleri kullanılmıştır.

6 Model Dönüşümleri

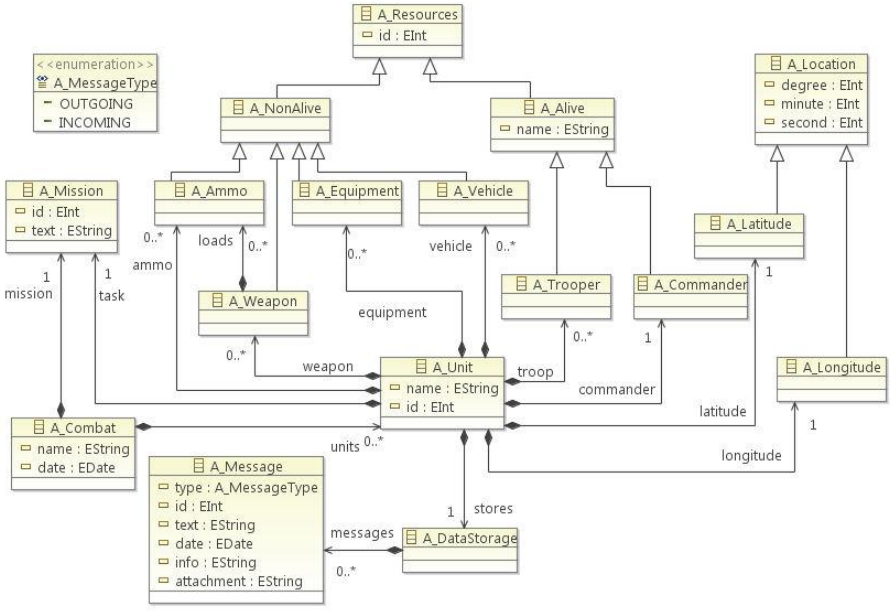
Model dönüşümü MDSD'nin en büyük faydalarındandır. Üst modele dayalı modeller arası dönüşüm yapılabilir. MDSD'de modeller yazılım tasarımı ve kod üretimi için asıl kaynaktır. Böylece, tasarım aşaması alt seviye programlama detaylarından soyutlanır. Model geliştirme aşaması zahmetli olsa da model dönüşümleri sayesinde yazılım geliştirme için gerekli toplam süre azalır ve üretkenlik, tekrar kullanılabilirlik, üretkenlik, müştereklik gibi unsurlar kaliteyi artırır.

Model-model dönüşüm için kullanılabilecek birçok araç vardır. Atlas dönüşüm dili (ATL), işlevsel ve ilişkisel sorgu bakış dönüşümleri (QVTO, QVTR) bunlardan birkaçıdır. Bu çalışmada, ATL kullanılarak yine bizim oluşturduğumuz Müttefik (Alliance)-THSYS (A_TBMS) modelinden TBMS modeline dönüşüm gösterilmiştir. A_TBMS müttefik çatışmalar için kullanılan daha kapsamlı bir üst modele dayanır. A_TBMS üst modelinin Ecore diyagramı Şekil-6'da gösterilmiş, dönüşüm için kullanılan örnek A_TBMS modeli Bölüm 6.1'de verilmiştir.

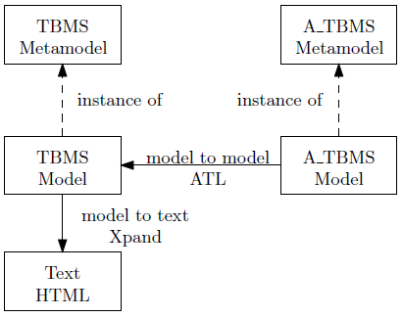
MDSD bakış açısıyla, model-model ve model-metin (kod) dönüşümleri aynıdır. Her ikisi de model dönüşümü olarak değerlendirilir çünkü kod da sistemin bir modelidir fakat kodun temel aldığı bir üst modelin tanımlanması gerekmez. MDA (Model Driven Architecture) yaklaşımı ile birlikte kod üretimi daha popüler olmuş, çeşitli araçlarla desteklenmiştir. Bu çalışmada, model-kod dönüşüm araçlarından en yaygın olanı Xpand kullanılmıştır. Kullanılan model dönüşümleri Şekil-7'de verilmiş, ileride detaylı olarak anlatılmıştır.



