

# SCXML 101

## An Introduction to State Chart XML

Rahul Akolkar



# Outline

- Introduction to State Chart XML (SCXML)
- Sample use cases for SCXML
- SCXML Extensibility
  - Data model
  - Custom actions
  - External communications module
- Specification Status
- Available implementations and tools

# Outline

- Introduction to State Chart XML (SCXML)
- Sample use cases for SCXML
- SCXML Extensibility
  - Data model
  - Custom actions
  - External communications module
- Specification Status
- Available implementations and tools

# State Chart XML (SCXML)

- **Generic and environment agnostic markup language for state machine definition**
- **Based on**
  - UML 2.0
  - Harel state transition tables
  - Oriented towards reactive systems
- **Powerful and generic controller with broad application**
  - Dialog flow in Voice applications
  - Interaction Manager for multimodal applications
  - Controller for multi-namespace documents (CDF type of documents)
  - Backend controller for business processes

# SCXML features

- States and transitions
  - Composite, Orthogonal and Final
  - Events, guards and target(s)
- Data model
  - Pluggable data representation, expression language
- Executable content
  - On entry, exit or transition
  - Extensible
- External communications module
  - Send, cancel
  - Invoke, finalize
- History



# Hello World and a transition

```
<scxml xmlns="http://www.w3.org/2005/07/scxml"  
  version="1.0" initial="hello">
```

```
  <state id="hello">
```

```
    <onentry>
```

```
      <log expr="Hello World" />
```

```
    </onentry>
```

```
    <transition target="done" />
```

```
  </state>
```

```
  <final id="done" />
```

