ÜST MODELE DAYALI MODEL DÖNÜŞÜMLERİ

Özlem MORKAYA Tahir Emre KALAYCI

EGE ÜNİVERSİTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

İçerik

1. Giriş
2. Üst Model ve Model Dönüşümü
   - Üst Model
   - Model Dönüşümü
3. Model Dönüşüm Araçları
   - ATL
     - ATL üst model hazırlanması
     - ATL dönüşüm dili
   - GReAT
     - GME Üst Modelleme Dili
     - GME Alana Özgü Modelleme Ortamı
     - GReAT Model Dönüşümü Tanımlama Dili
     - GReAT-E dönüşüm motoru
4. ATL ve GReAT'in Karşılaştırılması
5. Sonuç

1.Giriş-1

Model Tabanlı Yazılım Geliştirme Nedir?

- Model tabanlı yazılım geliştirme bir sistemin farklı yönleri ile modellenmesine ve bir modelden diğerine otomatik çevrim için dönüşümlerin tanımlanmasına olanak sağlayan bir yazılım geliştirme yaklaşımıdır.
- Model tanımlamasında gerçekleştirim detayları en sona bırakılarak;
  - Modellerin daha taşınabilir,
  - Yeni teknolojilere daha uyumlu olması ,
  - Farklı teknolojiye sahip diğer sistemlerle daha kolay birlikte çalışması sağlanır. (Fuentes, L. ve Vallecillo.2004)

1.Giriş-2

Model Tabanlı Mimari ("Model Driven Architecture"-MDA)

- OMG tarafından modellerin yazılım geliştirme sürecinde kullanılabilmesi için önerilen ve sistemin çalışma belirtimlerini, sistemin kullandığı platformun detaylarından ayırmak amacıyla geliştirilen bir yazılım tasarım yaklaşımıdır.
- Model Tabanlı Mimari;
  - Sistemi platformdan bağımsız tanımlamak
  - Platformları tanımlamak
  - Sistem için platform seçmek ve
  - Sistem tanımlamasını belirli bir platforma dönüştürmek

için gerekli yaklaşımı sağlar ve araçların geliştirilmesi hakkında tanımlamalar içerir. (MDA Guide Version 1.0.1)

1.Giriş-3

- MDA 3 tür model ve bu modeller arasındaki dönüşümleri tanımlar.
- Hesaplama bağımsız model (CIM): Sistemi çalışma ortamı içersinde, sistemin ihtiyaçlarını bilgisayar bağımsız şekilde tanımlar.
- Platform bağımsız model (PIM): Sistemin fonksiyonlarını gerçekleştirim ayrıntılarından uzak olarak tanımlar.
- Platform bağımlı model (PSM): Sistemin fonksiyonlarını gerçekleştirmek için geliştirilen sistemin çalıştığı donanım yazılım platformuna bağlı bir modeldir.
- MDA'nın hedefi tasarlanan PIM modelden PSM modelini otomatik olarak üretmektir.
- PSM modelden yola çıkılarak kod üretimi gerçekleşebilir. (MDA Guide Version 1.0.1)

1.Giriş-4

- Bu çalışmada ATL ve GReAT model dönüşüm araçları tanıtılacaktır.
- ATL MOF/QVT tabanlı bir araçtır.
- GReAT grafiksel dönüşüme dayanan ve Modelle Entegre Hesaplama (Model Integrated Computing (MIC)) yaklaşımı çerçevesinde geliştirilmiş bir araçtır.
- MIC yaklaşımı, MDA yaklaşımı kapsamında model dönüşümü gerçekleştirir ve dönüşüm sonucu üretilen platforma bağlı modelden (PSM) bir programlama diline ait kod üretimini de gerçekleştirir. (Agrawal, G. Karsai, F. Shi)
- Literatürde MIC ve MDA yaklaşımlarını karşılaştıran pek fazla örnek yoktur.
- Bu çalışmada MIC yaklaşımının örneği olan GReAT ile MOF/QVT tabanlı MDA yaklaşımının örneği olan ATL 'in karşılaştırılması yapılmıştır.

2.Üst Model ve Model Dönüşümü-1

Üst Model Nedir?

- Bir üst model (“Meta Model”) bir modelleme dilinde geçerli olan modelleri tanımlar.
- Bir modelin söz dizimsel yapısının nasıl olacağını o modelin üst modeli belirler.
- Modelin söz dizimsel olarak doğrulanması ve modelin geçerli bir model olup olmadığını tespit etmek için üst modele ihtiyaç vardır.

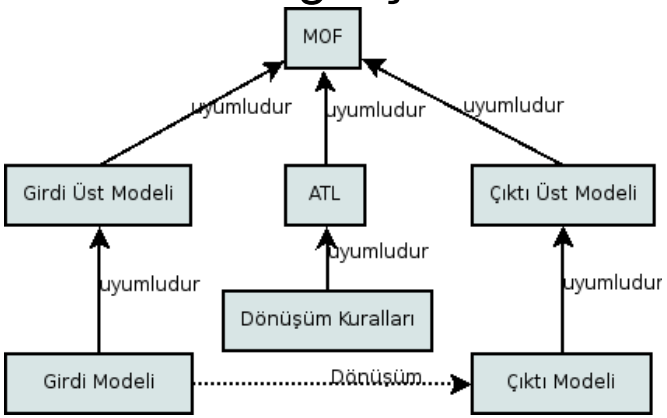
2.Üst Model ve Model Dönüşümü -2

Model Dönüşümü Nedir?