Şekil-5: Genişletilmiş üst modele uyumlu örnek model



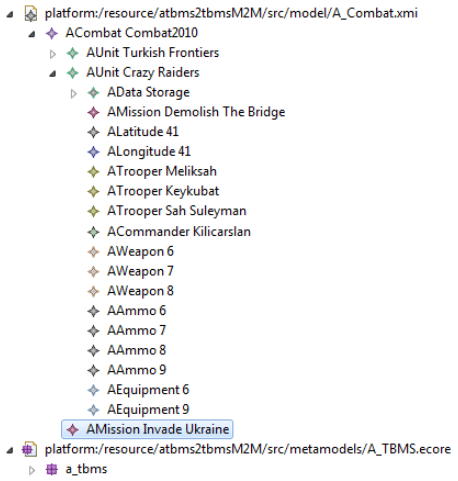
Şekil-6: Müttefik THSYS (A_TBMS) üst modeli



Şekil-7: Müttefik THSYS (A_TBMS) üst modeli

6.1 Model-Model Dönüşümü

Bölüm 6'da verilen üst modelden örnek bir A_TBMS modeli oluşturulmuş, ağaç yapısındaki Ecore model gösterimi Şekil-8'de verilmiştir.



Şekil-8: Müttefik THSYS (A_TBMS) modeli

A_TBMS modeli kendi üst modelindeki A_Combat elemanından dinamik örnekleme ile yaratılmış, örnek deniz kuvvetleri birimleri ile doldurulmuştur. Böylece TBMS modeline dönüştürülmeye hazırdır. Dönüşüm kuralları doğrudan üst modeller arasında tanımlanır. Her model için ayrıca dönüşüm kuralı tanımlamak gerekmez. Böylece kaynak ve hedef modellerin kendi üst modelleri ile uyumu mutlak olarak sağlanır. Dönüşüm kuralları Eclipse ortamında yaratılan ATL projesi içinde "atbms2tbms.atl" dosyasına yazılmıştır. A_TBMS'nin A_Combat sınıfından TBMS'nin Combat sınıfına tanımlanan örnek bir kural aşağıda verilmiştir.

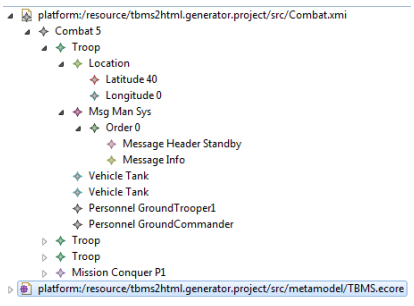
```
module atbms_2_tbms;
create OUT: TBMS from IN:A_TBMS;
rule A_Combat2Combat{
from r1_s: A_TBMS!A_Combat
to r1_t: TBMS!Combat(
ID ← r1_s.name.hashCode(),
operationDate ← r1_s.date.getTime(),
troops ← r1_s.units,
mission ← r1_s.mission)}
```

Bir sonraki bölümde, Eclipse Xpand aracı kullanılarak gerçeklenen model-metin dönüşümü açıklanmıştır.

6.1 Model-Metin Dönüşümü

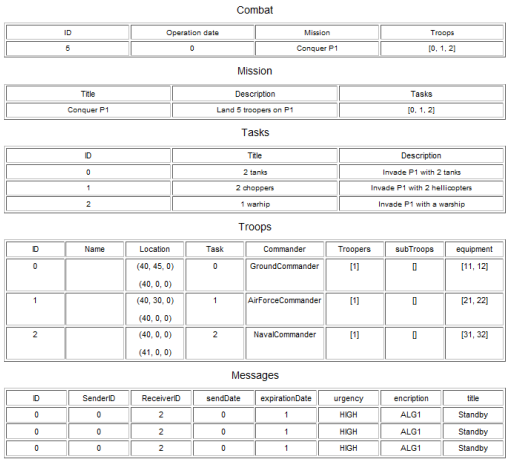
Model-metin dönüşümü model-model dönüşümü ile aynıdır, yalnızca metinin dayandığı bir üst model tanımlanmamıştır. Metin üretimi için genellikle bir şablon metin kullanılır. Bu şablon metnin ilgili kısımları kaynak model elemanları ile doldurulur. Kaynak model Şekil-9'da verilen örnek bir TBMS modelidir. TBMS modeline ait veri dökümünü verecek şekilde bir HTML çıktısı (Combat.html) üretecek olan şablon metnin bir kısmı aşağıda verilmiştir.

```
«IMPORT tbms»
«DEFINE main FOR Combat»
«FILE "Combat.html"»
<HTML><HEAD><TITLE>Combat Scenario</TITLE>
</HEAD> <BODY><P>Mission</P>
<TABLE>«EXPAND tsk FOREACH this.mission.tasks»
</TABLE> </BODY> </HTML>
«DEFINE tsk FOR Task»
<TR><TD><P> «this.id» </P></TD>
<TD><P> «this.title» </P></TD>
<TD><P> «this.description» </P></TD></TR>
«ENDDEFINE»
```



Şekil-9: Metin üretimi için kullanılan TBMS kaynak modeli

Üretilen hedef metin "Combat.html" Şekil-10'da sunulmuştur.



