Model Transformations for Hypertext modeling on Web Information Systems

Paloma Cáceres, Valeria de Castro, Juan M. Vara, Esperanza Marcos
Rey Juan Carlos University

{paloma.caceres, valeria.decastro, juanmanuel.vara, esperanza.marcos}@urjc.es

ABSTRACT
In response to the lack of suitable methods to build the navigation model of Web Information Systems (WIS), we presented in past works HM³, the Hypertext Modeling Method of MIDAS, a methodological framework for the agile development of WIS. We proposed a method to obtain the navigation model of a WIS, starting from the conceptual data model and the user requirements, collected in the use case model. In this work, continuing with the MDA approach for WIS development that we are following with HM³, we define the model to model transformations needed at the PIM level to get the hypertext model from the behavioral and content models. Firstly we specify the transformation rules with natural language to later map them to graph transformation rules.

Keywords
Model Transformation, Hypertext/Hypermedia, User Issues, Modeling Methodologies.

1. INTRODUCTION
Despite the differences found between the traditional methodologies for Web Information Systems development, all they follow a similar approach to obtain the navigational model: they start from the conceptual model [8] [9]. In previous works and in response to certain lacks suffered by this structural approach, we proposed to address the problem of the systematic construction of the navigation model from a user services oriented perspective [4] [7]. That is, we mainly take into account the services required by the user, what we call conceptual user services. This way, we proposed the Hypertext Modeling Method of MIDAS (HM³): a method to obtain the navigation model starting from the user services model, a use case model in which we identify conceptual user services as stereotyped use cases. In HM³ we also take into account the conceptual data model, but just in a second step, to complete the information collected in the output model.

The proposed method includes the process, the models and the mappings between them. As mentioned previous works show, the mappings between the models play a very important role, given that the method is based on a continuous development process in which, according to the Model Driven Architecture (MDA) principles [10], the models act as the prime actors. Each step of this process consists basically in the generation of an output model starting from one or more input models on which the mapping rules are applied. In the remaining steps of the process, this output model acts as one of the input models.

In this work, we complete the definition of HM³ by specifying a graph transformation approach to effectively realize the mappings defined in HM³. The term “Graph Transformation” is used to refer to a special kind of rule-based transformations that are typically represented diagrammatically [20]. So, given that we define the HM³ mapping rules in a rule-based manner, it seems properly to use a graph transformation approach to later formalize them.

HM³ is part of MIDAS, a model driven methodology for agile development of WIS. MIDAS proposes to model the WIS according to three basic aspects: hypertext, content and behavior. This work focuses on the MIDAS method to model the hypertext aspect whose final output is the navigation model, obtained by means of the model to model transformations at PIM level presented in this work.

2. THE FRAMEWORK AND THE PROCESS
In this section we summarize the main characteristics of the MDA framework at which this work is located and the modeling process that is completed with the model to model transformations specified in this work.

2.1 MIDAS Framework
MIDAS is a methodological framework for agile development of WIS, which proposes a Model Driven Architecture [5] based on the MDA proposed by the Object Management Group (OMG) [10]. MIDAS proposes to model the WIS according to two orthogonal dimensions (see Figure 1).

Figure 1. Model Driven Architecture of MIDAS.
On the one hand, MIDAS proposes to model the WIS taking into account the platform dependence degree (based on the MDA approach): firstly, specifying the whole system by Computation Independent Models (CIMs), Platform Independent Models (PIMs) and Platform Specific Models (PSMs); and secondly, specifying the mapping rules between these models. On the other hand, MIDAS proposes to model the WIS according to three basic aspects [11]: hypertext, content and behavior. Besides, MIDAS suggests using the Unified Model Language (UML) as unique notation to model the whole system, as well as the use of UML profiles (or the definition of new UML profiles) to model aspects not covered in the UML standard [12][13][23].

As mentioned previously, in this work we focus on the transformations between the PIM models that were introduced in [4], involved in the modeling of the hypertext aspect of a WIS.

2.2 The Hypertext Modeling Method of MIDAS

The Hypertext Modeling Method of MIDAS (HM³) [4], [7] is defined from a user services oriented perspective, that is, taking into account the services required by the users of the WIS. The inputs of the method in the HM³ process are the user requirements and the conceptual data model; the output is the extended navigation model.