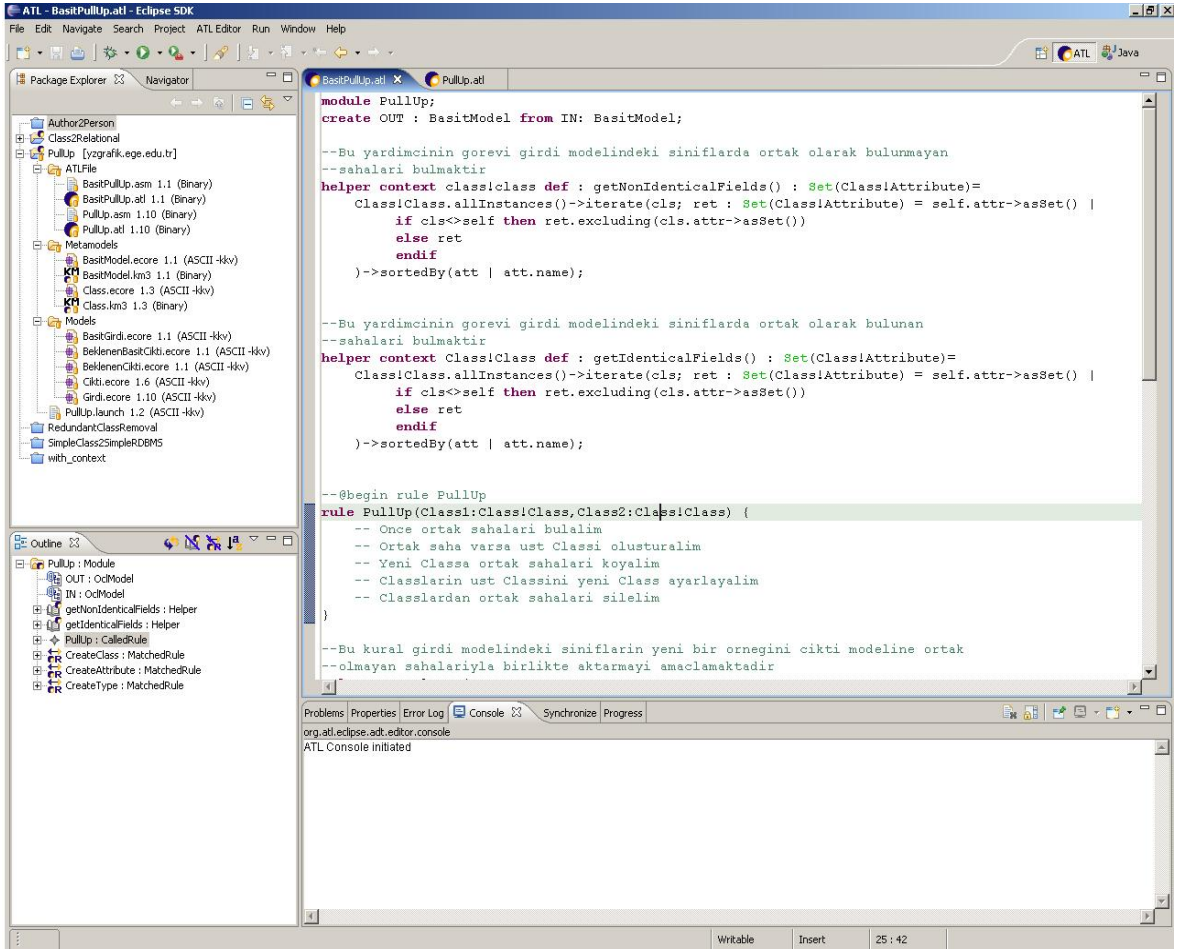
- Bir veya daha fazla girdi model üzerine bir takım dönüşüm kuralları uygulayıp sonuçta bir veya daha fazla çıktı model üretmeye *Model dönüşümü* ("Model Transformation") denir.
- Tüm girdi ve çıktı modellerin üst modeli olması gerekir.
- Girdi ve çıktı modellerin söz dizimsel olarak doğruluğu tespit etmek için üst model gereklidir.
- Sendall ve Kozaczynski model dönüşümlerini 3 türe ayırmıştır.
  - Doğrudan Model İşleme (Direct Model Manipulation):Dönüşüm genel amaçlı dillerle gerçekleştiğinden dönüşüm için gereken üst seviye soyutlamayı kısıtlar.
  - Ara Gösterim (Intermediate Representation):XML tabanlı UML modelleri arasında çevrim için geliştirilen bir standart olan XMI'dan modele, modelden XMI'a dönüşüm tekniğini kullanır.
  - Dönüşüm Dili Desteği (Transformation Language Support):Model dönüşümlerini tanımlamak için çalışma alanına ("domain") özgü bir dil kullanılmasını önerir. Dönüşüm dilleri genel olarak tanımlayıcı ("declarative"), emirsel ("imperative"), veya her ikisinin birleşimi özellikte olabilir.

3. Model Dönüşüm Araçları - ATL

- OMG tarafından yayınlanan MOF/QVT isteğine yanıt olarak Nantes üniversitesi ATLAS grubu tarafından geliştirilmiştir.
- Eclipse altında GTM için dönüşüm araçları sağlayan bir proje haline gelmiştir.
  - ATL dönüşümü için gereken kütüphaneler
  - ATL dönüşüm motoru
  - Eclipse eklentisi olan ATL geliştirme ortamından oluşmaktadır.

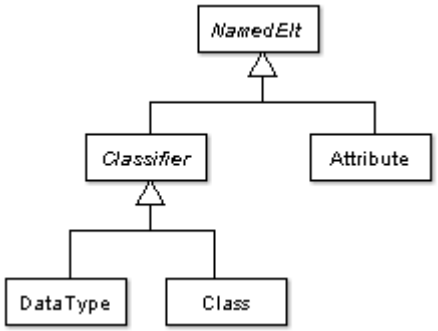


3. Model Dönüşüm Araçları - ATL



3. Model Dönüşüm Araçları – ATL

- KM3 Örneği (ATL Üst Model Hazırlanması)




```
package Class {
    abstract class NamedElt {
        attribute name : String;
    }
    abstract class Classifier extends NamedElt {
    }
    class DataType extends Classifier {
    }
    class Class extends Classifier {
        reference super[*] : Class;
        reference attr[*] ordered container : Attribute oppositeOf owner;
        attribute isAbstract : Boolean;
    }
    class Attribute extends NamedElt {
        attribute multiValued : Boolean;
        reference type : Classifier;
        reference owner : Class oppositeOf attr;
    }
}
package PrimitiveTypes {
    datatype Boolean;
    datatype Integer;
    datatype String;
}
```