Combat

ID	Operation date	Mission	Troops
5	0	Conquer P1	[0, 1, 2]

Mission

Title	Description	Tasks
Conquer P1	Land 5 troopers on P1	[0, 1, 2]

Tasks

ID	Title	Description
0	2 tanks	Invade P1 with 2 tanks
1	2 choppers	Invade P1 with 2 helliicopters
2	1 warhip	Invade P1 with a warship

Troops

ID	Name	Location	Task	Commander	Troopers	subTroops	equipment
0		(40, 45, 0) (40, 0, 0)	0	GroundCommander	[1]	[]	[11, 12]
1		(40, 30, 0) (40, 0, 0)	1	AirForceCommander	[1]	[]	[21, 22]
2		(40, 0, 0) (41, 0, 0)	2	NavalCommander	[1]	[]	[31, 32]

Messages

ID	SenderID	ReceiverID	sendDate	expirationDate	urgency	encription	title
0	0	2	0	1	HIGH	ALG1	Standby
0	0	2	0	1	HIGH	ALG1	Standby
0	0	2	0	1	HIGH	ALG1	Standby

Şekil-10: Üretilen metin: “Combat.html”

7 Tartışma

THSYS sistemleri alan olarak seçilmiş ve bu sistemler için ortak dil tanımlama amacıyla üç ayrı yöntem izlenmiştir. İlk önce BNF ve XText kullanılarak bir dil bilgisi geliştirilmiştir. BNF kendisi ile tanımlanan bir üst dildir. XText de aynı şekilde kendisi ile tanımlanır ve EBNF’nin özel biçimidir. Böylece “?", “*" ve “+" gibi dil geliştirmeyi kolaylaştırıcı yetenekleri vardır. Ne BNF ne de EBNF için öntanımlı veri tipi, örneğin eşsiz tip anlamına gelen “ID" veri tipi vardır. Bu da soyut sözdizimi modeli için tanımlanması gereken statik anlamsal kuralların tanımlanması ek yükünü getirmektedir. Diğer yandan, XText ile hem öntanımlı veri tipleri kullanılabilir hem de statik anlamsal kısıtlar tanımlamak mümkündür. Ayrıca XText, geliştirilen dil için otomatik olarak bir editör oluşturabilir, böylece geliştirilen dili kullanıcı rahatlıkla kullanarak fakat aynı zamanda da kısıtlara uymak zorunda kalarak kendi dil modelini oluşturabilir.

İkinci yöntem olarak THSYS için sıfırdan bir üst model oluşturulmuştur. Bu yaklaşım için MOF’un özel hali olan Eclipse Ecore üst üst modeli kullanılmıştır. Eclipse EMF, Ecore üst modelinin UML gösterimi kullanan bir görsel arayüz ile tasarlanmasına imkan sağlar. Ecore’un “EAnnotations” elamanları da statik anlamsal kuralların rahatlıkla üst modele eklenmesini sağlar. 2010 yazında kullanıma sunulacak olan EMF eklentisi ile OCL biçiminde kısıt tanımlamak daha da kolaylaşacaktır. Şimdiki son sürüm EMF yalnızca Java fonksiyonları ile kısıt tanımlamaya imkan vermektedir ki bunun tek tek elle yazılması kolay değildir.

Üçüncü yöntem olarak da var olan bir UML üst modeli biçimler tanımlanarak genişletilmiştir. Her sınıf için basit bir biçim tanımlanmış fakat biçimler arasında ilişkiler tanımlanmamıştır. Bu yöntem için de sıfırdan üst model tanımlamada belirtilen kısıt ekleme zorlukları aynen geçerlidir.

Bir alan için bahsedilen bu üç yöntemden hangisinin kullanılması gerektiği ile ilgili olarak bizim çıkarımlarımız şunlardır. Alana özel bir dil (DSL) tanımlanması için en zor yöntem dil bilgisi geliştirme yöntemidir çünkü tüm türetmeye açık ve kapalı (terminal/nonterminal) terimlerin ve özel veri tiplerinin tanımlanması ve dikkatle yazılması gerekmektedir. Bu durum dil bilgisi tanımlama yöntemini daha alt seviye programlama hatalarına açık bir konuma getirmektedir. Fakat dilbilgisi tanımlama yöntemi uzun yıllardır yazılımcılar tarafından kullanıldığından, bu yöntem için gerekli araçlara ulaşmak çok kolaydır. Son yöntem olarak incelenen UML üst modelinin biçimlendirilmesi yöntemi, gerçeklenmesi en kolay olan yöntemdir fakat bu yöntem çok fazla biçim değişikliği gerektirmeyen hafif biçimlendirme durumları için uygundur. Eğer uygulama alanı olarak THSYS gibi büyük bir sistem seçilmişse ve ihtiyaçların karşılanabilmesi için yüksek seviye (ağır) soyutlaştırma gerektiriyorsa, bu durumda izlenmesi gereken yöntem ikinci yöntem olarak sunulan sıfırdan üst model tanımlama yöntemi olmalıdır. Çizelge-6’da kullanılan üç yöntem için bir karşılaştırma sunulmuştur.