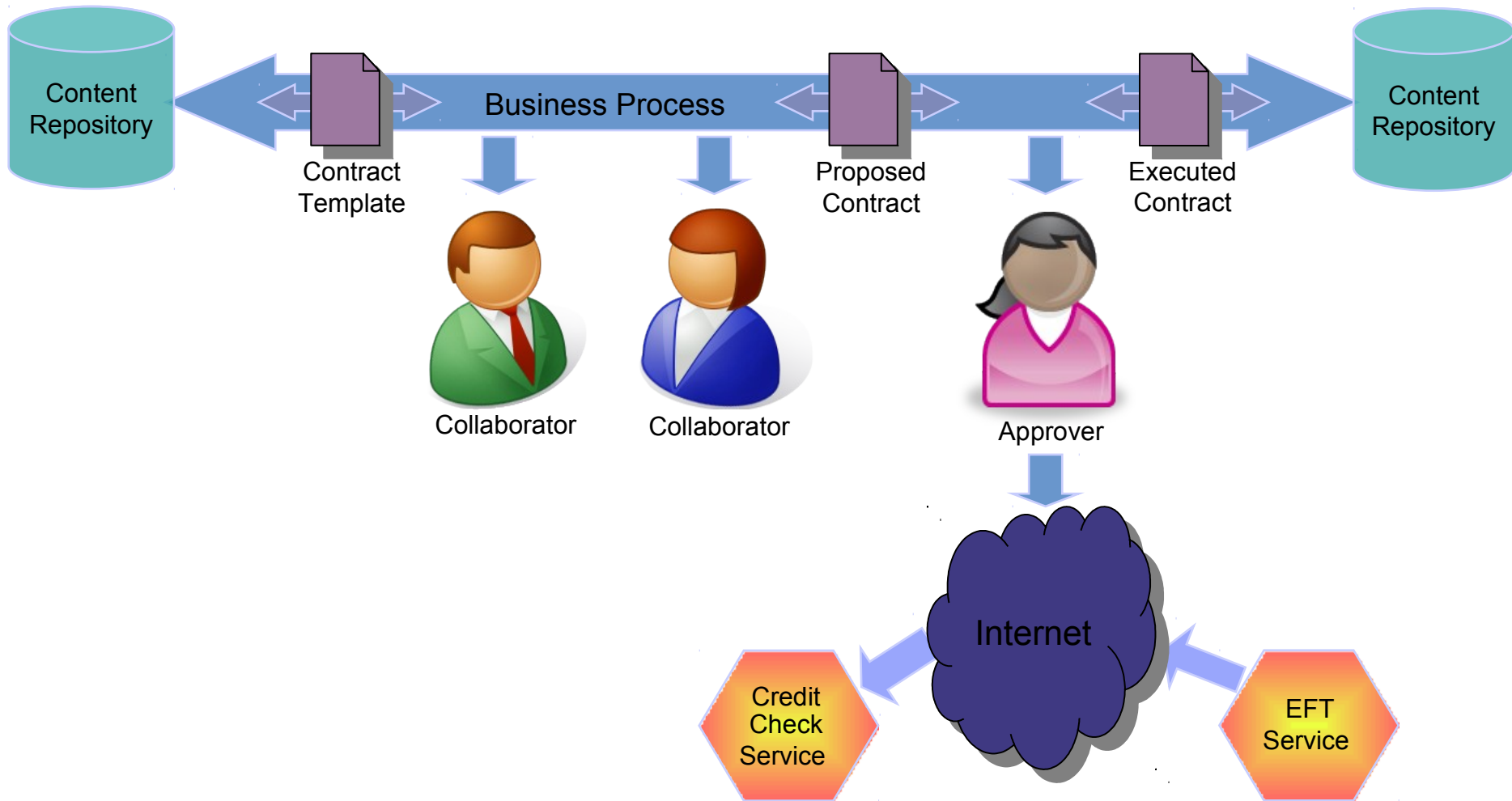
```
</scxml>
```



# Outline

- Introduction to State Chart XML (SCXML)
- **Sample use cases for SCXML**
- SCXML Extensibility
  - Data model
  - Custom actions
  - External communications module
- Specification Status
- Available implementations and tools

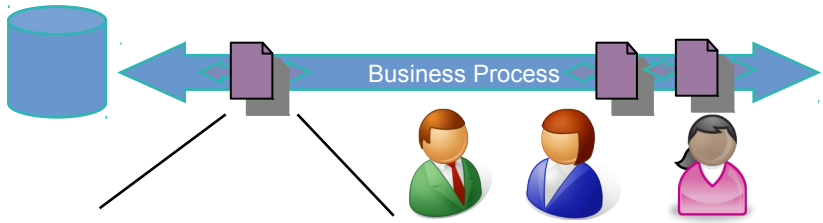
# Workflow - Collaborative Business Process Systems



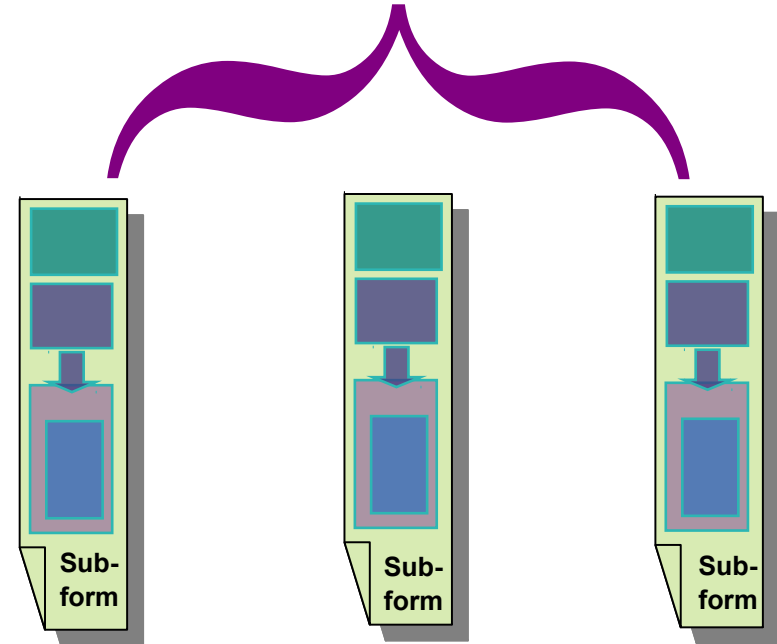
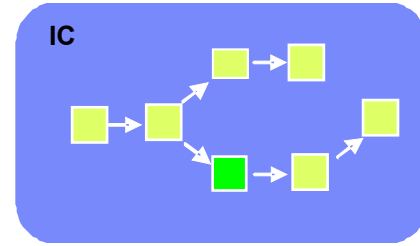