As can be shown in Figure 2(a), the process includes three new intermediate models to finally obtain the extended navigation model: the user services model, the extended user services model and the extended slices model. Besides, an Activity Diagram is used to model the use services composition and to obtain the route that will guide the user on his navigation through the WIS.

The HM³ process is an iterative and incremental process that sometimes requires the decision making by the system designer. To clarify this decision making, we have captured the whole process in the activity diagram showed in Figure 2(b). In this figure, we have identified the different tasks associated with each one of the steps, that is to say, with the generation of the different models. As showed in Figure 2(a) and Figure 2(b), the output of each one of the steps composing the process is a new model: user services model, extended use case model, extended slice model and extended navigation model.

This way, we have defined the process as the set of activities needed to generate each one of these models. By means of simple tasks to do over the input models, the respective output model is generated for each step. It becomes obvious that these tasks are no more than the set of mappings between models that must be carried out for each step of the process.

Figure 2. a) HM³ process summary and b) HM³ process detailed

2.3 UML Profile for Hypertext Modeling

As well as HM³ needs to represent UML elements with new semantics associated, we have extended the UML metamodel by defining a new profile (a package containing model elements that have been customized for a specific domain or purpose using extension mechanisms such as stereotypes, tagged definitions and constraints). The set of stereotypes used to define the profile, and the relative UML metaclass extended by each one can be found on Table 1, while the graphical presentation of these stereotypes is showed in Figure 3. This new UML extension is collected in the definition of four metamodels, one corresponding to each one of the new models introduced in HM³: User Services metamodel, Extended User Services metamodel, Extended Slices metamodel and Extended Navigational metamodel, and the restrictions over these metamodels defined with the OCL standard.

These metamodels are presented in Figure 4 with dashed lines pointing out the new elements added in each one. In next subsections we will give just a brief overview of each one of these metamodels. For an in-depth explanation on the semantics associated of anyone of the new elements included in the different metamodels, the reader is referred to [4] and [7].
2.3.1 The User Service Metamodel

In this metamodel, two stereotypes are considered: the Conceptual User Service and the User Service Actor stereotypes. The application of these stereotypes over the respective UML metamodel elements specified on Table 1 (Use Case and Actor), results in two new classifiers: the Conceptual User Service, a specific kind of Use Case and the User Service Actor, a specific kind of Actor. This way, one Actor in the User Services Model (that is one User Service Actor) can only be related to one or more Conceptual User Services and vice versa.

Table 1. HM³ profile Stereotypes

<table>
<thead>
<tr>
<th>UML meta-class</th>
<th>Stereotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor</td>
<td>User Service Actor</td>
</tr>
<tr>
<td>Use Case</td>
<td>Conceptual User Service, Composite Use Service,</td>
</tr>
<tr>
<td></td>
<td>Structural Basic Service, Functional Basic Service</td>
</tr>
<tr>
<td>Class</td>
<td>Structural Slice, Functional Slice, Index, Menu,</td>
</tr>
<tr>
<td></td>
<td>Query</td>
</tr>
<tr>
<td>Association</td>
<td>Route</td>
</tr>
<tr>
<td>Model</td>
<td>User Service Model, Extended User Service Model,</td>
</tr>
<tr>
<td></td>
<td>Extended Slice Model, Extended Navigational</td>
</tr>
<tr>
<td></td>
<td>Model, Conceptual Data Model</td>
</tr>
</tbody>
</table>

According to UML, these restrictions must be expressed in OCL:

```
context User Service Model inv Contents:
  self.contents->forAll(c |
    c.oclIsKindOf(Actor) And
    c.stereotype.name -> includes("User Service Actor")
  ) Or
  (c.oclIsKindOf(UseCase) and c.stereotype.name -> includes("Conceptual User Service"))
```

