ISBN: 978-972-8939-38-0 © 2011 IADIS

TRANSFORMING UML-ACTIVITY DIAGRAMS INTO IMS-LEARNING DESIGN MANIFESTS USING XMI AND XSL

José R. Hilera, Luis De-Marcos, José M. Gutiérrez, S. Otón, Roberto Barchino,
Eva García, José A. Gutiérrez, José J. Martínez, Antonio García
Department of Computer Science, University of Alcalá, Alcalá de Henares (Madrid) - Spain

ABSTRACT

This paper presents XSL source code to automatically produce a Unit of Learning manifest with a standard IMS-LD format, from the activity diagram with UML format that represents its workflow. Besides, a complementary XSL source code that performs the reverse process, namely to automatically obtain the activity diagram from the unit of learning manifest, is presented; so re-engineering of learning processes is also facilitated.

KEYWORDS

Learning design, IMS-LD, UML, Activity diagram, XMI, XSLT.

1. DESCRIPTION

The process for transforming models outlined in figure 1 has been implemented. An UML (OMG, 2009) modeling tool is used to create an activity diagram of the course that is going to be designed; and then the diagram is exported to a XMI (OMG, 2007) file, generating a *course.xmi* file. Next, a XSLT (W3C, 1999) transformation engine parses the code specified in the file UMLtoLD.xsl (excerpt in table 1 left), generating the *imsmanifest.xml* file of the unit of work. This later file is then integrated in the package generated by the IMS-LD (IMS, 2003) editor (*course.zip*) when the designing process of the unit of work has concluded. Reverse transformation is similar. In this case the transformation sheet *LDtoUML.xsl*, which is complementary to the previous one, is employed (excerpt in table 1 right).

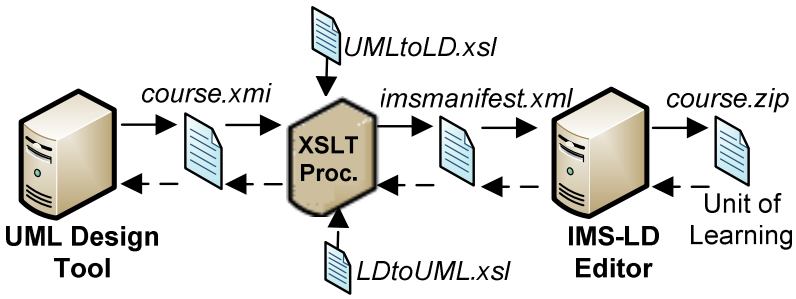

Figure 1. Transforming UML and IMS-LD models

As it can be observed in table 1 left, the code loops through the input UML document locating those that represent activities ("uml:CallOperationAction" in this simple code, but it can be extended checking other possible labels representing simple activities or actions, as "uml:CallBehaviorAction" or, in general, "uml:ActivityNode"), generating the corresponding activities of the IMS-LD document in the required output format. Then activity structures (<activity-structure> nodes) that represent the concatenation of activities, using sequences or alternative paths, are generated. Due to space restrictions the complete source code cannot be included but it is accessible at www.cc.uah.es/hilera/sw/uml_ld.zip. In the complementary transformation (table 1 right), activity-structures are located in the input file and a new UML "CallOperationAction" node is

generated for each activity. References to incoming and outgoing flows to other activities must be included in the last section (edges) of the file too. Its generation is not a trivial matter as it requires to create local variables and to use predefined functions, like *position()* or *last()*, among others.

Table 1. Extracts from the XSL transformation Sheets

Archive *UMLtoLD.xsl*

```
<?xml version="1.0" encoding="UTF-8" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:fo="http://www.w3.org/1999/XSL/Format"
xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
xmlns:uml="http://schema.omg.org/spec/UML/2.1.2"
 xmi:version="2.1">
 <xsl:output method="xml" />
 <xsl:template match="/">
  <xsl:apply-templates select="xmi:XMI" />
 </xsl:template>
 <xsl:template match="xmi:XMI">
  <imscp:manifest
   xmlns:imscp="http://www.imsglobal.org/xsd/....
   ...
   <imscp:organizations>
    <imsld:learning-design>
    ...
     <imsld:components>
      <imsld:activities>
       <xsl:for-each  select="uml:Package/
        packagedElement/node">
        <xsl:if test="@xmi:type=
         'uml:CallOperationAction'">
         <imsld:learning-activity>
          <xsl:attribute name="identifier">
           <xsl:value-of select="@xmi:id" />
          </xsl:attribute>
          <imsld:title>
            <xsl:value-of select="@name" />
          </imsld:title>
         </imsld:learning-activity>
        </xsl:if>
       </xsl:for-each>


       <imsld:activity-structure>
        ... Generation of sequence and selection
            structures
       </imsld:activity-structure>


      </imsld:activities>
     </imsld:components>
    </imsld:learning-design>
   </imscp:organizations>
   <imscp:resources>
    ...
   </imscp:resources>
  </imscp:manifest>
 </xsl:template>
</xsl:stylesheet>
```