# Interactive Web Documents

## Interaction Controllers

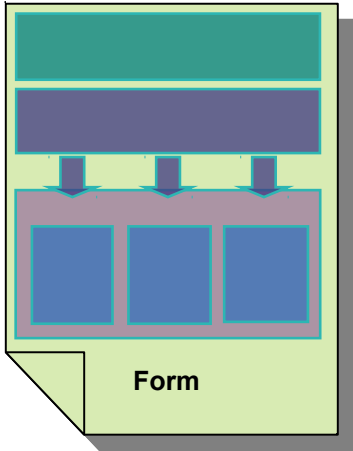


## Interaction Controller (example: State Chart XML)



**(XHTML + AJAX sub form) \* N**

**Multi-page XFDL  
Form**



# Use Case: SCXML as SIP Controller

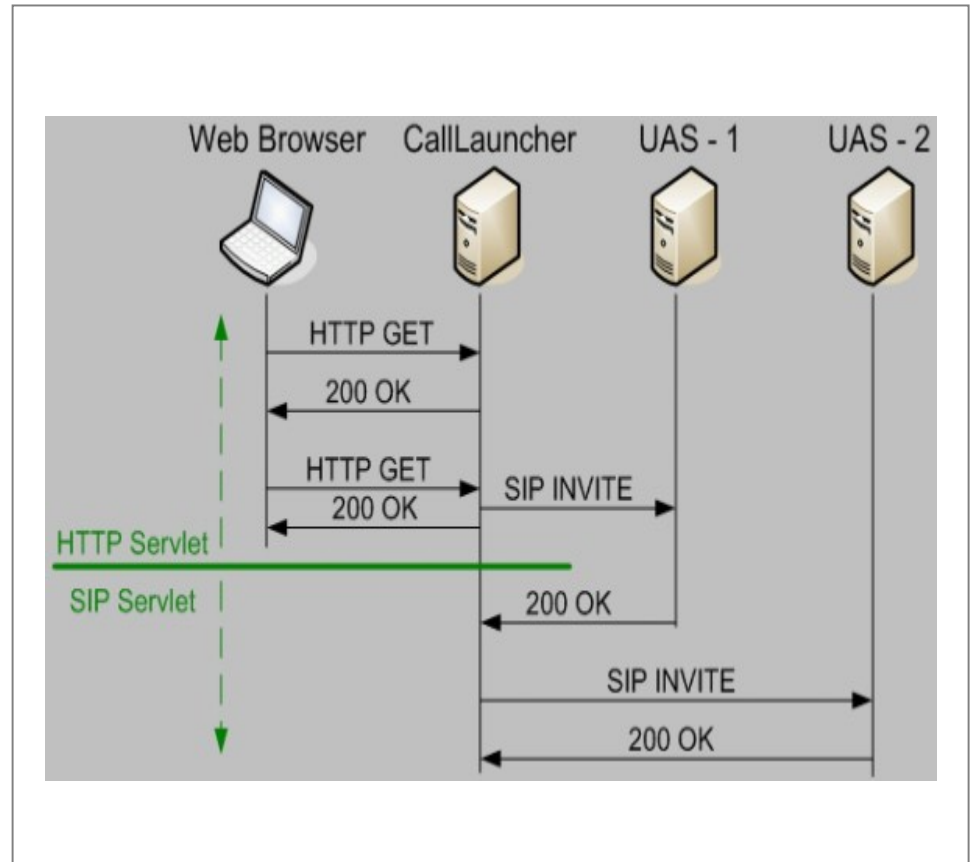
## Problems with SIP Servlets

- Complex to author – low level Java
- Hard to express common patterns for event-driven, or parallel paths
- Difficult to scale, reuse, compose

## Solution using SCXML

- Inherently event-driven semantics
- Direct support for parallelism, hierarchical composition
- Natural replacement for Java-based SIPlet implementations

Example Call/Invite SIP event flow



## Use Case: SCXML in the J2EE Web Container as JSF Controller

- **Problems with current JSF controllers**

- Ad-hoc state-machine like language
- Difficult to scale, reuse, and compose

- **Solution using SCXML**

- Shale “dialog manager” for cross-JSF page navigation in Apache
- Apache Commons SCXML engine in Shale runtime environment
- Invokes JSF pages or actions

Wizard Page 2 - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

IBM Business Transfo... IBM Internal Help Ho...

