BEHAVIOR3D:

An XML-Based Framework for 3D Graphics Behavior

Raimund Dachselt, Enrico Rukzio
Dresden University of Technology, Multimedia Technology Group

ACM Web3D 2003 Symposium, March 9-12, 2003, St. Malo, France

Outline

- Motivation and Vision
- Related Work
  - X3D: Behavior Definitions and Extensibility
- BEHAVIOR3D
  - Basic Node Concept
  - Declaration, Usage, Implementation
  - Demonstration
- Conclusion & Future Work

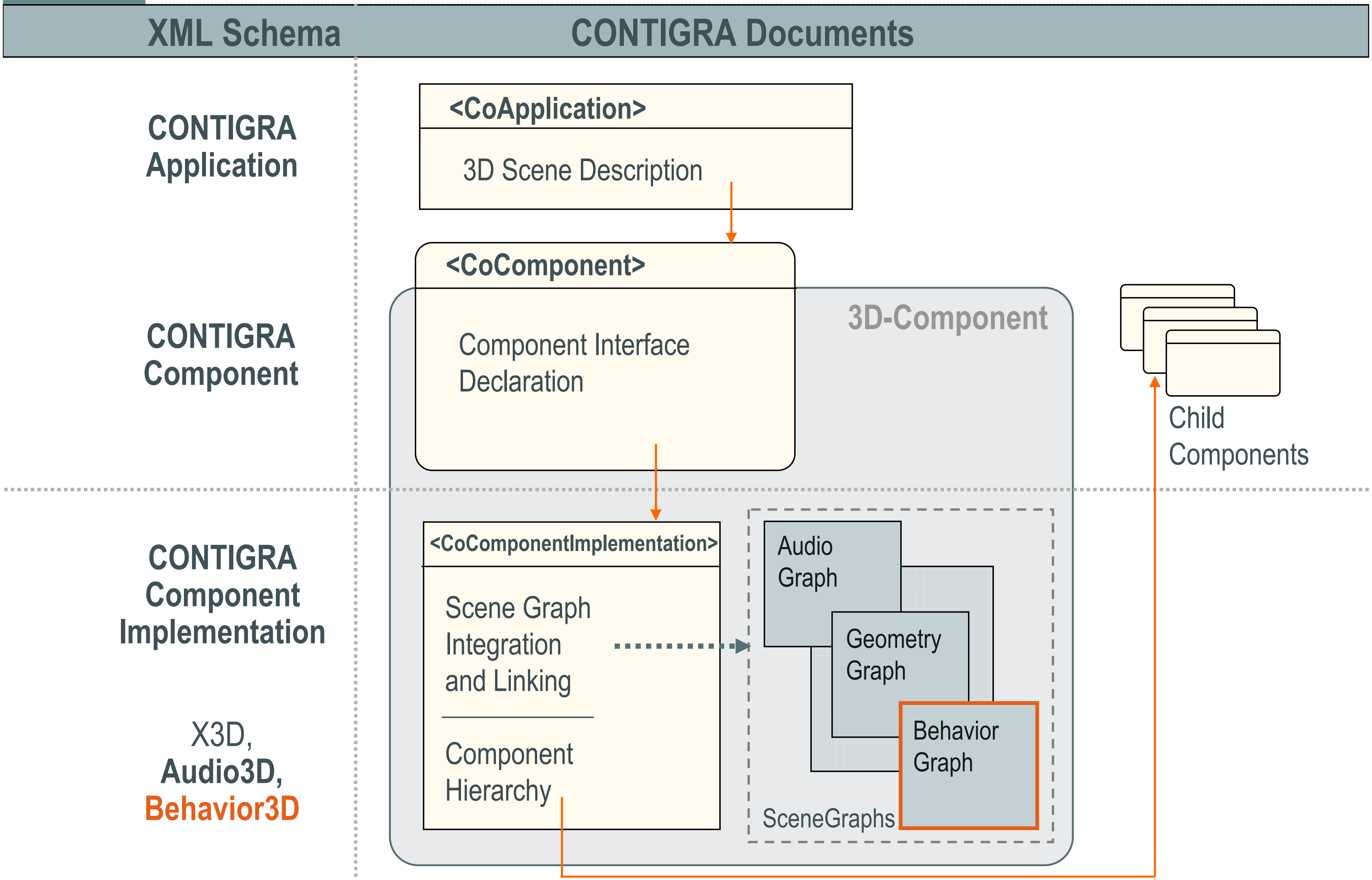
Motivation and Vision

- Current Situation
  - Increasing number of 3D enhanced Web applications
  - Need for media-rich and highly interactive content
  - Variety of 3D formats, associated modeling and authoring tools
- ☹ Problems
  - Tools & behavior definitions tailored to specific domains
  - Limited in producing interactive and dynamic scenes, basically simple animation and behaviors
  - Complex behaviors & extensions only through script languages
  - Non-programmers remain excluded, authoring still tedious work
  - Few concepts of reusing behavior building blocks

Motivation and Vision

- ☺ Future Vision & Requirements
  - Extensible, flexible and unifying description format for 3D graphics behaviors and interactions
  - Integrate well into X3D standard
  - Rich and extensible set of predefined and classified behavior modules → reuse of high-level 3D Behaviors
  - Reduction of programming efforts → declarative format (XML)
- CONTIGRA - Framework [Dachselt et al. 2002]
  - Document-centered, declarative 3D component architecture
  - XML-documents describe interfaces, implementation, configuration, and assembly of components
  - High-level view, hides scene graph details, based on X3D

CONTIGRA



XML Schema

CONTIGRA Documents

CONTIGRA Application