2.3.2 The Extended User Services Metamodel

The stereotypes incorporated in the Extended User Services metamodel are the Composite Service, the Functional Basic Service and the Structural Basic Service stereotypes. As well as with the User Service metamodel, certain restrictions must be
defined to ensure the consistency of any instance of this metamodel.

```java
context Extended User Services Model inv Contents:
self.contents->forAll(c |
(c.oclIsKindOf(Use Case) and
(c.stereotype.name -> includes("Composite Service")) or
(c.oclIsKindOf(Class) and
(c.stereotype.name -> includes("Index"))) or
(c.oclIsKindOf(Dependency) and
(c.stereotype.name -> includes("User Service Actor")) or
(c.oclIsKindOf(Actor) and
(c.stereotype.name -> includes("include")))

This way, the OCL restriction ensures that only Composite Services, Basic Services and User Services Actors can be included in a Extended User Service Model. At the same time, the OCL restriction defines the ways in which these elements can be related in the model.

2.3.3 The Extended Slices Metamodel

Starting from the Slice metamodel of the RMM methodology [9], the Extended Slices metamodel includes two new classifiers, the Structural Slice and the Functional Slice and a new kind of association, the Route, resulted from the application of the respective stereotypes on the UML metamodel elements. Once again, an OCL restriction must be defined to ensure the consistency of the instances of this metamodel:

```java
context Extended Slices Model inv Contents:
self.contents->forAll(c |
(c.oclIsKindOf(Class) and
(c.stereotype.name -> includes("Structural Slice")) or
(c.oclIsKindOf(Class) and
(c.stereotype.name -> includes("Functional Slice")) or
(c.oclIsKindOf(Dependency) and
(c.stereotype.name -> includes("Route")))

This restriction ensures that any Extended Slices Model will only contain instances of the elements included in the respective metamodel.

