

## UML TABANLI BİR METAMODELLEME ALTYAPISI İLE GÖREV UZAYI KAVRAMSAL MODELİ GELİŞTİRME

**N. Alpay KARAGÖZ<sup>(a)</sup>, Onur DEMİRÖRS<sup>(b)</sup>**

<sup>(a)</sup> Bilgi Grubu Ltd., ODTÜ Teknokent Gümüş Blok. No:3, 06531, Ankara, [alpay@bg.com.tr](mailto:alpay@bg.com.tr)

<sup>(b)</sup> Orta Doğu Teknik Üniversitesi, Enformatik Enstitüsü, 06531, Ankara, [demirors@metu.edu.tr](mailto:demirors@metu.edu.tr)

### ÖZET

Kavramsal modelleme simülasyon geliştirmenin ayrılmaz bir parçası ve simülasyon gereksinimlerini anlamayı kolaylaştıran verimli bir araç olarak bilinmektedir. Simülasyon sistemleri geliştirme yaşam döngüsünde kavramsal modelleme faaliyeti, problem uzayının daha iyi anlaşılması ve problem analizi aşamasında etkin bir yöntem sunmasına rağmen simülasyon dünyasında bu aracın etkin kullanımını destekleyecek altyapılar yaygın değildir. Benzer şekilde modeller yazılım geliştirme yaşam döngüsünde önemli bir yere sahiptir ancak simülasyon sistemlerinden farklı olarak yazılım geliştirmede modelleme dilleri ve bunları destekleyen araçlar yaygın olarak kullanılmaktadır. Yazılım geliştirmede UML (Birleşik Modelleme Dili) en yaygın kullanılan modelleme dillerinden biri olmanın ötesinde sunduğu metamodelleme mimarisi ile alana özgü modelleme dillerinin geliştirilmesinde önemli bir yere sahiptir.

Bu bildiride, simülasyon geliştirme yaşam döngüsünde görev uzayı kavramsal modelleme sürecini destekleyen UML tabanlı bir metamodelleme altyapısı tanıtılmıştır. Böylece simülasyon yazılımlarının geliştirilmesinde yaygın olarak kullanılan UML ile benzer bir yapıya sahip bir kavramsal modelleme dili geliştirilerek modeller arası geçişlere olanak sağlayacak altyapı da oluşturulmuş olacaktır.

**Anahtar Kelimeler:** görev uzayı kavramsal model, metamodel, simülasyon

### A UML-BASED METAMODELING FRAMEWORK for MISSION SPACE CONCEPTUAL MODEL DEVELOPMENT

Conceptual models of the mission space are an essential part of simulation development and provide an effective way to understand the simulation requirements. Although the conceptual analysis activities performed during simulation development lifecycle are effective means in capturing the mission space, the simulation world lacks the frameworks that support the widespread use of these means. Similarly, models play an important role in the software development lifecycle; however modeling languages and tools that support these languages are extensively used. UML (Unified Modeling Language) is not only one of the most commonly used modeling languages,

but also has an important role in the development of domain specific modeling languages by its metamodeling architecture.

In this paper, a UML based metamodeling framework that supports the development of mission space conceptual model in the simulation development lifecycle is introduced. Our aim is to provide a conceptual modeling language based on UML, which has a widespread use in the simulation development projects, and establish an infrastructure that allows the transformation among models at different abstraction levels.