<CoApplication>

3D Scene Description

CONTIGRA Component

<CoComponent>

Component Interface Declaration

3D-Component

Child Components

CONTIGRA Component Implementation

X3D, **Audio3D**, **Behavior3D**

<CoComponentImplementation>

Scene Graph Integration and Linking

Component Hierarchy

Audio Graph

Geometry Graph

Behavior Graph

SceneGraphs

Related Work

- Four levels of behavior [Roehl 1995]
- Independent behavior graph [Döllner & Hinrichs 1998]
- Declarative languages (partly XML-based)
  - **VRML97, X3D** as a basis: built-in nodes + behavior extensions, e.g. [Seidman 1998]
  - **SMIL 2.0** - intuitive time and animation concepts, also sketch of integration into X3D [Kemkes 2001]
  - **Viewpoint** - scene interactors, state machine paradigm
- Object-Oriented Extensions Working Group [OOE-VRML] and VRML++ [Diehl 1997]

Related Work: VRML97 / X3D

- Built-in behavior-related nodes
  - For defining simple object animations and interactions
    - time, sensors, interpolators, triggers, and sequencers
  - X3D-Components: functionally related X3D objects/nodes
    - Environmental Sensor, Event Utilities, Interpolation, Key device sensor, Networking, Point Device Sensor, Scripting, Time
  - Steps towards node hierarchy: X3D-Schema, SAI
  - Insufficient for complex animations, state-based modeling

Related Work: VRML97 / X3D

```
<ExternProtoDeclare name="AnimateRotation" url="File.x3d">
        <field accessType="field" name="key" type="Floats"/>
        <field accessType="field" name="to" type="Rotations"/>
        …
</ExternProtoDeclare>
…
<ProtoInstance name="AnimateRotation">
        <fieldValue name="key" value="0 1"/>
        <fieldValue name="to" value="1 0 0 -1.7, 1 0 0 0"/>
</ProtoInstance>
```