### Wizard Page 2

Name:

Address 1:

Address 2:

City:

State:

Zip Code:  [Pop Up](#)

### Test Description

This is one page of a wizard dialog. Use the navigation buttons provided to cycle between the various pages, or cause the dialog to be completed (finished or cancelled).

To enable triggering validation errors that redisplay the current page, all fields on this page except Address 2 are required.

[Back](#) to main menu.

Done

# Outline

- Introduction to State Chart XML (SCXML)
- Sample use cases for SCXML
- **SCXML Extensibility**
  - Data model
  - Custom actions
  - External communications module
- Specification Status
- Available implementations and tools

## Data model extensibility – XML data, XPath

```
<datamodel>
  <data id="cities">
    <list xmlns="">
      <city id="nyc" count="0">New York</city>
      <city id="bos" count="0">Boston</city>
    </list>
  </data>
</datamodel>

<assign
  location="$cities/list/city[@id='nyc']/@count"
  expr="1"/>
```

# Data model extensibility – JSON data, ECMAScript

```
<datamodel>
  <data id="cities">
    { "list" : [
      {id : "nyc", count : 0, city : "New York"},
      {id : "bos", count : 0, city : "Boston"}
    ] }
  </data>
</datamodel>
```

```
<assign
  location="cities.list[1].count"
  expr="1"/>
```



# Outline

- Introduction to State Chart XML (SCXML)
- Sample use cases for SCXML
- SCXML Extensibility
  - Data model
  - **Custom actions**
  - External communications module
- Specification Status
- Available implementations and tools

# Custom actions

- `<onentry>`, `<onexit>`, `<transition>`
  - open content model
  - Implementation or application specific namespaced actions

```
<scxml:onentry>  
  <my:email to="$employee/email"  
            cc="$approver/email"  
            subject="$message/title"  
            content="$message/content"/>  
</scxml:onentry>
```



# Outline

- Introduction to State Chart XML (SCXML)
- Sample use cases for SCXML
- SCXML Extensibility
  - Data model
  - Custom actions
  - **External communications module**
- Specification Status
- Available implementations and tools

# External communications module - send

- Send target types

```
<scxml:send type="scxml" target="..." />
```

```
<scxml:send type="x-foo" target="..." />
```

- I/O processors

- SCXML

- HTTP

- DOM



# VoiceXML 3.0 Transition Controllers, e.g. SCXML

```
<v3:form>  
  <scxml:scxml initial="...">  
    <!--  
      Controller  
    -->  
  </scxml:scxml>  
</v3:form>
```

DOM events as glue:

- Initiate execution of a VoiceXML field

```
<scxml:send target="#vfield" type="DOM" event="DOMActivate"/>
```

- Notification on filling fields / slots

```
<scxml:transition event="filled.vfield" cond="vfield == 'foo'"  
  target="statefoo" />
```



# External communications module - invoke

## ■ Invoke source types

```
<invoke src="dialog.vxml#welcome" type="vxm13">  
  <param name="skinpath" expr="res.paths.skin"/>  
  <finalize>  
    <script>finalizeAskHit();</script>  
  </finalize>  
</invoke>
```

## ■ Data transformations

- param (outbound)
- finalize (inbound)



# Outline

- Introduction to State Chart XML (SCXML)
- Sample use cases for SCXML
- SCXML Extensibility
  - Data model
  - Custom actions
  - External communications module
- **Specification Status**
- Available implementations and tools

