

Language Specification (Appendix B)

Deliverable 2.2

Owners	Alessandra Cavarra and Jim Davies
Approvers	Alan Hartman, Ian Craggs, Laurent Mounier, Klaudia Dussa-Zieger
Status	public
Date	08/08/2001

AGEDIS

Contents

A Modelling Language: XML	3
A.1 An XML schema	3
A.2 Language specific abstract types	4
A.3 Language specific elements	5
A.4 Object related types and elements	8
A.5 Action related types and elements	9
A.6 Event related types and elements	10
A.7 State machine related types	11
A.8 Extras for IF	16

A Modelling Language: XML

The following XML schema describes the pre-release version of the modelling language; we plan to update it with incorporate explicit tags for IF constructs.

A.1 An XML schema

We will define a subset of the UML meta-model, in XML, that can be created and manipulated with (at most) simple extensions to the current tools. A key feature of this subset, and the following schema, is that it should be readable and accessible to anyone familiar with the meta-model. It contains strictly less information than any of the current tool formats. Of course, appropriate XSLTs can be used to translate to and from XML.

```
<xsd:element name="Model">
  <xsd:complexType>
    <xsd:sequence minOccurs="0" maxOccurs="unbounded">
      <xsd:element ref="Class" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="ObjectDiagram" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element ref="StateMachine" minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="IFText">
        <xsd:complexType>
          <xsd:sequence>
            <xsd:element name="IFFunction"/>
          </xsd:sequence>
        </xsd:complexType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="Class">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="Class"/>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
<xsd:element name="EnumerationClass" type="EnumerationClass"
substitutionGroup="Class"/>
<xsd:element name="SignalClass" type="SignalClass"
substitutionGroup="Class"/>
<xsd:element name="ObjectDiagram" type="ObjectDiagram"/>
<xsd:element name="StateMachine" type="StateMachine"/>
```

A.2 Language specific abstract types

```
<xsd:complexType name="ClassModifier" abstract="true">
  <xsd:annotation>
```

This represents the extra information which a specific programming language might add to the definition of a class: inheritance, whether it is an interface, etc.

```
  </xsd:annotation>
</xsd:complexType>
<xsd:complexType name="AttributeModifier" abstract="true">
  <xsd:annotation>
```

This defines the extra information which a specific language implementation may contribute to an attribute. Things such as modifiers, or initial values, can be defined here.

```
  </xsd:annotation>
</xsd:complexType>
<xsd:complexType name="ConstructorSignatureModifier" abstract="true">
  <xsd:annotation>
```

This defines the extra information which a specific language implementation may add to a constructor.

```
  </xsd:annotation>
</xsd:complexType>
<xsd:complexType name="MethodSignatureModifier" abstract="true">
  <xsd:annotation>
```

This defines the extra information which a specific language implementation may add to a method.

```
  </xsd:annotation>
</xsd:complexType>
<xsd:complexType name="ParameterModifier" abstract="true">
  <xsd:annotation>
```

This acts as a placeholder. Derived types will hold extra language specific information.

```
  </xsd:annotation>
</xsd:complexType>
<xsd:complexType name="ObjectModifier" abstract="true"/>
<xsd:complexType name="ExecutableExpression" abstract="true">
  <xsd:annotation>
```

This is another placeholder.

```
  </xsd:annotation>
</xsd:complexType>
<xsd:complexType name="ConditionalExpression" abstract="true">
  <xsd:annotation>
```

This is another placeholder.

```

</xsd:annotation>
</xsd:complexType>
<xsd:complexType name="TimeExpression" abstract="true">
  <xsd:annotation>

```

This is yet another placeholder.

```

</xsd:annotation>
</xsd:complexType>
<xsd:complexType name="ReferenceExpression" abstract="true">
  <xsd:annotation>