BEHAVIOR3D - Nodes

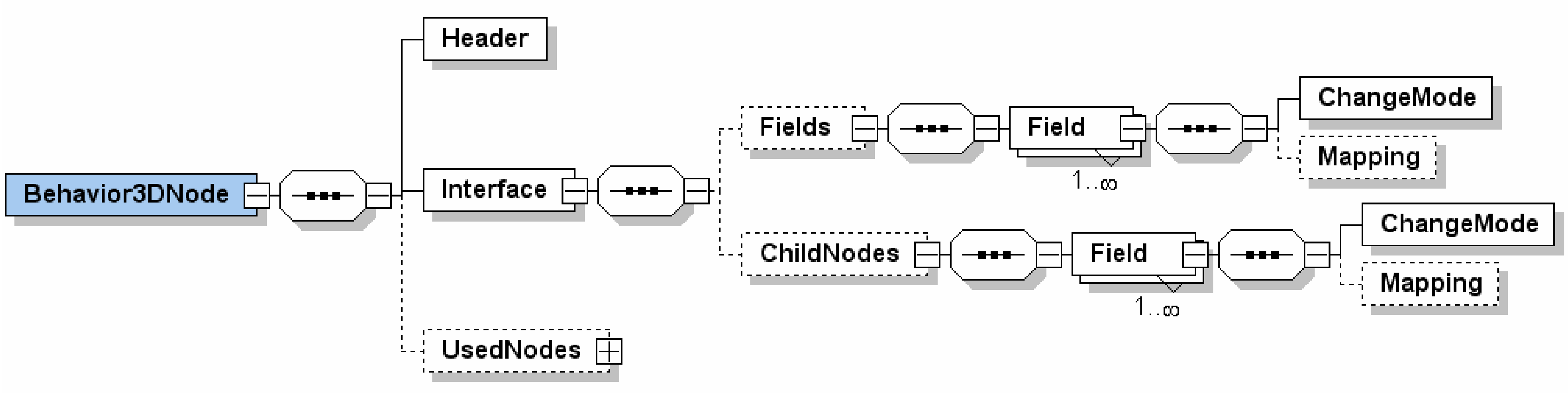
- Basic Node Concept

	Combinations			corresponds to X3D field access type
	configurable	***receives Events***	***generates Events***	
1	false	false	false	-
2	false	false	true	***outputOnly*** *(eventOut)*
3	false	true	false	***inputOnly*** *(eventIn)*
4	false	true	true	-
5	true	false	false	***initializeOnly*** *(field)*
6	true	false	true	-
7	true	true	false	-
8	true	true	true	***inputOutput*** *(exposedField)*

  - Improved field concept:
    name, type, possible default value, 3 change modes

BEHAVIOR3D - Nodes

- Declaration of new Behavior3D Nodes
  - XML Schema grammar *Behavior3DNode*



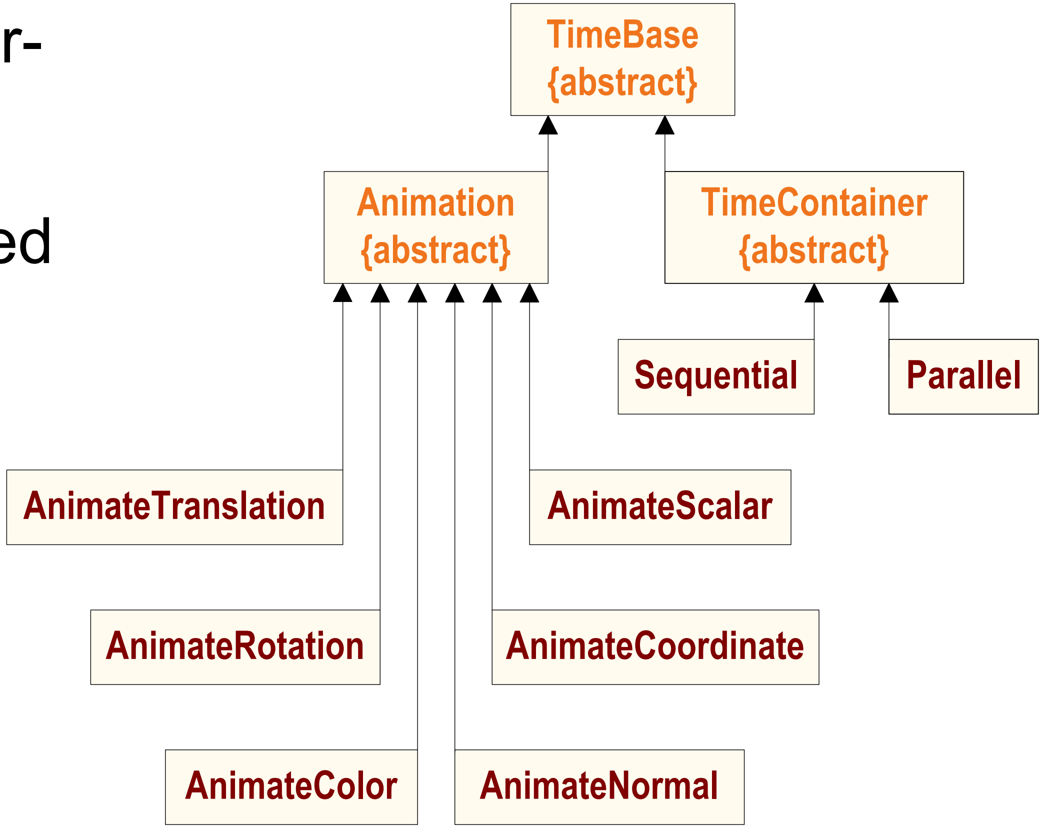
Header:	name, documentation
Fields:	none-node datatypes (Color, Rotation)
ChildNodes:	node datatypes (TimeBase)
UsedNodes:	node composition