2.3.4 The Extended Navigation Metamodel

Finally, the Extended Navigational metamodel incorporates three new specializations of the meta-element class: Query, Menu and Index. All of them identify navigational structures to be included on the WIS. In this case, the corresponding OCL restriction must be added to the one defined for the Extended Slices metamodel to ensure the consistency of the generated models:

```java
context Extended Navigation Model inv Contents:
self.contents->forAll(c |
(c.oclIsKindOf(Class) and
(c.stereotype.name -> includes("Query"))) or
(c.oclIsKindOf(Class) and
(c.stereotype.name -> includes("Menu")))

3. MAPPINGS IN HM³

Since models are the conducting thread in the HM³ process in which each step involves the generation of one model from one or more input models, as well as a lower abstraction level, it comes clear that the HM³ process is a Model Driven process [2] [17]. Once we have defined the metamodels (consequently the models) that must be considered, to complete the definition of HM³ according to MDA principles, it rests to define the mappings between the different models that must be generated along the process [14]. According to [18], we can refer to this process as model transformation.

3.1 HM³ Model Transformation Rules

According to [14] “the mapping description may be in natural language, an algorithm in an action language, or a model in a mapping language”. In this CASE, and as a first approach to model transformation on HM³, we have decided to describe the transformation rules in natural language for later express them as graph transformation rules.

In the transformation process of HM³, the initial Conceptual User Services identified in the User Services Model are broken up into Composite and Basic Services. At the same time these Composite Services are broken up into Basic Services. The resulting Basic Services are identified as Functional or Structural Services to conform the Extended User Services Model. Besides, for each Conceptual User Service, an activity diagram is generated to model the service composition and to identify the route that will guide the user in its navigation through the WIS. In the next transformation level, the Extended Slices Model is generated by adding one slice for each Basic Service in the Extended User Services Model. The attributes for each slice are taken from the Conceptual Data Model and the routes are identified from the Activity Diagram associated to each Conceptual User Service. We have collected these mappings in a set of rules defined in natural language. As gathered in [14], since some of the mapping rules of the transformation process in HM³ require design decisions, it is not possible to automate them completely. So, we have made the distinction between the mapping rules that can be automated and those that can not. Table 2 shows all the mapping rules of HM³ described in natural language and its grade of automation: Completely (c) or Partial (P). Once again, for an in-depth presentation of the concepts included on the mapping rules, we suggest the reader to refer to [4] and [7].

<table>
<thead>
<tr>
<th>Table 2. HM³ Transformation Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User Services Model (USModel) to Extended User Services Model (XUSModel)</strong></td>
</tr>
<tr>
<td>1. For every User Service Actor identified in the USModel (source model), there will be a User Service Actor in the XUSModel preserving the same name.</td>
</tr>
<tr>
<td>2. Every Conceptual User Service (CUS) found in the USModel will be split on one or more Composite and/or Basic Services (CS and BS). The new CUSs and BSs will be identified with the name of the respective CUS, followed by the name chosen for the new Services (that is CUSName.BSName).</td>
</tr>
<tr>
<td><strong>2.1 Every Composite Service generated will be split into one or more Basic Services.</strong></td>
</tr>
<tr>
<td><strong>2.2 Every Basic Service will be identified as Structural or Functional service.</strong></td>
</tr>
</tbody>
</table>
User Service Model (USmodel) and Extended User Services Model (XUSmodel) to Activity Diagram (AD)

3. For every Conceptual User Service (CUS) identified at the USmodel, there will be a new Activity Diagram.C

4. For every Basic Service (BS) corresponding to a same Conceptual User Service (CUS) (as stated in rule 2, if BS’s name is CU$\text{Name}$ BS$\text{Name}$, we refer to those BSs who shares the same CU$\text{Name}$), there will be an activity in the Activity Diagram generated for that CUS.C

5. Every extend association identified in the XUSmodel will be represented in the Activity Diagram with a fork. The activity corresponding to the source Basic Service of the extend association (SAct) must be previous to the activity corresponding to the target Basic Service of the extend association (TAct).C

5.1 If the extend association has only one source, the fork will present the SAct as an alternative to another path with no activities. Later, both paths will meet.C

5.2 If the extend association has several sources, the fork will present the different SActs as mutual alternatives to another path with no activities. Later, all these paths will meet.C

6. Whenever a include association is found in the XUSmodel, the Activity corresponding to the source Basic Service of the include association (SAct) must be subsequent to the activity corresponding to the target Basic Service of the include association (TAct).C

6.1 If the include association has several targets, the designer must decide the appropriate sequence for the different TActs (that will be obviously previous to the SAct).P

Activity Diagrams (AD), Conceptual Data Model (CDmodel) and Extended User Service Model (XUSmodel) to Extended Slices Model (XSmodel)

7. For each Structural Basic Service in the XUSmodel, there will be a Structural Slice in the XSmodel.C

8. For each Functional Basic Service in the XUSmodel, there will be a Functional Slice in the XSmodel.C

9. The attributes for each Slice in the XSmodel will be obtained directly from the respective classes in the CDmodel.P

10. Every AD will be represented in the XSmodel as a service route named by the Conceptual User Service corresponding to the AD. That is CUS$\text{Name}$.P

11. Each fork in any AD will be represented in the XSmodel by a subroute. The name for the subroute will be composed by the name of the respective Conceptual User Service plus a new name to univocally identify the subroute. That is CUS$\text{Name}.IDSubroute$.P

Extended Slices Model (XSmodel) to Extended Navigation Model (XNmodel)

12. Each fork in the XSmodel will be mapped to an index in the XNmodel.C