```

And again, a placeholder.

```

</xsd:annotation>
</xsd:complexType>

```

A.3 Language specific elements

These are abstract and should be replaced using substitution groups, such as `JavaClassModifiers` for `ClassModifier`.

```

<xsd:element name="ClassModifier" type="ClassModifier" abstract="true"/>
<xsd:element name="AttributeModifier" type="AttributeModifier"
abstract="true"/>
<xsd:element name="ConstructorSignatureModifier"
type="ConstructorSignatureModifier" abstract="true"/>
<xsd:element name="MethodSignatureModifier" type="MethodSignatureModifier"
abstract="true"/>
<xsd:element name="ParameterModifier" type="ParameterModifier"
abstract="true"/>
<xsd:element name="ObjectModifier" type="ObjectModifier" abstract="true"/>
<!------>
<!--Class specific types: -->
<xsd:complexType name="Class">
  <xsd:annotation>

```

This describes the information stored by a class.

```

</xsd:annotation>
<xsd:sequence>
  <xsd:element name="Name" type="xsd:string"/>
  <xsd:element name="Attribute" type="Attribute" minOccurs="0"
maxOccurs="unbounded"/>

```

```

    <xsd:element name="AbstractMethodSignature"
type="AbstractMethodSignature" minOccurs="0" maxOccurs="unbounded"/>
    <xsd:element name="ClassModifier" type="ClassModifier" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:ID" use="required"/>
</xsd:complexType>
<xsd:complexType name="ClassIDREF">
  <xsd:annotation>

```

This references a class.

```

  </xsd:annotation>
  <xsd:attribute name="idref" type="xsd:IDREF" use="required"/>
</xsd:complexType>
<xsd:complexType name="EnumerationClass">
  <xsd:annotation>

```

This is an enumeration class.

```

  </xsd:annotation>
  <xsd:complexContent>
    <xsd:restriction base="Class">
      <xsd:sequence>
        <xsd:element name="Name" type="xsd:string"/>
        <xsd:element name="Attribute" type="Attribute" minOccurs="0"
maxOccurs="unbounded"/>
        <xsd:element name="AbstractMethodSignature"
type="AbstractMethodSignature" minOccurs="0" maxOccurs="0"/>
        <xsd:element name="AttributeModifier" type="AttributeModifier"
minOccurs="0"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SignalClass">
  <xsd:annotation>

```

A signal class has attributes for the names and types of the parameters which may be passed by a signal.

```

  </xsd:annotation>
  <xsd:complexContent>
    <xsd:restriction base="Class">
      <xsd:sequence>
        <xsd:element name="Name" type="xsd:string"/>
        <xsd:element name="Attribute" type="Attribute" minOccurs="0"
maxOccurs="unbounded"/>

```

```

    <xsd:element name="AbstractMethodSignature"
type="AbstractMethodSignature" minOccurs="0" maxOccurs="0"/>
    <xsd:element name="AttributeModifier" type="AttributeModifier"
minOccurs="0"/>
    </xsd:sequence>
  </xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="Attribute">
  <xsd:annotation>

```

A class attribute.

```

  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:string"/>
    <xsd:element name="TypeIDREF" type="ClassIDREF"/>
    <xsd:element name="AttributeModifier" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:ID" use="required"/>
</xsd:complexType>
<xsd:complexType name="AttributeIDREF">
  <xsd:attribute name="idref" type="xsd:IDREF" use="required"/>
</xsd:complexType>
<xsd:complexType name="AbstractMethodSignature" abstract="true">
  <xsd:annotation>

```

This is required to make the substitution group for method and constructor signatures work.

```

  </xsd:annotation>
  <xsd:attribute name="id" type="xsd:ID" use="required"/>
</xsd:complexType>
<xsd:complexType name="MethodSignatureIDREF">
  <xsd:attribute name="idref" type="xsd:IDREF" use="required"/>
</xsd:complexType>
<xsd:complexType name="MethodSignature">
  <xsd:annotation>

```

This describes a method signature.

```

  </xsd:annotation>
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:string"/>
    <xsd:element name="InputParameters" minOccurs="0">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="Parameter" type="Parameter"/>

```

```

    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="OutputTypeIDREF" type="ClassIDREF"/>
<xsd:element name="MethodSignatureModifier"
type="MethodSignatureModifier" minOccurs="0"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ConstructorSignature">
  <xsd:annotation>

```

Describes a constructor signature.

```

</xsd:annotation>
<xsd:sequence>
  <xsd:element name="InputParameters" minOccurs="0">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Parameter" type="Parameter"/>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element name="ConstructorSignatureModifier"
type="ConstructorSignatureModifier" minOccurs="0"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Parameter">
  <xsd:annotation>