**Keywords:** mission space conceptual model, metamodel, simulation

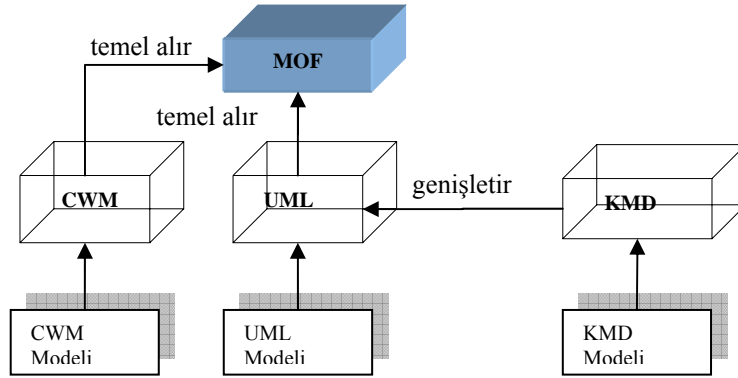
## 1. GİRİŞ

Kavramsal modelleme simülasyon geliştirmenin ayrılmaz bir parçası ve simülasyon gereksinimlerini anlamayı kolaylaştıran verimli bir araç olarak bilinmektedir. Simülasyon sistemleri geliştirme yaşam döngüsünde kavramsal modelleme faaliyeti, problem uzayının daha iyi anlaşılması ve problem analizi aşamasında etkin bir yöntem sunmasına rağmen simülasyon dünyasında bu aracın etkin kullanımını destekleyecek altyapılar yaygın değildir. Benzer şekilde modeller yazılım geliştirme yaşam döngüsünde önemli bir yere sahiptir ancak simülasyon sistemlerinden farklı olarak yazılım geliştirmede modelleme dilleri ve bunları destekleyen araçlar yaygın olarak kullanılmaktadır.

Simülasyon projelerinde görev uzayı kavramsal modeli geniş bir tanımla; işlevsel ve davranışsal yetenekleri tanımlayan, kullanıcılar ve geliştiriciler için ortak bir dil tanımlayan, gerçekleştirimden bağımsız bir gösterim sunan, varlıkları, ilişkileri ve etkileşimleri belirten, mantıksal kararları ve algoritmaları içeren, daha detaylı modellere dönüşüm için gerekli tanımları barındıran, doğrulama ve geçişleme etkinliklerinde kullanılabilen bir tanımlar bütünüdür [1].

Simülasyon geliştirme süreci simülasyon hedeflerinin belirlenmesi ile başlar ve görev uzayının tüm ayrıntılarıyla tanımlanmasına kadar sürer. Sonraki adımlar simülasyon sistemi yazılımının geliştirilmesi olacağı için görev uzayı kavramsal modelleri olabildiğince formal gösterim yöntemleri kullanılarak tanımlanmalı ve modeller arası geçişlere olanak sağlanmalıdır. Ancak gösterim yöntemleri tanımlanırken alan uzmanları tarafından da anlaşılabilir olması kısıtı dikkate alınmalıdır.

Metamodel en geniş tanımıyla bir modelleme dilinin modelidir [2]. Tanımdan da anlaşılacağı üzere metamodel de bir model olmasına rağmen temel iki farklılık vardır. Öncelikle metamodel tanımladığı dilin sözdizimsel ve anlambilimsel kuralları gibi temel özelliklerini içermelidir. İkinci olarak metamodel bir metamodel mimarisinin parçası olmalıdır. Böylelikle metamodel de bir model olarak ele alınabilir. Örneğin bir modelleme dili olan UML [3], yine OMG (Object Management Group) tarafından geliştirilmiş olan MOF (Meta Object Facility) [4] metamodelleme mimarisi içinde yer almaktadır. Bu mimari Şekil 1'de gösterilmiştir.



**Şekil 1.** Meta-metamodel, modelleme dili ve model

UML, önceleri yazılım geliştirme dünyasına bütünleşik bir modelleme dili sunma amacıyla sunulmuştu. Birbirinden çok farklı alanlarda farklı problemlere tek bir dilin yetmediği görülünce UML yaklaşımı evrensel bütünleşik bir modelleme dilinden uzaklaşıp, alana özgü modelleme dillerinin tanımlanabileceği bir modelleme dilleri ailesine dönüştü. Şekil 1’de görüldüğü gibi MOF’u esas olarak tanımlanan bir dil olan UML’e benzer olarak OMG tarafından geliştirilen CWM (Common Warehouse Metamodel) [5] veya bu çalışmada tanıtilen KMD (Kavramsal Modelleme Dili) gibi dillerin de tanımlanabileceği bir metamodelleme mimarisinin bir parçası haline geldi.

Şekil 1’de doğrudan MOF’u esas olarak oluşturulan CWM’nin yanında UML’nin profil mekanizmasının kullanılarak genişletilmesiyle oluşturulan KMD modelleme dili de genişleme mimarisine örnek olarak sunulmuştur.

Metamodelerin karmaşık problemlere çözüm bulmak amacıyla kullanılması da bu şekilde mümkün kılındı. Aynı metamodelleme mimarisinde yer alan farklı modelleme dilleri arasında geçişleri tanımlamak kolaylaştı. Bunun yanında farklı seviyelerde soyutlama gerektiren problem alanları için farklı dillerin tanımlanmasını sağlayan bir altyapı da tanımlanmış oldu.