Archive *LDtoUML.xsl*

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xsl:stylesheet version="1.0"
 xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
 xmlns:fo="http://www.w3.org/1999/XSL/Format"
 xmlns:imsld="http://www.imsglobal.org/xsd/imsld_v1p0"
 xmlns:imscp="http://www.imsglobal.org/xsd/imscp_v1p1">
 <xsl:output method="xml" />
 <xsl:template match="/">
  <xsl:apply-templates select="imscp:manifest" />
 </xsl:template>
 <xsl:template match="imscp:manifest">
  <xmi:XMI
   xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"
   xmlns:uml="http://schema.omg.org/spec/UML/2.1.2"
   xmi:version="2.1">
   <uml:Package>
    ...
    <packagedElement xmi:type="uml:Activity">
     ...
     <xsl:for-each select="imscp:organizations/imsld:Le
      arning-design/imsld:components/imsld:activities">
      <xsl:for-each select="imsld:activity-structure">
       ...
       <xsl:if test="@structure-type='sequence'">
        <xsl:for-each select="imsld:learning-activity-ref">
        <node xmi:type="uml:CallOperationAction">
         <xsl:attribute name="xmi:id">
          <xsl:value-of select="@ref" />
         </xsl:attribute>
         <xsl:attribute name="name">
          <xsl:value-of select="../../imsld:learning-
           activity[@identifier=@ref]/imsld:title" />
         </xsl:attribute>
         <incoming>
         <xsl:if test="position()='1'">
          ...Generation reference to UML initial node
         </xsl:if>
        </incoming>
        <outgoing>
         <xsl:if test="position()!=last()">
          ...Generation of reference to final UML node
         </xsl:if>
        </outgoing>
       </node>
       <edge xmi:type="uml:ControlFlow">
        ... Generation of the node sequence
       </edge>
      </xsl:if>
     </xsl:for-each>
    </packagedElement>
   </uml:Package>
  </xmi:XMI>
 </xsl:template>
</xsl:stylesheet>
```

2. CONCLUSIONS

Although there are learning design editors that display a graphical representation of the workflow of units of learning (for example LAMS: http://www.lamsfoundation.org), many designers has used and still use UML modeling tools to design learning actions. This is due to the fact that there are many free and open source

UML tools, but also because UML diagrams can be exported to the standard XMI format which facilitates diagrams interchange among compliant tools.

In this paper we have presented an open source program that aims to help designers, generating automatically the first version of the manifest of the learning unit, including its workflow and the involved activities. A complementary program that infers the manifest from a previously existing unit of learning has also been presented. It is also possible to visualize or modify it using an UML modeling tool.

At the moment, the software developed only support the transformation of sequential and conditional structures, but in the future these structures will be extended with activity diagram transitions and parallel activities, in order to be transformed to appropriate LD structures. Authors are now working in this way to improve the transformation style-sheets and to develop a Java (desktop and web) application with a friendly user interface that does not make necessary to use the command line, and that also works directly over complete units of learning in .zip files (not just imsmanifest.xml files). A web service is also being developed to provide both transformation functionalities through SOAP protocol.

ACKNOWLEDGEMENT

This research is funded by the Government of Madrid Autonomous Community and University of Alcalá (grant CCG10-UAH/TIC-5915). Authors also want to acknowledge support from the TIFyC research group.

REFERENCES

IMS (2003). *IMS Learning Design Specification*. IMS Global Learning Consortium, Burlington, USA (2003). http://www.imsglobal.org/ learningdesign.

OMG (2007). *XML Metadata Interchange (XMI)*. Object Management Group. http://www.omg.org/spec/XMI/.

OMG (2009). *Unified Modeling Language (UML)*. Object Management Group. http://www.omg.org/spec/UML/.

W3C (1999). *XSL Transformations (XSLT)*. World Wide Web Consortium (1999). http://www.w3.org/TR/xslt/

AUTHOR INDEX