```

Describes what a parameter is.

```

</xsd:annotation>
<xsd:sequence>
  <xsd:element name="TypeIDREF" type="ClassIDREF"/>
  <xsd:element name="Name" type="xsd:string"/>
  <xsd:element name="ParameterModifier" type="ParameterModifier"
minOccurs="0"/>
</xsd:sequence>
</xsd:complexType>

```

A.4 Object related types and elements

```

<xsd:complexType name="Object">
  <xsd:sequence>
    <xsd:element name="ClassIDREF" type="ClassIDREF"/>

```

```
<xsd:element name="AttributeObject" minOccurs="0" maxOccurs="unbounded">
  <xsd:annotation>
```

Each attribute object pair links an attribute of this object with a reference to another object.

```
  </xsd:annotation>
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="AttributeIDREF" type="AttributeIDREF"/>
      <xsd:element name="ObjectIDREF" type="ObjectIDREF"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element name="StateIDREF" type="StateVertexIDREF" minOccurs="0"/>
<xsd:element name="ObjectModifier" type="ObjectModifier" minOccurs="0"/>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:ID" use="required"/>
</xsd:complexType>
<xsd:complexType name="ObjectIDREF">
  <xsd:attribute name="idref" type="xsd:IDREF" use="required"/>
</xsd:complexType>
<xsd:complexType name="ObjectDiagram">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:string" minOccurs="0"/>
    <xsd:element name="Object" type="Object" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="SequenceOfReferenceExpressions">
  <xsd:sequence minOccurs="0" maxOccurs="unbounded">
    <xsd:element name="ReferenceExpression" type="ReferenceExpression"/>
  </xsd:sequence>
</xsd:complexType>
```

A.5 Action related types and elements

```
<xsd:complexType name="Action" abstract="true"/>
<xsd:complexType name="InterpretedAction">
  <xsd:complexContent>
    <xsd:extension base="Action">
      <xsd:sequence>
        <xsd:element name="TargetIDREF" type="ClassIDREF"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
```

```

</xsd:complexType>
<xsd:complexType name="UninterpretedAction">
  <xsd:complexContent>
    <xsd:extension base="Action">
      <xsd:sequence>
        <xsd:element name="ExecutableExpression" type="ExecutableExpression"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SendAction">
  <xsd:complexContent>
    <xsd:extension base="InterpretedAction">
      <xsd:sequence>
        <xsd:element name="SignalEvent" type="SignalEvent"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CallAction">
  <xsd:complexContent>
    <xsd:extension base="InterpretedAction">
      <xsd:sequence>
        <xsd:element name="MethodSignatureIDREF" type="MethodSignatureIDREF"/>
        <xsd:element name="SequenceOfReferenceExpressions"
type="SequenceOfReferenceExpressions"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

A.6 Event related types and elements

```

<xsd:complexType name="Event" abstract="true"/>
<xsd:complexType name="TimedEvent">
  <xsd:complexContent>
    <xsd:extension base="Event">
      <xsd:sequence>
        <xsd:element name="TimeExpression" type="TimeExpression"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

<xsd:complexType name="SignalEvent">
  <xsd:complexContent>
    <xsd:extension base="Event">
      <xsd:sequence>
        <xsd:element name="SignalEventTypeIDREF" type="ClassIDREF"/>
        <xsd:element name="References" type="SequenceOfReferenceExpressions"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CallEvent">
  <xsd:complexContent>
    <xsd:extension base="Event">
      <xsd:sequence>
        <xsd:element name="MethodSignatureIDREF" type="MethodSignatureIDREF"/>
        <xsd:element name="SequenceOfReferenceExpressions"
type="SequenceOfReferenceExpressions"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