3. Model Dönüşüm Araçları - ATL

- Örnek Bir Dönüşüm

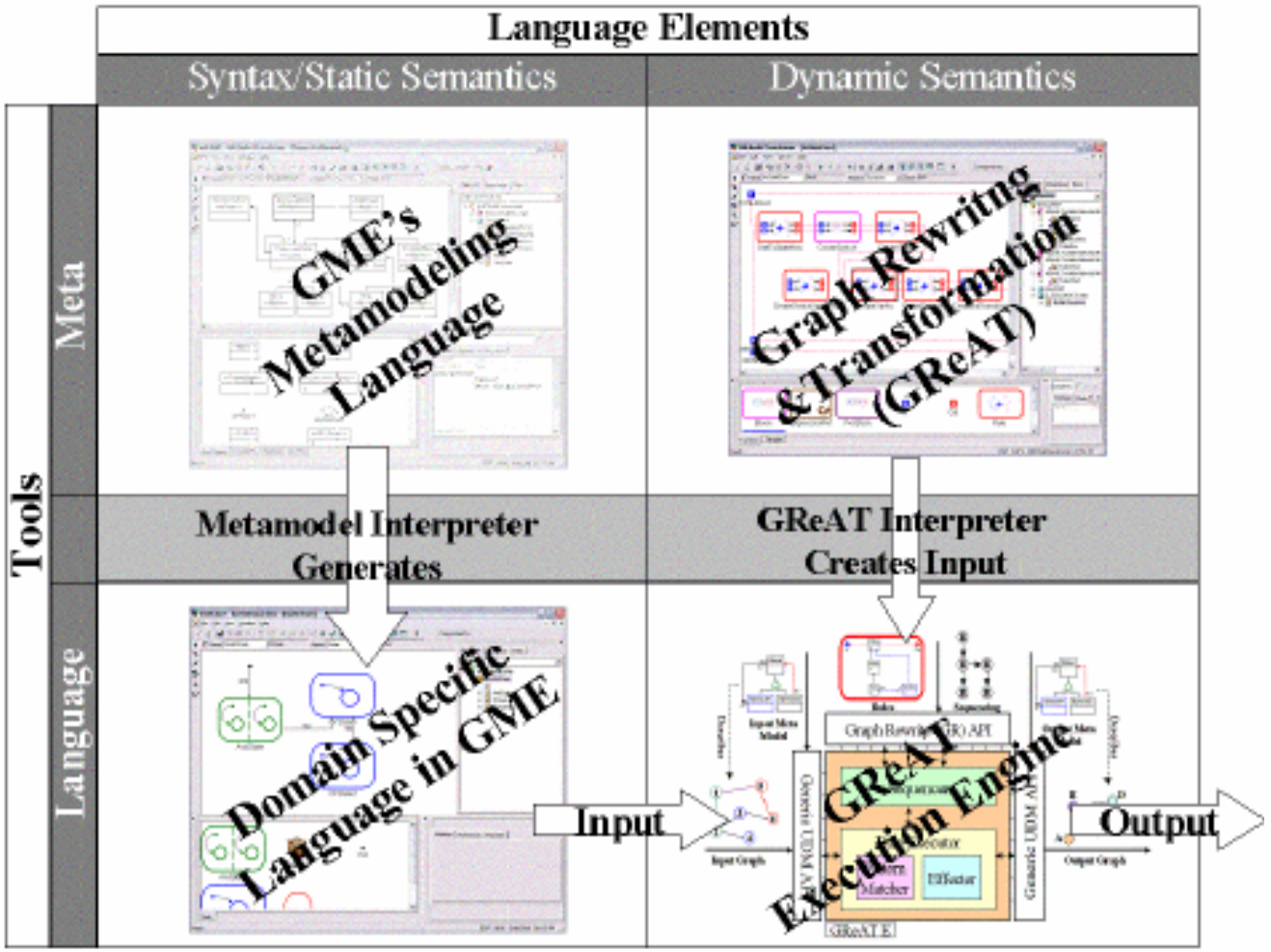
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xmi:XMI xmi:version="2.0"
        xmlns:xmi="http://www.omg.org/XMI" xmlns="Class">
  <Class name="Author" >
    <attr name="name" type="/2"/>
    <attr name="publisher" type="/2"/>
    <attr name="lastPublishDate" type="/2"/>
  </Class>
  <Class name="Reader">
    <attr name="name" type="/2"/>
    <attr name="lastBook" type="/2"/>
  </Class>
  <DataType name="String"/>
</xmi:XMI>
```

```
--Girdi modelindeki sinifin sahalarini
--cikti modeline aktaralim
rule CreateClass {
     from girdi : Class!Class
     to cikti : Class!Class (
          name <- girdi.name,
          attr <- girdi.attr,
          isAbstract <- girdi.isAbstract
     )
}

--Girdi modelindeki sahalari yeni
--modele uygun sekilde aktaralim
rule CreateAttribute {
     from girdi : Class!Attribute
     to cikti : Class!Attribute (
          name <- girdi.name,
          type <- girdi.type,
          owner <- girdi.owner,
          multiValued <- girdi.multiValued
     )
}