BEHAVIOR3D - Nodes

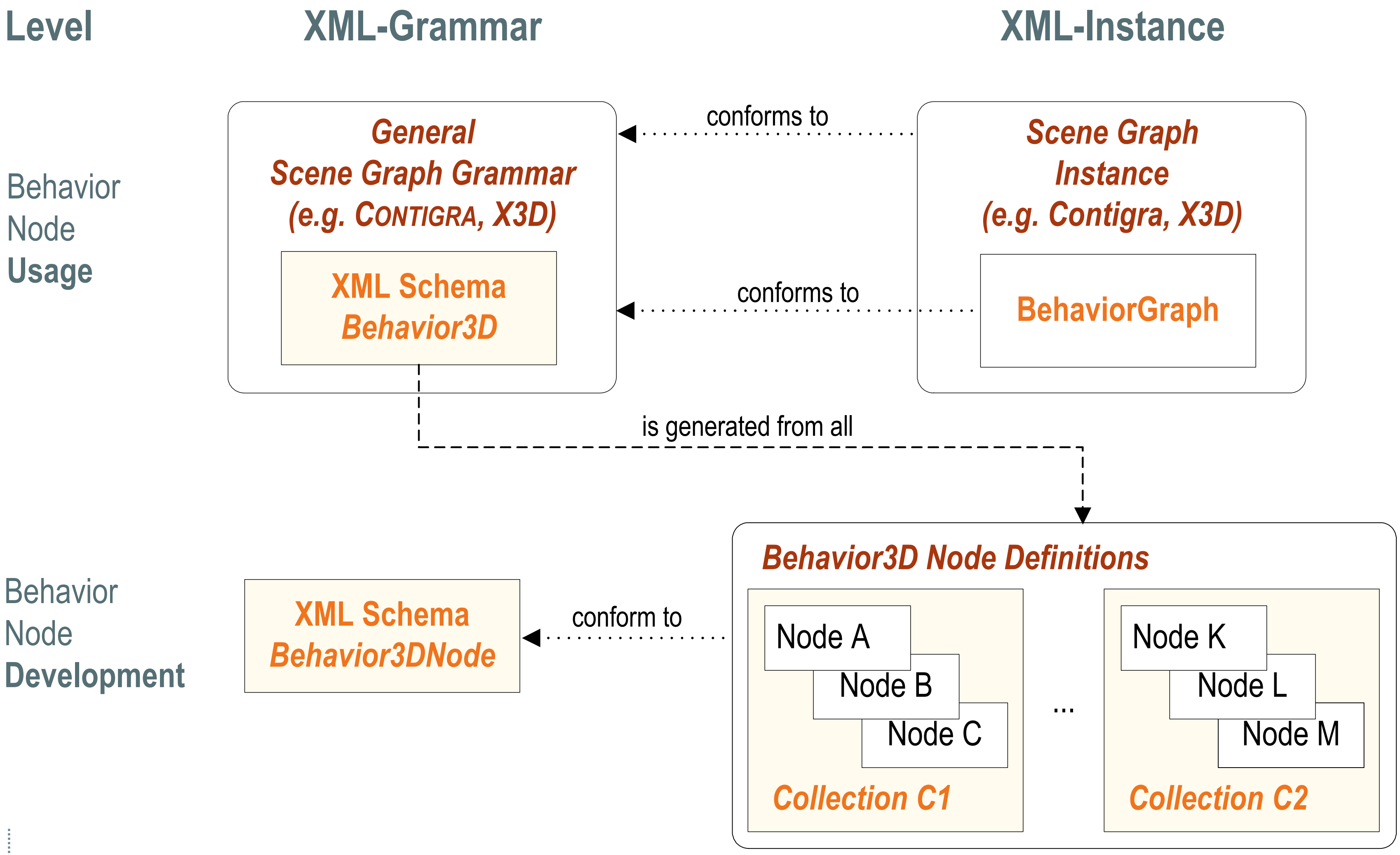
```
<Behavior3DNode>
  <Header name="TimeContainer"/>
  <Interface nodeType="abstract" extends="TimeBase">
    <ChildNodes>
      <Field dataType="TimeBase"
                minOccurs="0" maxOccurs="unbounded">
        <ChangeMode configurable="true" receivesEvents="false"
                          generatesEvents="false"/>
      </Field>
    </ChildNodes>
  </Interface>
</Behavior3DNode>
```