A.7 State machine related types

```

<xsd:complexType name="Transition" abstract="true">
  <xsd:sequence>
    <xsd:element name="SourceIDREF" type="StateVertexIDREF"/>
    <xsd:element name="TargetIDREF" type="StateVertexIDREF"/>
    <xsd:element ref="Trigger" minOccurs="0"/>
    <xsd:element ref="Guard" minOccurs="0"/>
    <xsd:element ref="Effect" minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:ID" use="required"/>
</xsd:complexType>
<xsd:complexType name="InternalTransition">
  <xsd:complexContent>
    <xsd:restriction base="Transition">
      <xsd:sequence>
        <xsd:element name="SourceIDREF" type="StateVertexIDREF" minOccurs="0"
maxOccurs="0"/>
        <xsd:element name="TargetIDREF" type="StateVertexIDREF" minOccurs="0"
maxOccurs="0"/>
        <xsd:element ref="Trigger"/>

```

```

    <xsd:element ref="Guard" minOccurs="0"/>
    <xsd:element ref="Effect"/>
  </xsd:sequence>
  <xsd:attribute name="id" use="prohibited"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="TransitionIDREF"/>
<xsd:complexType name="TransitionIDREFS">
  <xsd:attribute name="idrefs" type="xsd:IDREFS" use="required"/>
</xsd:complexType>
<xsd:complexType name="StateVertex">
  <xsd:sequence>
    <xsd:element name="Name" type="xsd:string"/>
    <xsd:element name="IncomingTransitionIDREFS" type="TransitionIDREFS"
minOccurs="0"/>
    <xsd:element name="OutgoingTransitionIDREFS" type="TransitionIDREFS"
minOccurs="0"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:ID" use="required"/>
</xsd:complexType>
<xsd:complexType name="StateVertexIDREF">
  <xsd:attribute name="idref" type="xsd:IDREF" use="required"/>
</xsd:complexType>
<xsd:complexType name="PseudoState">
  <xsd:complexContent>
    <xsd:extension base="StateVertex"/>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="State">
  <xsd:complexContent>
    <xsd:extension base="StateVertex">
      <xsd:sequence>
        <xsd:element name="Entry" type="Action" minOccurs="0"/>
        <xsd:element name="Exit" type="Action" minOccurs="0"/>
        <xsd:element name="DeferrableEvents" type="Event" minOccurs="0"
maxOccurs="unbounded"/>
        <xsd:element name="InternalTransitions" type="InternalTransition"
minOccurs="0" maxOccurs="unbounded"/>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="FinalState">
  <xsd:complexContent>

```

```

    <xsd:restriction base="State">
      <xsd:sequence>
        <xsd:element name="Name" type="xsd:string"/>
        <xsd:element name="IncomingTransitionIDREFS" type="TransitionIDREFS"
minOccurs="0"/>
        <xsd:element name="OutgoingTransitionIDREFS" type="TransitionIDREFS"
minOccurs="0" maxOccurs="0"/>
        <xsd:element name="Entry" type="Action" minOccurs="0" maxOccurs="0"/>
        <xsd:element name="Exit" type="Action" minOccurs="0" maxOccurs="0"/>
        <xsd:element name="DeferrableEvents" type="Event" minOccurs="0"
maxOccurs="0"/>
        <xsd:element name="InternalTransitions" type="InternalTransition"
minOccurs="0" maxOccurs="0"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SimpleState">
  <xsd:complexContent>
    <xsd:extension base="State"/>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CompositeState">
  <xsd:complexContent>
    <xsd:extension base="State">
      <xsd:sequence>
        <xsd:element name="Concurrent" type="xsd:boolean" default="false"
minOccurs="0"/>
        <xsd:element name="SubVertices" minOccurs="0">
          <xsd:complexType>
            <xsd:choice maxOccurs="unbounded">
              <xsd:element name="Fork">
                <xsd:complexType>
                  <xsd:complexContent>
                    <xsd:extension base="PseudoState"/>
                  </xsd:complexContent>
                </xsd:complexType>
              </xsd:element>
              <xsd:element name="Join" type="PseudoState"/>
              <xsd:element name="Initial" type="PseudoState"/>
              <xsd:element name="Choice" type="PseudoState"/>
              <xsd:element name="Final" type="FinalState"/>
              <xsd:element name="Simple" type="SimpleState"/>
              <xsd:element name="Composite" type="CompositeState"/>
            </xsd:choice>
          </xsd:complexType>
        </xsd:element>
      </xsd:sequence>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

    </xsd:complexType>
  </xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SubmachineState">
  <xsd:complexContent>
    <xsd:restriction base="CompositeState">
      <xsd:sequence>
        <xsd:element name="Name" type="xsd:string"/>
        <xsd:element name="IncomingTransitionIDREFS" type="TransitionIDREFS"
minOccurs="0"/>
        <xsd:element name="OutgoingTransitionIDREFS" type="TransitionIDREFS"
minOccurs="0"/>
        <xsd:element name="Entry" type="Action" minOccurs="0" maxOccurs="0"/>
        <xsd:element name="Exit" type="Action" minOccurs="0" maxOccurs="0"/>
        <xsd:element name="DeferrableEvents" type="Event" minOccurs="0"
maxOccurs="0"/>
        <xsd:element name="InternalTransitions" type="InternalTransition"
minOccurs="0" maxOccurs="0"/>
        <xsd:element name="Concurrent" type="xsd:boolean" minOccurs="0"
maxOccurs="0"/>
        <xsd:element name="SubVertices" minOccurs="0" maxOccurs="0">
          <xsd:complexType/>
        </xsd:element>
        <xsd:element name="StateMachineIDREF" type="StateMachineIDREF"/>
      </xsd:sequence>
    </xsd:restriction>
  </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="StateMachine">
  <xsd:sequence>
    <xsd:element name="ClassIDREF" type="ClassIDREF" minOccurs="0"/>
    <xsd:element name="Composite" type="CompositeState"/>
    <xsd:element name="Transition" type="Transition" minOccurs="0"
maxOccurs="unbounded"/>
  </xsd:sequence>
  <xsd:attribute name="id" type="xsd:ID" use="required"/>
</xsd:complexType>
<xsd:complexType name="StateMachineIDREF"/>
<xsd:element name="Trigger">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="Event">