--Girdi modelindeki sahalarin tiplerini
--yeni modele uygun sekilde aktaralim
rule CreateType {
     from girdi : Class!DataType
     to cikti : Class!DataType (
          name <- girdi.name
     )
}
```

3.2.GReAT

- GReAT, Vanderbilt Universitesi tarafından geliştirilmiş üst modele dayalı model dönüşümü gerçekleştiren bir model dönüşüm aracıdır.
- MIC yaklaşımını destekler ve grafiksel dönüşüm yöntemi ile model dönüşümünü gerçekleştirir
- MIC yaklaşımı, MDA yaklaşımından daha eskidir.
- MDA gibi çıktı olarak platform bağımlı model üretir ve MDA'dan farklı olarak bu modelden yola çıkıp otomatik kod üretimini hedefler (Agrawal, G. Karsai, F. Shi)
- 4 bileşene sahip bir çerçeve tanımlanmıştır
  - GME üst modelleme dili (GME's metamodeling language)
  - GME modelleme ortamı (Generic Modeling Environment)
  - GReAT model dönüşümü tanımlama dili (Graph Rewriting and Transformation)
  - GReAT dönüşüm motoru (Execution Engine (GReAT-E)) (Agrawal, A., Karsai, G., and Ledeczi, A., 2003).

3.2.GReAT-2

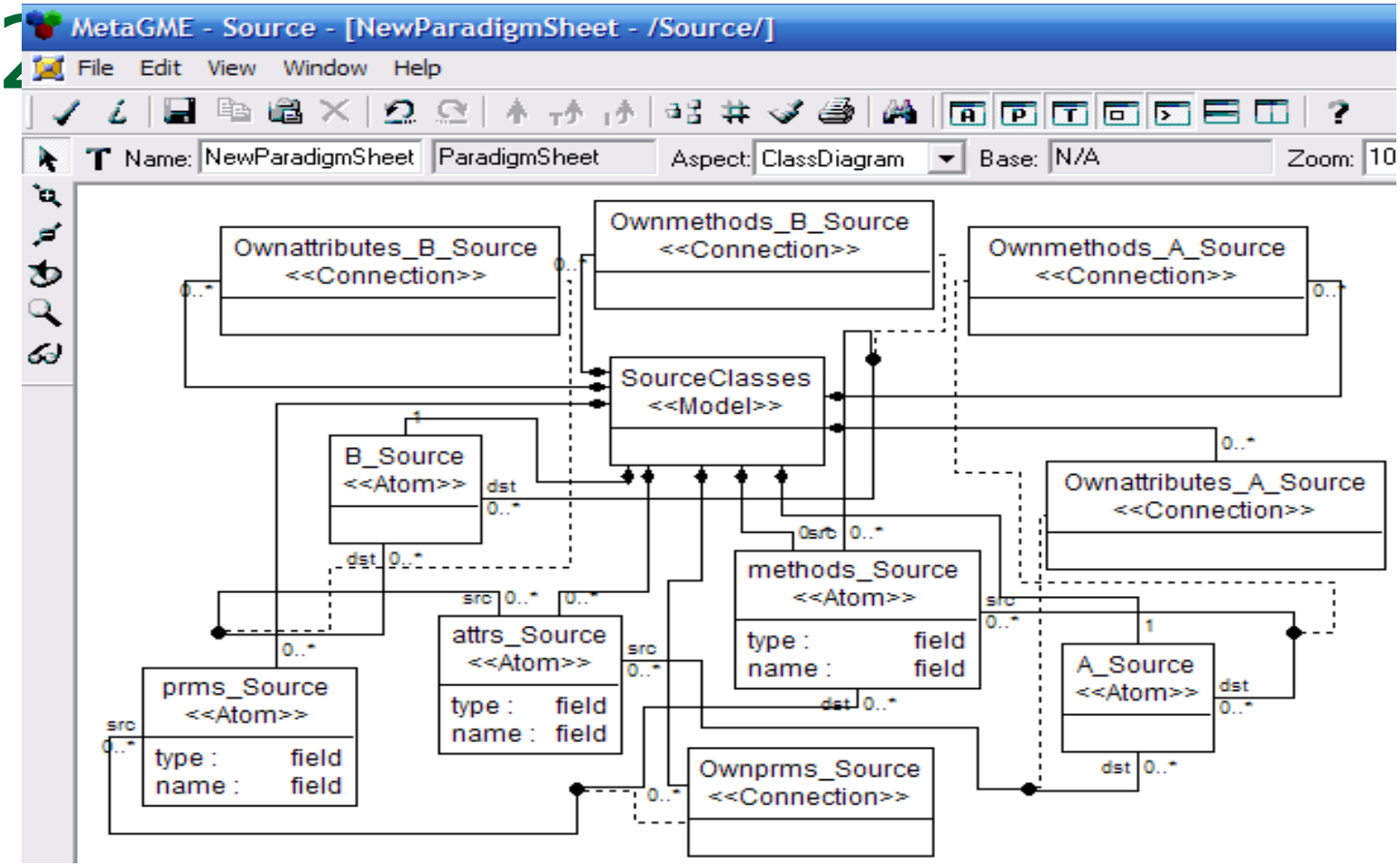


Şekil 6: Çalışma alanına özgü geliştirme çerçevesi

3.2.1GME Üst Modelleme Dili-1

- Grafiksel olarak, alana özgü üst model tanımlama olanağı sunan bileşendir.
- Üst model sisteme paradigma olarak kaydedilir ve üst modelden GME'de çalışma alanına özgü modelleme yapılabilir.
- Tanımlanan üst modeller GReAT model dönüşüm aracında kullanılır.
- Üst modellemede söz dizimsel tanımlamalar için UML sınıf diagramları kullanılır.
- Statik anlamsallık ise OCL ile tanımlanan koşullar aracılığı ile belirtilir.
- Üst modelden, model tanımlama ve model dönüşümünde üst modeli kullanmada üst modelin görsel olarak gösterimi için UML'e eklentiler yapmak gerekir. Bu eklentiler genelde önceden tanımlı nesne özellikleridir (Ledeczi, M. Maroti, A. Bakay, G. Karsai, J. Garrett, C. Thomason, G. Nordstrom, J. Sprinkle, P. Volgyesi ,2001).
- Üst modelin kapsadığı UML sınıf diagramları klasörler şeklinde gruplanır.
- Klasörlerin içinde atom (atom), model (model), görünüm (aspect), referans(references), bağlantı (connection) ve küme (set) olarak adlandırılan kısaca FCO (First class objects) denilen sınıf nesneleri bulunur.

3.2.1GME Üst Modelleme Dili-

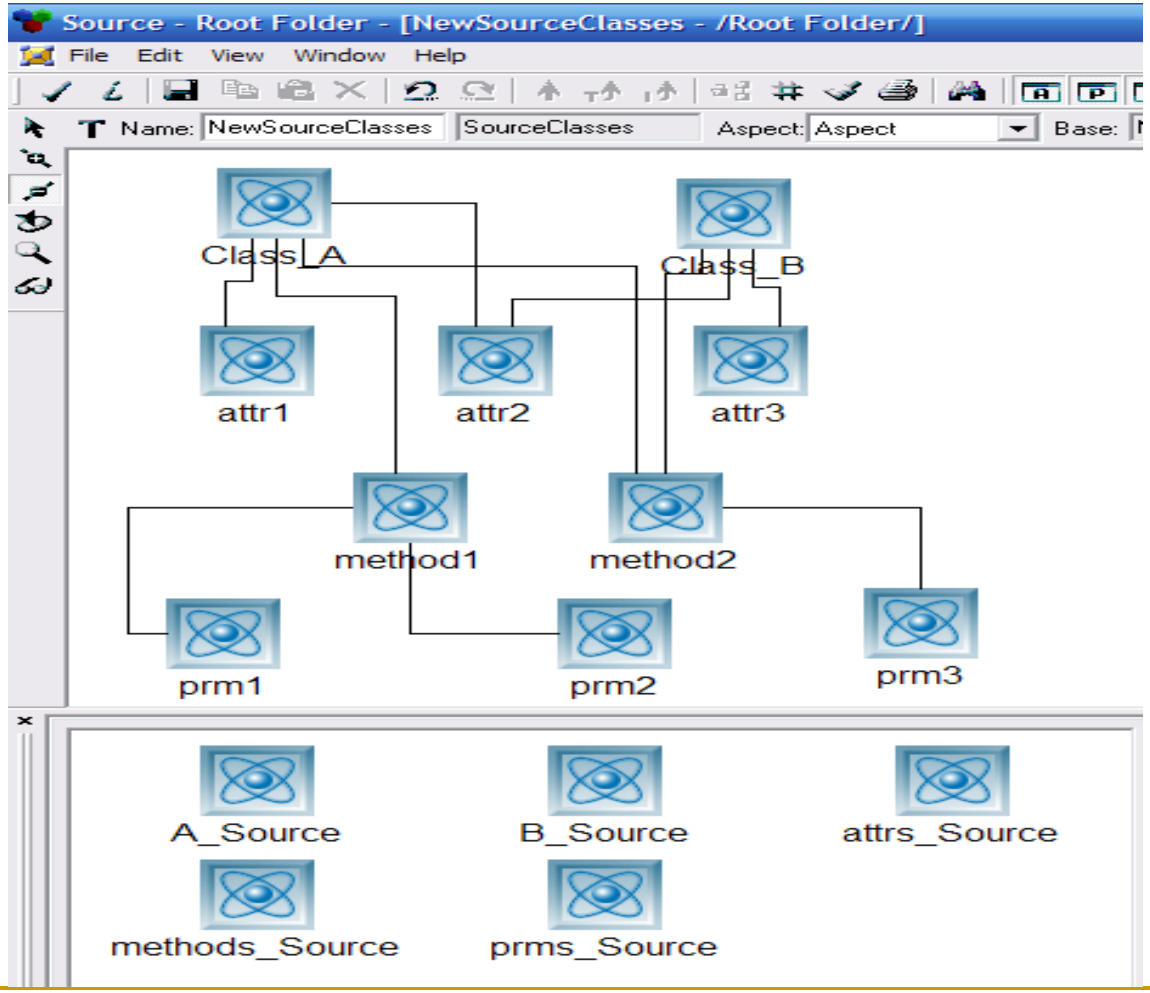


Şekil 7: Bir üst model

3.2.2 GME Alana Özgü Modelleme Ortamı-1

- Bu ortamda üst modelden model tanımlanması grafiksel olarak gerçekleştirilir.
- Üst modele ait olan atomlar grafiksel olarak modele eklenir ve bu atomlar arasında üst modelin izin verdiği şekilde bağlantılar kurulur.