Çizelge-6: Alana özel dil geliştirme yöntemlerinin kıyaslanması

Kıstas	Dil bilgisi	Sıfırdan	Genişletme
Bilinirlik	en eski	en yeni	yeni
Araç desteği	çok	az	orta/çok
Soyutlama	orta	en çok	çok
Tasarım kolaylığı	zor	kolay	en kolay
Kullanım kolaylığı	zor	kolay	kolay
Tekrar kullanma	zor	kolay	kolay

8 Sonuç

Bu çalışmada taktiksel harp sahası yönetim sistemi (THSYS) alan olarak seçilmiş, alandaki benzer çalışmalar gözden geçirilip bu alana özel üç farklı dil geliştirme yöntemi incelenmiştir. İlk olarak Eclipse XText aracı kullanılarak dil bilgisi geliştirme yöntemi uygulanmıştır. Daha sonra

model güdümlü yaklaşımlardan, Eclipse Ecore kullanılarak sıfırdan üst model geliştirme yöntemi ve UML biçimlendirme kullanılarak var olan UML üst modelinin hafif biçimlendirilmesi yöntemi uygulanarak THSYS için üst modeller elde edilmiştir. Eclipse GMF aracı ile THSYS için özel olarak hazırlanan somut sözdizimi modeli gerçek hayattan simgeler kullanılarak oluşturulmuş, üst modellere uyumlu örnek modeller sunulmuştur.

Uygulanan üç yöntem karşılaştırılmıştır. Geleneksel dil bilgisine dayalı yöntemin kullanımının zor olduğu ve mevcut ihtiyaçları karşılamak, özellikle ortak dil oluşturmak için yeterli olmadığı sonucuna varılmıştır. Tasarımı alt düzey gerçekleme ayrıntılarından soyutlayabilen ve model dönüşümleri ile alan dillerinin bütünleşmesi için üst düzey bir platform sağlayan model güdümlü yaklaşım incelenmiştir. Yeni geliştirilmekte olan açık kaynak yazılımlar ile desteklenen model güdümlü yöntemlerin zaman içinde daha başarılı sonuçlar vereceği düşünülmektedir.

MDSD'nin en büyük faydalarından biri model dönüşümüdür. Bu çalışmada sunulan model-model ve model-metin dönüşümleri ile model güdümlü yaklaşımda modellerin tasarımın çekirdeğini oluşturduğu vurgulanmış, böylece üretkenlik, tekrar kullanılabilirlik, taşınabilirlik, müştereklik gibi kalite beklentilerinin nasıl karşılandığı anlatılmıştır. Model dönüşümü için kullanılan ATL ve Xpand araçlarının dil bilgisi, üst model ve model oluşturmak için kullanılan araçlar kadar tatmin edici olduğu sonucuna varılmıştır.

9 Teşekkür

Dr. Bedir Tekinerdoğan'a değerli katkılarından dolayı teşekkür ederiz.

10 Kaynakça

[1] Thales, 2010. Tactical battlefield management systems: T-BMS, comm@nder. Kaynak: http://www.thalesgroup.com

[2] M. Suppliers and News, 2010. Tactical battlefield management systems - TROP. Kaynak: http://www.armedforces-int.com/article/tactical-battlefield-management-systems-trop.html

[3] Systematic, 2010. Sitaware battle management. Kaynak: http://www.systematic.com

[4] B. Systems, 2010. Battle management system. Kaynak: http://www.baesystems.com

[5] R. Blobalnet, 2010. BattleHawk combat management system. Kaynak: http://www.rfglobalnet.com/product.mvc/Combat-Management-Systems

[6] L. Martin, 2010. Tactical battle management core systems (TBMCS). Kaynak: http://www.deagel.com/Display-and-Network-Systems/TBMCS_a001286001.aspx

[7] W. P. Sudnikovich, J. M. Pullen, M. S. Kleiner ve S. A. Carey, 2004. "Extensible battle management language as a transformation enabler", sayı 80, no.12, sayfa 669--680.

[8] SISO, 2010. C-BML product development group. Kaynak: http://www.sisostds.org

[9] A. Bordertsky, R. B. L. Duffy, E. Bach ve C. Oros, 2004. "A proposed model of battle rhythm at the tactical level", 9. International Command and Control Research and Technology Symposium, San Diego, CA.

[10] R. W. Jacobs, 2004. "Model-driven development of command and control capabilities for joint and coalition warfare", 9th International Command and Control Research and Technology Symposium, San Diego, CA.

[11] The Eclipse Modeling Framework (EMF) overview, 2010. Kaynak: http://help.eclipse.org