BEHAVIOR3D - Collections

- Collections
  - Group functionally and semantically related nodes
  - Include all behavior-related X3D nodes
  - Completely declared and implemented Collections: *StateMachine, Animation*



BEHAVIOR3D - Levels



Level

XML-Grammar

XML-Instance

Behavior Node **Usage**

General Scene Graph Grammar (e.g. CONTIGRA, X3D)

XML Schema *Behavior3D*

conforms to

Scene Graph Instance (e.g. Contigra, X3D)

BehaviorGraph

conforms to

is generated from all

Behavior Node **Development**

XML Schema *Behavior3DNode*

conform to

Behavior3D Node Definitions

Node A

Node B

Node C

Collection C1

...

Node K

Node L

Node M

Collection C2

BEHAVIOR3D - Levels

- Node Declaration

```
<Behavior3DNode>
  <Header name="AnimateRotation"/>
  <Interface nodeType="public" extends="Animation">
    <Fields>
      <Field name="key" dataType="Floats" default="[]">
        <ChangeMode configurable="true" receivesEvents="true"
         generatesEvents="true"/>
      </Field>
    </Fields>
  </Interface>
</Behavior3DNode>
```

BEHAVIOR3D - Levels

- Resulting Grammar

```
<element name="AnimateRotation" type="AnimateRotationType“
                substitutionGroup="Animation"/>

<complexType name="AnimateRotationType">
   <complexContent>
      <extension base="AnimationType">
         <attribute name="key" type="x3d:Floats"/>
         <attribute name="to" type="x3d:Rotations"/>
         <attribute name="by" type="x3d:Rotations"/>
      </extension>
   </complexContent>
</complexType>
```