- GME'de sisteme paradigma olarak kaydedilen bir üst modelden model oluşturma olanağı sayesinde üst modele uymayan model tanımlanmasının önüne geçilir.
- Modelin geçerliliğinin daha modelin oluşturulma aşamasında kontrol edilmesini sağlar.
- Tanımlanan modeller GReAT model dönüşümü aracında girdi model olarak kullanılır ve bu modeller üzerinde model dönüşümü gerçekleştirilir.

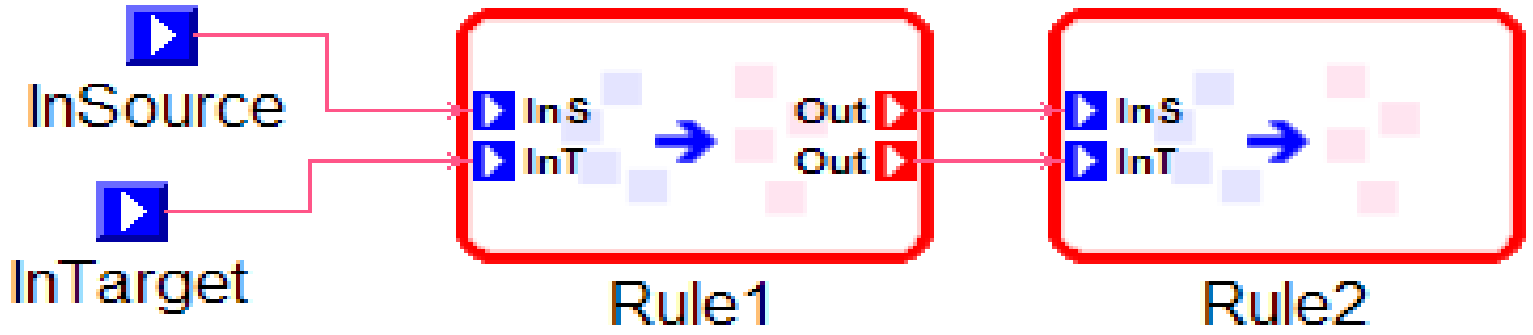
3.2.2 GME Alana Özgü Modelleme Ortamı-2



Şekil 8: Şekil 7'de sunulan üst modelden model tanımlama

3.2.3. GReAT Model Dönüşümü Tanımlama Dili

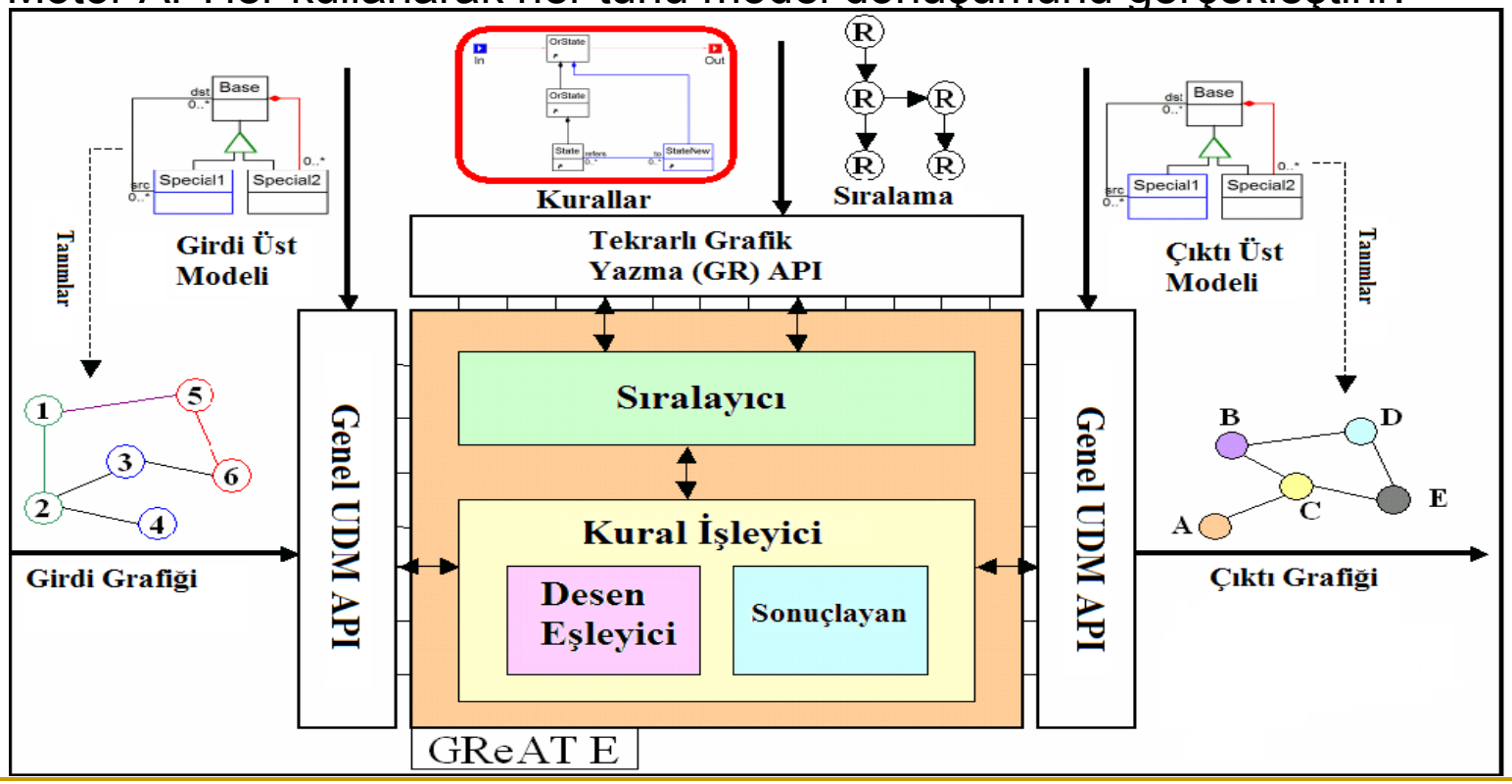
- Bu bileşen üst modele dayalı model dönüşümünü tanımlama dilidir.
- Dönüşüm dili olarak UMT (Universal Model Transformer) adlı bir dil kullanılır.
- Burada ifade (expression) tüm kural tanımlamaları için taban sınıf görevini görür.
- İfadeler özelleşerek basit (primitive) kuralları, bileşik (compound) kuralları ve testleri oluşturur.



Şekil 9: GReAT model dönüşümü

3.2.4 GReAT-E Dönüşüm Motoru-1

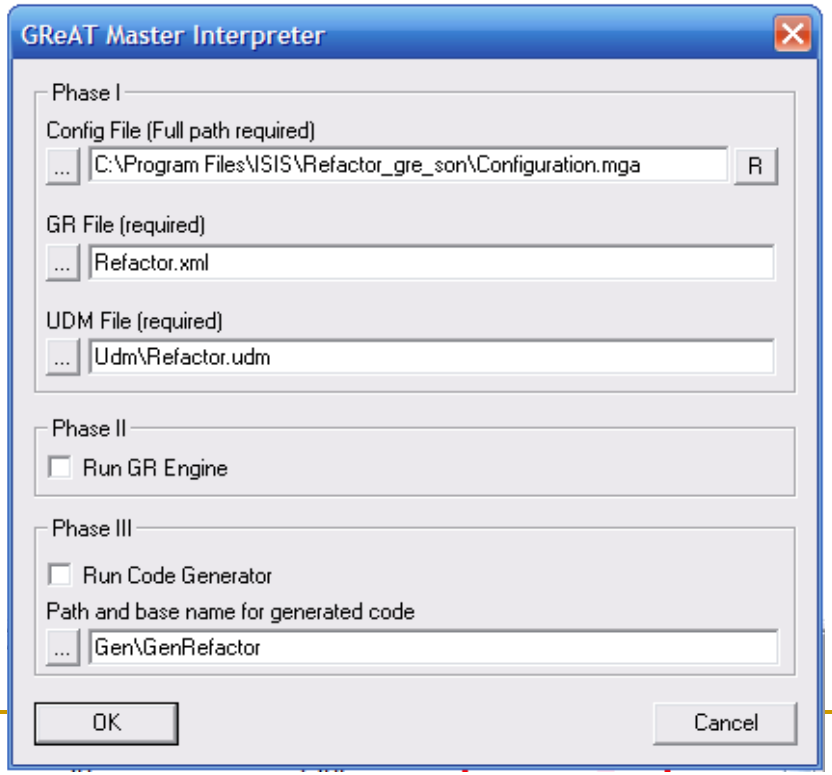
- GME dönüşüm motoru bir yorumlayıcı olarak çalışır.
- Model dönüşümünü veri yapısı şeklinde olan bir program olarak alır.
- Bu programı girdi grafik üstünde çıktı grafiği üretmek için çalıştırır.
- Motor API'ler kullanarak her türlü model dönüşümünü gerçekleştirir.



Şekil 10: GReAT-E model dönüşüm motoru (Agrawal, A., Karsai, G., and Ledeczi, A., 2003)

3.2.4 GReAT-E Dönüşüm Motoru-2

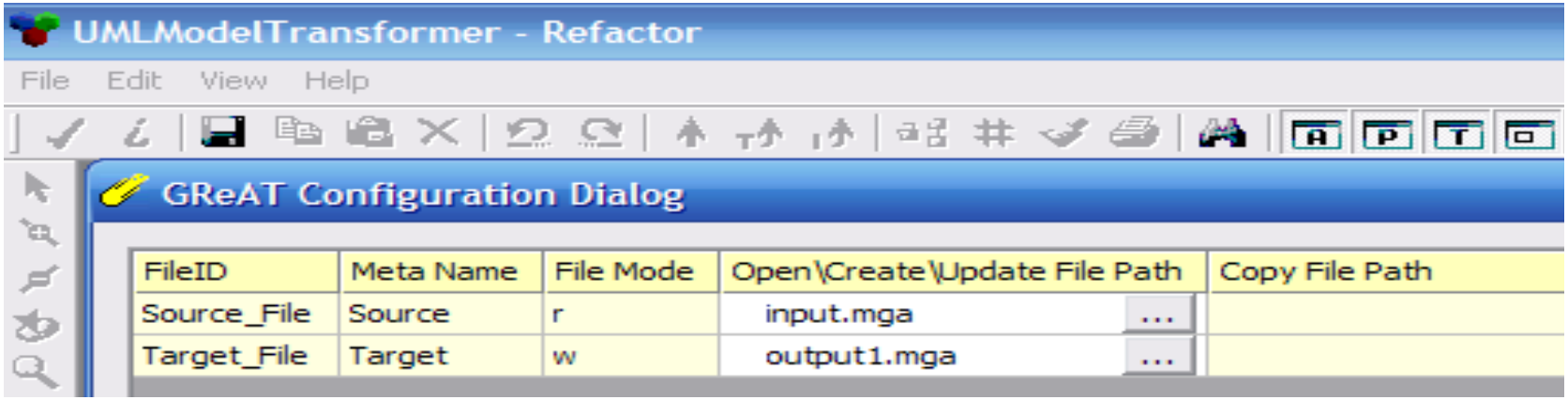
- Model dönüşümü tanımlamasının ardından GReAT-E ile dönüşüm çalıştırılabilir.
- Bunun için önce GReAT ortamında ile  gösterilen (GReAT Master Interpreter) yorumlayıcının çalıştırılması gerekir.
- Çalışma alanına özgü UDM dosyaları, konfigürasyon dosyası ve GReAT-E motorunun kullandığı tekrar yazma (rewriting) kurallarını oluşturur (GReAT Tutorial, GReAT 1.5.0 Release. 11/07/05)



Şekil 11: (GReAT Master Interpreter) Yorumlayıcı ekranı

3.2.4 GReAT-E Dönüşüm Motoru-3

- GReAT-E'nin çalıştırılması için GReAT'de (Invoke Engine) dönüşüm motorunun çalıştırılması sonucu girdi modelden çıktı model ve kod üretimi (Java, C++) gerçekleşir. (GReAT Tutorial, GReAT 1.5.0 Release, 11/07/05)



Şekil 12: GReAT-E model dönüşüm motoru ekranı

4.ATL ve GReAT'in Karşılaştırılması

- Model dönüşüm araçlarını tartışırken;
  - Araçların çalışma ortamları,
  - Model ve üst model tanımlama için sundukları olanaklar,
  - Model dönüşümü için sahip oldukları özellikler