```

```

    <xsd:choice>
      <xsd:element name="TimedEvent" type="TimedEvent"/>
      <xsd:element name="SignalEvent" type="SignalEvent"/>
      <xsd:element name="CallEvent" type="CallEvent"/>
    </xsd:choice>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="Effect">
  <xsd:complexType>
    <xsd:complexContent>
      <xsd:extension base="Action">
        <xsd:choice maxOccurs="unbounded">
          <xsd:element name="SendAction" type="SendAction"/>
          <xsd:element name="CallAction" type="CallAction"/>
          <xsd:element name="UninterpretedAction" type="UninterpretedAction"/>
        </xsd:choice>
      </xsd:extension>
    </xsd:complexContent>
  </xsd:complexType>
</xsd:element>
<xsd:element name="Guard">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element name="ConditionalExpression" type="ConditionalExpression"/>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<!-->
<!--This defines basic modifiers required - these are empty-->
<xsd:complexType name="UMLTimeExpression">
  <xsd:annotation>

```

A string

```

</xsd:annotation>
<xsd:complexContent>
  <xsd:extension base="TimeExpression">
    <xsd:sequence>
      <xsd:element name="UML" type="xsd:string"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>

```

```
<xsd:complexType name="UMLReferenceExpression">
  <xsd:annotation>
```

A string

```
</xsd:annotation>
<xsd:complexContent>
  <xsd:extension base="ReferenceExpression">
    <xsd:sequence>
      <xsd:element name="UML" type="xsd:string"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
```

A.8 Extras for IF

```
<xsd:complexType name="IFExecutableExpression">
  <xsd:annotation>
```

Used for assignment in IF - still a string

```
</xsd:annotation>
<xsd:complexContent>
  <xsd:extension base="ExecutableExpression">
    <xsd:sequence>
      <xsd:element name="IF" type="xsd:string"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="IFConditionalExpression">
  <xsd:annotation>
```

Used for guards in IF - still a string

```
</xsd:annotation>
<xsd:complexContent>
  <xsd:extension base="ConditionalExpression">
    <xsd:sequence>
      <xsd:element name="IF" type="xsd:string"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="IFReferenceExpression">
  <xsd:annotation>
```

A string

```
</xsd:annotation>
<xsd:complexContent>
  <xsd:extension base="ReferenceExpression">
    <xsd:sequence>
      <xsd:element name="UML" type="xsd:string"/>
    </xsd:sequence>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:schema>
```