Bu bildiri öncelikle görev uzayı kavramsal modelinin simülasyon geliştirme sürecindeki yeri açıklanmış, metamodel oluşturma ile ilgili temel bilgiler sunulmuştur. Bildirinin 2. bölümünde UML tabanlı metamodel oluşturma yöntemi açıklanmıştır. 3. bölümde görev uzayı kavramsal modeli oluşturmak için kullanılacak UML tabanlı metamodel altyapısı tanımlanmıştır. 4. bölüm bu altyapının kullanılması ile elde edilecek kazanımları ve diğer sonuçları özetlemektedir.

## 2. UML TABANLI METAMODEL OLUŞTURMA

### OMG tarafından MOF 1.X standartlarında [6] 4 seviyeli olarak tanımlanan metamodelleme mimarisi

Çizelge 1’de özetlenmiştir. Yukarıdaki metamodelleme mimarisinde her bir seviye bir üst numaralı seviyenin bir örneği (instance) olarak düşünülebilir.

**Çizelge 1.** Metamodelleme mimarisi

Kısa Ad	Seviye	Açıklama
M0	Model Örneği	Uygulama örneğini içerir, örneğin bir sınıfın koşturma zamanındaki örneği veya veritabanındaki bir satır. (ör. Nesne)
M1	Model	Uygulamayı içerir, örneğin nesne yönelimli bir sistemdeki sınıflar veya ilişkisel veritabanındaki tablo tanımlamaları. (ör. Kullanıcı Modeli)
M2	Metamodel	Modelleme dilini içeren metamodel, örneğin UML'de tanımlı sınıf, özellik, operasyon gibi model elemanlarının tanımlandığı seviye. (ör. UML)
M3	Meta-metamodel	Tüm metamodelerin sahip olabileceği davranış ve özellikleri tanımlayan seviye. (ör. MOF)

UML tabanlı bir metamodel oluştururken iki temel yol izlenebilir. Birinci yol Şekil 1'de gösterilen örnekte olduğu gibi UML'nin de esas aldığı MOF tanımlarına dayalı bir modelleme dili oluşturmaktır. Bu şekilde daha çok esneklik sunma olanağı doğarken, modelleme dilinin tanımlanması için daha fazla işgücü harcamak gerekebilir.

Diğer yöntemde ise UML ve genişleme mekanizması olan profiller esas alınarak bir modelleme dili tanımlanır. Bu yöntem UML'yi genişletme olarak tanımlanabileceğinden daha üst seviye yapılar üzerinde ve daha az işgücü ile modelleme dilinin oluşturulmasına olanak tanırken bazı UML kısıtlarının da oluşturulan modelleme diline aktarılması ile sonuçlanabilir.

Bu çalışmada UML'nin profil kullanılarak genişletilmesiyle bir görev uzayı kavramsal model oluşturma metamodeli oluşturulmuştur.

### **3. GÖREV UZAYI KAVRAMSAL MODEL OLUŞTURMA METAMODELİ**

Görev uzayı kavramsal modellerinin alan uzmanları tarafından kolaylıkla geliştirilebilmesi için alan uzmanlarının diline yakın modelleme ve gösterim yöntemlerinin kullanılması gerekmektedir. Fakat bir yandan kullanım kolaylığı sağlanırken kavramsal modelin sonraki aşamalara yararlı bir girdi oluşturabilmesi için olabildiğince formal ifade edilmesi gerektiği de unutulmamalıdır. Bu kendi içinde çelişkili durumu aşmak amacıyla alan uzmanına yakın terimlerden oluşan ama sözdizimsel olarak UML tanımlarını esas alan, bazı anlambilimsel kuralların da OCL (Object Constraint Language) [7] kısıtları olarak gösterilebileceği bir görev uzayı kavramsal model oluşturma dili tanımlanmalıdır. Bu bölümde bu modelleme dilinin temel özellikleri tanıtılmıştır.

Görev uzayı kavramsal model oluşturma etkinliklerinin simülasyon geliştirme yaşam döngüsündeki yerinin [8] tanımlanması modelleme dilinin oluşturulmasında esas alınmıştır. Kavramsal model oluşturma süreci incelenerek öncelikle bakış açıları tanımlanmış, bu bakış açılarını ifade etmeye yarayacak diyagramlar ve diyagramlarda kullanılan model elemanları ile aralarındaki ilişkiler ve kurallar tanımlanmıştır.