  bu araçların kullanım kolaylığını ve verimli kullanılmasını belirleyen önemli karşılaştırma kriterleridir.
- ATL ve GReAT 4 başlık altında karşılaştırılacaktır.
  - Genel Özellikler
  - Modelleme Ortamı
  - Üst Model Hazırlanması
  - Dönüşümlerin Tanımlanması

4.1 Genel Özellikler

- ATL, GReAT'e göre daha yeni bir araçtır.
- GReAT'e ait döküman sayısı daha fazladır.
- GME MS/COM (Component Object Model) teknolojisine dayanan bir mimariye sahiptir.
- ATL dili ise EMF modelleme anaçatısının üzerinde çalışmaktadır (Bézivin, J., Brunette, C., Chevrel, R., Jouault, F. ve Kurtev I, 2005).
- ATL Eclipse Java ortamına entegre edilerek çalışır.
- GReAT'in ise çalışması için Microsoft VS.NET yazılımına gerek vardır.
- Her iki araç da ücretsiz olarak kullanılan model dönüşüm araçları olmasına karşın GReAT'in Microsoft VS.NET'e gereksinimi bu noktada ATL'i daha öne çıkarmaktadır.

4.2 Modelleme Ortamı

- GReAT'in model tanımlama ortamı grafiksel bir tasarım ortamıdır.
- Kullanıcıya kolay ve zevkli bir çalışma ortamı sağlar.
- Model tanımlanırken sadece üst modelin izin verdiği bağlantılar model elemanları arasında kurulabilir.
- Bu daha model tanımlama aşamasında modelin geçerliliğinin kontrol edilmesini sağlar.
- ATL'de ise modeller XML döküman dosyalarında tanımlanırken üst model ise bu XML dökümanının ad uzayı olarak belirtilir.
- Üst modelin ad uzayı olarak belirtilmesi de modelin üst modele uygunluğunun kontrol edilmesini sağlar.

4.3 Üst Model Hazırlanması

- GReAT'de UML diagramları benzeri diagramlarla üst model tanımlanır.
- UML'in kısıtları tanımlamada yeterli gelmediği durumlarda ise OCL ile kısıt tanımlaması gerçekleştirilir.
- ATL ise üst model tanımlamada grafiksel bir ortam yerine metinsel bir ortam sunar.
- ATL'de üst model tanımlamak için KM3 dosyaları kullanılır.
- KM3 olarak tanımlanan üst modeller Ecore veya MOF üst modellerine çevrilebilir.
- GReAT'de ise tanımlanan üst modeller UML üst modeline çevrilir.

4.4 Dönüşümlerin Tanımlanması

- Model dönüşümünde ATL genellikle tanımlayıcı (“declarative”) ve biraz da emirsel (“imperative”) bir yaklaşım izler.
- ATL model dönüşümünü ifade etmek için OCL ifadelerini kullanır.
- ATL, MOF tabanlı model dönüşümü gerçekleştirir.
- GReAT ise grafik dönüşümüne odaklanmıştır ve grafik grammeri ve dönüşümü GGT (Graph grammars and graph transformations) algoritmalarını model dönüşümünde kullanır.
- GReAT’de model dönüşümünde grafiksel olarak ifade edemediği kısıtları tanımlamak için OCL ‘i kullanır.