13. Each association whose target element is a Structural Slice in the XSmodel will be mapped to an index or a query in the XNmodel. The designer will take the decision for each case.P

14. The XNmodel will include an initial menu with one item for each route included in the XSmodel.C

To clarify the description of the rules referring to the extends and include associations found in the Extended User Services Model, we have specified which elements in the source and target models are affected by each transformation, as well as some partial diagrams that helps on the understanding of these rules. These considerations can be found on Table 3.

Table 3. HM³ Transformation Rules to generate the Activity Diagrams

<table>
<thead>
<tr>
<th>Source Model</th>
<th>Source Elements</th>
<th>Target Elements</th>
<th>Target Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extended User Services</td>
<td><code>&lt;include&gt;</code></td>
<td>B</td>
<td>Activity Diagram</td>
</tr>
<tr>
<td></td>
<td><code>&lt;extend&gt;</code></td>
<td>C</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;include&gt;</code></td>
<td>B</td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>&lt;include&gt;</code></td>
<td>B'</td>
<td></td>
</tr>
</tbody>
</table>

1236
3.2 Graph Transformations in HM³

To observe the MDA principles, the model to model transformation of HM³ must be automated, at least in some extent. To achieve this objective we have decided to use a graph transformation approach ([11] [6] [20]), since projects like the Attributed Graph Grammar System (AGG) [3] [19] will provide us with the facilities to automate model to model transformations defined as graph transformations. Moreover, as previously mentioned the term Graph Transformation is used to refer to a particular category of rule-based transformations that are typically represented diagrammatically [20]. So, given that we have already formally defined the HM³ mapping in a set of rules, it seems appropriate to translate these rules to graph transformations. In this way, we can think of UML-like models as graphs. A graph must be defined. These rules follow the structure LHS:= RHS (Left Hand Side:= Right Hand Side). Both, the LHS and the RHS are typically represented diagrammatically [20]. So, given that we have already formally defined the HM³ mapping in a set of rules, it seems appropriate to translate these rules to graph transformations. Finally, from a pure mathematical point of view, we can think on UML-like models as graphs. A graph has nodes and arcs while a UML model have classes and associations between those classes.

To express model transformations by graph grammars, a set of graph rules must be defined. These rules follow the structure LHS:= RHS (Left Hand Side:= Right Hand Side). Both, the LHS and the RHS are graphs: the LHS is the graph to match while the RHS is the replacement graph. If a match is found on the source model, then it is replaced by the RHS in the target model [18].

In this work we have used the approach proposed in [15] to define the graph rules that collects the transformation rules proposed in Table 2.

The main guidelines of this approach are [15]:

- The nodes in the LHS will be identified by consecutive numbers. These numbers make possible to identify the respective nodes in the RHS.
- All the properties of the different nodes will have an initial value. To point out that this value is undefined, the term ‘???’ is used.
- To refer to a LHS node in the RHS, the expression ‘match(x)’ will be used, being ‘x’ the number that identifies the node in the LHS.
- Likewise, when referring to an attribute of a LHS node, the dot notation will be used, for example ‘match(x).name’.
- As the nodes in the LHS, the nodes in the RHS will be numbered. The next guidelines must be considered in relation with these numbers:

![Graph Transformation Diagram](image)

**Figure 5. User Service Actors and Conceptual User Services in the User Services Model mapped to User Service Actors and Composite or Basic Services in the Extended User Services Model.**

- If the same number appears in the Left and the Right Hand Side, the type of the node in the RHS will be the same of the respective node in the LHS.
- If a node in the RHS is identified with a number followed by an apostrophe (x’), the type of this node will be different from the type of the respective node in the LHS. All the connections with other elements will be preserved.
- If a number appears in the LHS but not in the RHS, the respective node from the LHS will be deleted, as well as the connections in which it participated.
In this work we have presented the model to model transformations needed to complete the Hypertext Modeling Method of MIDAS (HM³). Moreover, we have defined a new UML profile to support the Hypertext modeling and we have used this profile to specify the metamodels for the different models considered in HM³.

According to these guidelines, we have defined the graph rules for the HM³ transformations that were susceptible of being expressed by graph grammars. In figures 5 to 8 we present just some of these graph rules next to the respective definition rules in natural language. When referring to Activity Diagrams, the reader should be aware that we have used the UML metamodel for Activity Diagrams.

4. CONCLUSIONS
In this work we have presented the model to model transformations needed to complete the Hypertext Modeling Method of MIDAS (HM³). Moreover, we have defined a new UML profile to support the Hypertext modeling and we have used this profile to specify the metamodels for the different models considered in HM³.

Since this is the first approach to model transformation in HM³, we have defined the transformation rules in a declarative manner for later map them to graph rules with the intention of automating this graph rules using some of the existing facilities to automate graph transformations.

This work serves as a clear example of the value of model transformations in software development: the model to model transformations presented in this work complete the definition of the HM³ process, a contrasted and published method that
founds in model transformation the piece that remained to become a completely feasible methodology.

At the present time we are working in the integration of HM in a case tool under development in our research group and which its early functionalities have already been presented in previous works [21] [22]. Besides, the open issue of making automatic the graph transformations by using existing technologies like AGG will be tackled.

5. ACKNOWLEDGMENTS

This research is granted by the project: DAWIS (TIC 2002-04050-C02-01), financed by the Ministry of Science and Technology of Spain.

6. REFERENCES