Tanımlanan bakış açıları, diyagramlar ve model elemanları arasındaki ilişkiler **Error! Reference source not found.**'de özetlenmiştir.

**Çizelge 2.** Kavramsal Model Bakış Açıları

Bakış Açısı	Diyagramlar	Model Elemanları
Yapısal	Varlık Ontoloji	Varlık, miras, bütün-parça ilişkisi
	Komuta Hiyerarşisi	Varlık, komuta birimi, ast-üst ilişkisi
	Teşkilat Yapısı	Varlık, aktör, miras, rol
	Varlık İlişkileri	Varlık, miras, ilişki, bütün-parça ilişkisi
	Görev Uzaı	Görev, aktör, hedef, ölçüt, başarıml ölçütü, sorumludur, gerçekleştirir, genişletir, içerir, ulaşır
Davranışsal	İş Akışı	İş, akış, karar noktası, senkronizasyon çubuğu, başlama durumu, sonlanma durumu, aktör, girdi-çıkıtı, durum, girdi oluşturur ilişkisi, çıkıtı oluşturur ilişkisi
	Varlık Durumları	Durum, olay, geçiş, başlama durumu, sonlanma durumu

### 3.1 Kavramsal Model Bakış Açıları

Görev uzaı kavramsal modelleme etkinlikleri aslında iş süreçlerinin modellenmesi ve yazılım gereksinim analizi etkinliklerine oldukça benzemektedir. Bu nedenle kavramsal model bakış açılarının UML bakış açıları [3] ve ARIS bakış açıları [9] ile benzerlik göstermesi şaşırtıcı olmamalıdır.

#### 3.1.1 Yapısal Bakış Açısı

Görev uzaı kavramsal modelini oluşturan yapısal model elemanlarını ve bunlar arasındaki zaman-bağımlı olmayan ilişkileri gösteren diyagramları içerir. Yapısal bakış açısının temel yapıtaşları varlık ve görevler ve bunlar arasında tanımlı ilişkilerdir. Alan uzmanı görev uzaıdaki anlamlı bilgi kümelerini varlık olarak tanımlayabilir. Varlıklar farklı ayrıntı, gizlilik ve sadakat seviyelerinde tanımlanabilir ve tanımlanan varlıklar UML sınıfları ile gösterilir. Bir görev uzaı kavramsal modelindeki varlık sayısının çok artmasından kaynaklanabilecek olası erişim sorunlarını aşmak için OWL [10] gibi ontoloji dillerinin kullanılması ve UML'den bu dillere geçiş yöntemlerinin [11] uygulanması gerekebilir.

Varlık ontolojisinin çok büyüyeceği dikkate alınarak komuta hiyerarşisinde bulunabilen varlıklar arasındaki ilişkiler komuta hiyerarşisi diyagramlarında ayrıca ele alınmıştır. Genel varlık ontolojisine göre daha az değişken olan komuta hiyerarşisi aynı zamanda görevlere özgü rol tanımlarının yapılmasına ve teşkilat yapısının oluşturulmasına kaynak oluşturur. Komuta hiyerarşisi diyagramında genel tanımlı komuta birimleri

bulunurken (ör: tim), teşkilat yapısı diyagramlarında görevlere özel olarak kullanılan rol tanımları (ör: sızma timi) bulunur.

Geliştirilecek simülasyon sisteminin işlevselliği görevler ile tanımlanır ve görev uzayı diyagramlarıyla gösterilir. Görev uzayı diyagramlarında simülasyon hedeflerini karşılayacak görevler ve bu görevlerin gerçekleştirilmesinden sorumlu aktörler tanımlanır. Görevler arasındaki genişletme ve içerme türündeki ilişkiler de tanımlanır. Görev uzayı diyagramları ile kavramsal modelin çerçevesi netleştirilmiş ve simülasyon geliştiriciye iyi tanımlanmış bir problem tanımı bırakılmış olur.

### 3.1.2 Davranışsal Bakış Açısı

Kavramsal modelin yapısal bakış açısını oluşturan model elemanları arasındaki zaman bağımlı etkileşimler davranışsal bakış açısında ele alınır.