- ATL’de ise kurallar “atl” uzantılı bir dosyada metinsel olarak tanımlanır.
- Aynı ATL dosyasında çok sayıda kural tanımı bulunabilir.

5.Sonuç-1

- Model tabanlı yazılım geliştirme, yazılım teknolojilerinin hızla ilerlemesi sonucu mevcut yazılımların yeni teknolojilere uyumu, platform bağımlılığı gibi sorunlara çözüm olarak önerilmiştir.
- Model tabanlı yazılım geliştirme uygulama için kod yazımından ziyade uygulamanın çalışma alanının modellenmesine yoğunlaşır.
- ATL ve GReAT model dönüşüm araçlarının kullanılması ile model, üst model ve model dönüşümü aşamaları adım adım anlatılmıştır.
- Kullanıcıların model tabanlı yazılım geliştirmeyi ve onun aşamalarını kavraması hedeflenmiştir.

5.Sonuç-2

- Literatürde MDA konusunda yapılan çalışmalar ya ayrıntılı olarak tek bir araç üzerinde yoğunlaşır ya da genel özellikleri ile model dönüşüm araçlarından bahsederek onları sınıflandırır.
- Bu çalışmada ise literatürde yapılandan farklı olarak iki ayrı model dönüşüm aracı ayrıntılı biçimde ele alınıp karşılaştırılmıştır.
- Kullanıcılar iki farklı model dönüşüm aracını inceleme olanağı bulmuş ve karşılaştırmalı olarak model dönüşümü aşamalarını görme fırsatı yakalamıştır.
- Model tabanlı yazılım geliştirmenin, hem sağladığı soyutlama artışı hem de üretkenlik artışı ile yakın gelecekte yazılım mühendisliğinde önemli bir araştırma alanı olacağı ve yazılım geliştirmede daha yoğun olarak yer alacağı düşünülmektedir.
- Bu çalışma ile bu alandaki mevcut literatürün ve var olan geliştirme araçlarının kısa bir tanıtımı amaçlanmıştır.

TEŞEKKÜR

- Model tabanlı yazılım geliştirme konusunda ders vererek bizlere bu konuyu tanıttığı ve bu çalışmamıza katkıda bulunduğu için değerli hocamız ***Prof.Dr. Yasemin TOPALOĞLU***'na teşekkürü bir borç biliriz.

6.Kaynakça-1

- Schmidt, D. C., ``Model-Driven Engineering", *IEEE Computer*, 39(2):25-31, 2006.
- Fuentes, L. ve Vallecillo. A., “An Introduction to UML Profiles”, *UPGRADE, The European Journal for the Informatics Professional*, 5 (2): 5-13, 2004.
- Seidewitz, E., "What Models Mean", *IEEE Software* , 20(5):26-32, 2003.
- Atkinson, C. ve Kühne, T., “Model-Driven Development: A Metamodeling Foundation”, *IEEE Software,* 20(5):36-41, 2003.
- Sendall, S. ve Kozaczynski, W., “Model Transformation – the Heart and Soul of Model-Driven Software Development”, *IEEE Software*, 20(5):42-45, 2003.
- Wagner; A., “A Pragmatic Approach to Rule-Based Transformations within UML using XMI.difference”, *WITUML: Workshop on Integration and Transformation of UML models (held at ETAPS 2001)*, 2001.