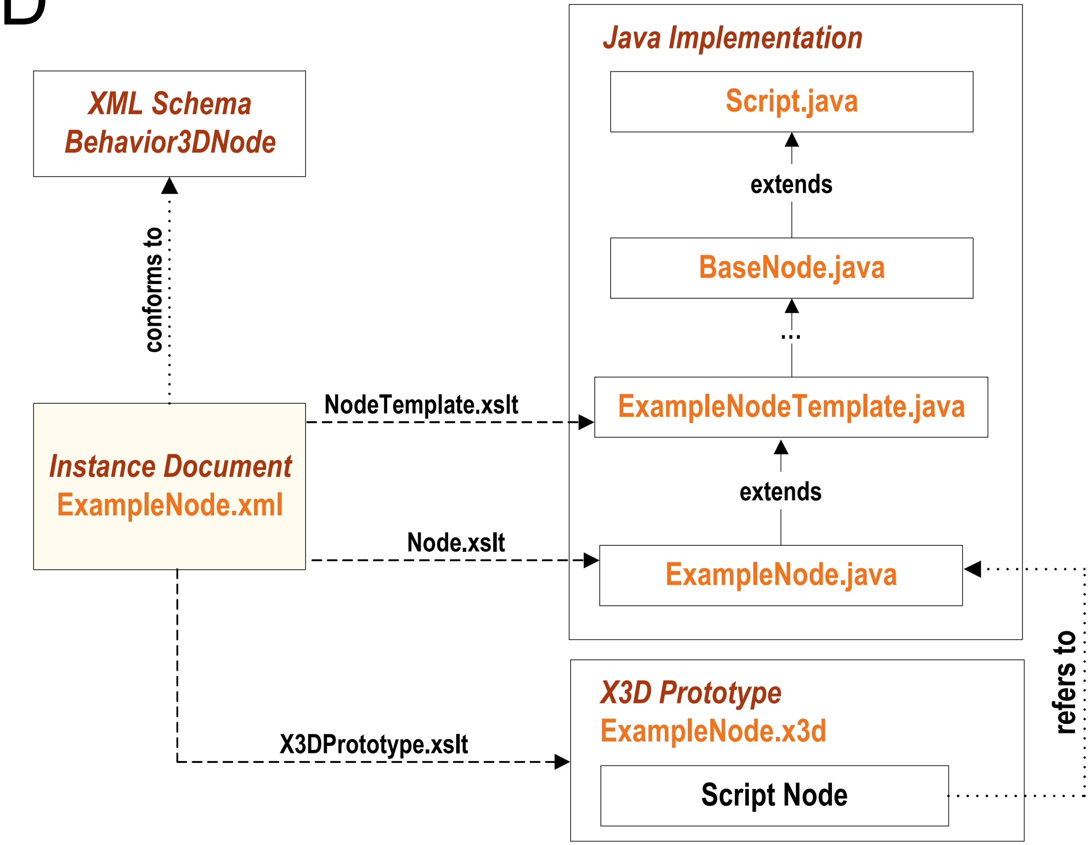
BEHAVIOR3D - Levels

- Node Usage

```
<Sequential begin="5.0">
        <AnimateRotation key="0 1" to="1 0 0 0, 1 0 0 -1.5"/>
        <AnimateRotation key="0 1" to=" 1 0 0 -1.5 , 1 0 0 0"/>
</Sequential>
```

BEHAVIOR3D - Implementation

- First implementation of Behavior3D nodes with VRML97/X3D



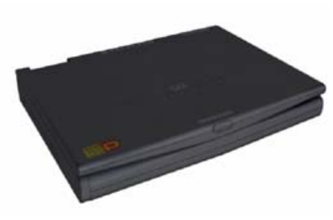
Demo

- Interactive Laptop
  - Entirely realized with Behavior3D nodes
  - Far easier and shorter coding than with X3D
  - Translated to VRML97/X3D with XSLT Stylesheets

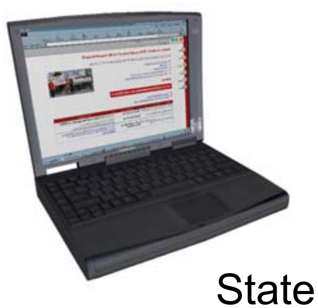
```
<AnimateRotation DEF="OpenLaptop"
 key="0 1" to="1 0 0 0, 1 0 0 -1.7"/>

<Sequential DEF="OpenKeyboard">
 <AnimateTranslation DEF="Open_Translation"
  key="0 1" to="0 0 0, 0 0.05 0" />
 <AnimateRotation DEF="Open_Rotation"
  key="0 1" to="1 0 0 0, 1 0 0 -1.5" />
</Sequential>
```

Demo

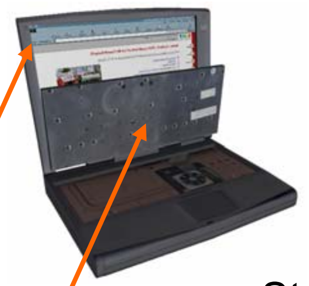
```
<StateMachine stateCount="3" transitions="“
 1 2  LCD_Sensor.touchTime  OpenLaptop.startTime,
 2 1  LCD_Sensor.touchTime  CloseLaptop.startTime,
 2 3  Keyboard_Sensor.touchTime OpenKeyboard.startTime,
 3 2  Keyboard_Sensor.touchTime CloseKeyboard.startTime"/>
```



State 1



State 2



State 3

```
<bno:TouchSensor DEF="LCD_Sensor"/>

<bno:TouchSensor DEF="Keyboard_Sensor"/>
```

Conclusion & Future Work

- Major Features
  - Inheritance, strong typing, polymorphism
  - Easy definition of new first-class nodes
  - Automated implementation-code generation
  - Smooth language integration through novel grammar generation mechanism
- Future Work
  - Visual Authoring tool for editing 3D graphics behavior
  - Sets of predefined behavior nodes (collections) to be extended. Candidates for X3D-components?
  - Dynamic scene graph grammar generation for X3D?

Discussion

Thank you for your attention!

www.CONTIGRA.com

Translation