Görev uzayında tanımlı görevlerin işletilmesi ile ilgili ayrıntı iş akış diyagramlarında belirtilir. Görev uzayı ve iş akış diyagramları arasındaki ilişki UML'de tanımlı kullanım durumu ve etkinlik diyagramları [3] arasındaki ilişkiye benzetilebilir. İş akış diyagramında tanımlı her bir işin ayrıntısı yine bir iş akış diyagramı ile gösterilebilir. Bu özyinelemeli yaklaşım, işlerin ayrıntı seviyesine bir sınır getirmeden kavramsal modelleme olanağı tanımaktadır.

Sözdizimsel olarak görev ve iş kavramlarının aynı yapıya sahip olmaları bu özyinelemeli yaklaşımı mümkün kılmaktadır. Her bir görev iş akış diyagramlarında tanımlı işler arasındaki akışlar ile gösterilebilmektedir. Bir kavramsal modelde görev olarak tanımlanan bir kavram, diğer bir modelde iş olarak tanımlanabilir. Bu durumda görev ve iş içinde buldukları bağlama göre anlam kazanan kavramlar olarak karşımıza çıkmaktadır.

Davranışsal bakış açısının diğer bir bileşeni de varlıkların davranış modelidir. Varlık ontolojisinde tanımlı varlıkların yaşam döngüleri boyunca içinde bulunabilecekleri durumlar ve bu durumlar arasındaki geçişler varlık durumları olarak tanımlanır. Böylece henüz kavramsal analiz düzeyinde varlıkların davranışları ve geçişleri tetikleyen olaylar tanımlanmış olur. Bu tanımlamalar UML durum diyagramları [3] olarak yapılacağı için simülasyon geliştiriciye oldukça net tanımlanmış bir model olarak aktarılabilir.

### **3.2 Metamodel**

Görev uzayı kavramsal model oluşturma sürecinde Bölüm 3.1'de tanımlanan bakış açılarına kaynak oluşturan metamodel Şekil 2'de UML sınıf diyagramı olarak gösterilmiştir. Kavramsal modelde geçerli olan kuralların metamodel düzeyinde OCL ifadeleri olarak tanımlanmasıyla modelleme aşamasında yapılacak hataların en aza indirilmesi hedeflenmektedir. Bu kurallar aynı zamanda görev uzayı kavramsal modellerinin doğrulanması amacıyla da kullanılabilir.

Metamodelde tanımlı model elemanları kavramsal model oluşturma sürecinde tanımlı oldukları için alan uzmanları tarafından kolaylıkla kullanılabilir.



## 6. KAYNAKÇA

1. N. A. Karagöz, O. Demirörs, "Simülasyon Sistemleri İçin Kavramsal Model Geliştirmeye Model Tabanlı Bir Yaklaşım", USMOS05, Ankara, 2005
2. T. Clark, A. Evans, P. Sammut, J. Willans, "Applied Metamodelling A Foundation for Language Driven Development Version 0.1", <http://www.xactium.com>
3. James Rumbaugh, Ivar Jacobson, Grady Booch, "*The Unified Modeling Language Reference Manual*", Addison Wesley, 1998
4. Object Management Group, "Meta Object Facility Documentation", <http://www.omg.org/cgi-bin/doc?ptc/2004-10-15>
5. Object Management Group, "Common Warehouse Metamodel Documentation", <http://www.omg.org/technology/cwm/>
6. Object Management Group, "MDA Guide Version 1.0.1", Document Number: omg/2003-06-01, <http://www.omg.org/>
7. Object Management Group, "Unified Modeling Language: Superstructure, v2.0", <http://www.omg.org/cgi-bin/doc?formal/05-07-04>
8. N.A. Karagöz, O. Demirörs, Ç. Ündeğer, Ç. Gencel, "Simülasyon Sistemlerinde Görev Uzayı Kavramsal Modeli Geliştirme: Bir Süreç Tanımı", SAVTEK06'ya gönderildi
9. ARIS Methods v6, IDS Scheer, September 2001
10. Michael K. Smith, Chris Welty, and Deborah L. McGuinness, Editors, W3C Recommendation, "OWL Web Ontology Language Guide", <http://www.w3.org/TR/owl-guide/>.
11. D. Gasevic, D. Djuric, V. Devedzic, V. Damjanovic, "From UML to Ready to Use OWL Ontologies", Second IEEE International Conference on Intelligent Systems, Haziran 2004