6.Kaynakça-2

- Agrawal, A., Karsai, G., and Ledeczi, A., “An End-to-End Domain-Driven Software Development Framework”, *Conference on Object Oriented Programming Systems Languages and Applications*, 2003.
- Ledeczi, M. Maroti, A. Bakay, G. Karsai, J. Garrett, C. Thomason, G. Nordstrom, J. Sprinkle and P. Volgyesi, The Generic Modeling Environment, Vanderbilt University, ISIS (WISP'2001, May, 2001 in Budapest, Hungary), IEEE 2001
- Agrawal, Z. Kalmar,G. Karsai, F. Shi, A. Vizhanyo, “GReAT User Manual”, ISIS, Vanderbilt University, 2003.
- Agrawal, G. Karsai, F. Shi, “Graph Transformations on Domain-Specific Models”, GReAT Technical Report, ISIS, Vanderbilt University
- GReAT Tutorial, GReAT 1.5.0 Release, 11/07/05, Available at : http://repo.isis.vanderbilt.edu/downloads/,

6.Kaynakça-3

- Object Management Group: OMG/RFP/QVT MOF 2.0 Query/Views/Transformations RFP. October 2002, Available at : http://www.omg.org/docs/ad/02-04-10.pdf
- Bezivin, J., Dupé, G., Jouault, F., Pitette, G. ve Rougui, J.E., "First experiments with the ATL modeltransformation language: Transforming XSLT into Xquery", *OOPSLA Workshop on Generative Techniques in the context of MDA,* .2003.
- Gerber, A., Lawley, M., Raymond, K., Steel, J. ve Wood, A., "Transformation: The Missing Link of MDA", *ICGT*, 2002
- Bezivin, J., Jouault, F., Rosenthal, P., Valduriez, P., "The AMMA platform support for modeling in the large and modeling in the small", *Lina Research Report No:04.09*. 2005
- Bezivin J., Jouault F., Touzet D. , "An introduction to the ATLAS Model Management Architecture". *Lina Research Report No:05.01*. 2005

6.Kaynakça-4

- MDA Guide Version 1.0.1, Erişim: http://www.omg.org/cgi-bin/doc?omg/03-06-01
- Bézivin, J., Brunette, C., Chevrel, R., Jouault, F. ve Kurtev I., "Bridging the Generic Modeling Environment (GME) and the Eclipse Modeling Framework (EMF)", *OOPSLA Workshop on Best Practices for Model Driven Software Development*, 2005.
- ATLAS Group: ATL User Manual, ATL User Manual, version 0.7. 2006. Erişim: http://www.eclipse.org/m2m/atl/doc/ATL_User_Manual%5Bv0.7
- Mens, T. ve Van Gorp, P., "A Taxonomy of Model Transformation", *Proceedings of the International Workshop on Graph and Model Transformation*, 2006.
- Czarnecki, K. ve Helsen, S., "Classification of Model Transformation Approaches", *OOPSLA Workshop on Generative Techniques in the Context of MDA*, 2003.

KATILIMINIZ İÇİN

TEŞEKKÜRLER...