# State Chart XML (SCXML)



## State Chart XML (SCXML): State Machine Notation for Control Abstraction

W3C Working Draft *13 May 2010*

**This version:**

<http://www.w3.org/TR/2010/WD-scxml-20100513/>

**Latest version:**

<http://www.w3.org/TR/scxml/>

**Previous version:**

<http://www.w3.org/TR/2009/WD-scxml-20091029/>

**Editors:**

Jim Barnett, Genesys (Editor-in-Chief)  
Rahul Akolkar, IBM  
RJ Auburn, Voxeo  
Michael Bodell, Microsoft  
Daniel C. Burnett, Voxeo  
Jerry Carter, (until 2008, when at Nuance)  
Scott McGlashan, HP  
Torbjörn Lager, Invited Expert



# W3C Specification Status

- Currently Seventh Working Draft
- Last Call Working Draft expected soon
- Assertions and test suite being produced
- Multiple implementations exist



# Outline

- Introduction to State Chart XML (SCXML)
- Sample use cases for SCXML
- SCXML Extensibility
  - Data model
  - Custom actions
  - External communications module
- Specification Status
- Available implementations and tools



# Apache Commons SCXML (Java runtime)

**Apache Commons**  
<http://commons.apache.org/>

Last Published: 23 November 2009 | Version: 0.10-SNAPSHOT

**Commons SCXML**

[State Chart XML \(SCXML\)](#) is currently a Working Draft published by the World Wide Web Consortium (W3C). SCXML provides a generic state-machine based execution environment based on Harel State Tables. SCXML is a candidate for the control language within multiple markup languages coming out of the W3C (see Working Draft for details). *Commons SCXML* is an implementation aimed at creating and maintaining a Java SCXML engine capable of executing a state machine defined using a SCXML document, while abstracting out the environment interfaces.

**Model**

**Middleware**

**Runtime**


**Commons SCXML**

```
<?xml xmlns="http://www.w3.org/2005/07/XMLSchema" version="1.0" xmlns:state="http://www.w3.org/2005/07/SCXML" >
  <state id="test">
    <state id="test">
      <transition event="test.done">
        <target state="twenty">
          </transition>
        </state>
      <!-- Follow up with a composite state -->
      <state id="twenty">
        <transition>
          <target state="twenty_one">
            </transition>
          </state>
        <state id="twenty_one">
          <transition event="twenty_one.done">
            <target state="twenty_two">
              </transition>
            </state>
          <state id="twenty_two">
            <transition event="twenty_two.done">
              <target state="twenty_three">
                </transition>
              </state>
            </state>
          </state>
        </state>
      </state>
    </state>
  </state>
</?xml>
```

The use cases for an SCXML engine are multiple and varied. Anything that can be represented as a UML state chart -- business process flows, view navigation bits, interaction or dialog management, and many more -- can leverage an SCXML engine library.

# Apache Commons SCXML-JS (JavaScript compiler)


Commons SCXML JS - Overview



## Apache Commons

<http://commons.apache.org/>

Last Published: 26 June 2010 | Version: 1.0-SNAPSHOT [ApacheCon](#) | [Apache](#) | [Commons](#) | [Sandbox](#)



1-5 November  
Atlanta, GA

### Commons SCXML JS

- Overview
- Wiki
- Development
  - History
  - Mailing Lists
  - Issue Tracking
  - Team
  - Source Repository
- Project Documentation
  - Project Information
    - About
    - Continuous Integration
    - Dependencies

## Commons SCXML - JavaScript

The Commons SCXML JavaScript project has two goals. The first is to develop an SCXML-to-JavaScript compiler optimized for User Interface development on the World Wide Web, to allow developers to elegantly describe and implement Web-based UIs with complex behavioural requirements. The second goal is to generate graphical depictions of statecharts, which may then be animated in response to live UI events, to allow developers to better comprehend the dynamic behaviour described by their statecharts.

### Status

- This code is in the Commons *Sandbox*
- The code is unreleased
- Methods and classes can and will appear and disappear without warning
- If you like the code and want to push it towards a release, join the [mailing list](#)!

## Releases

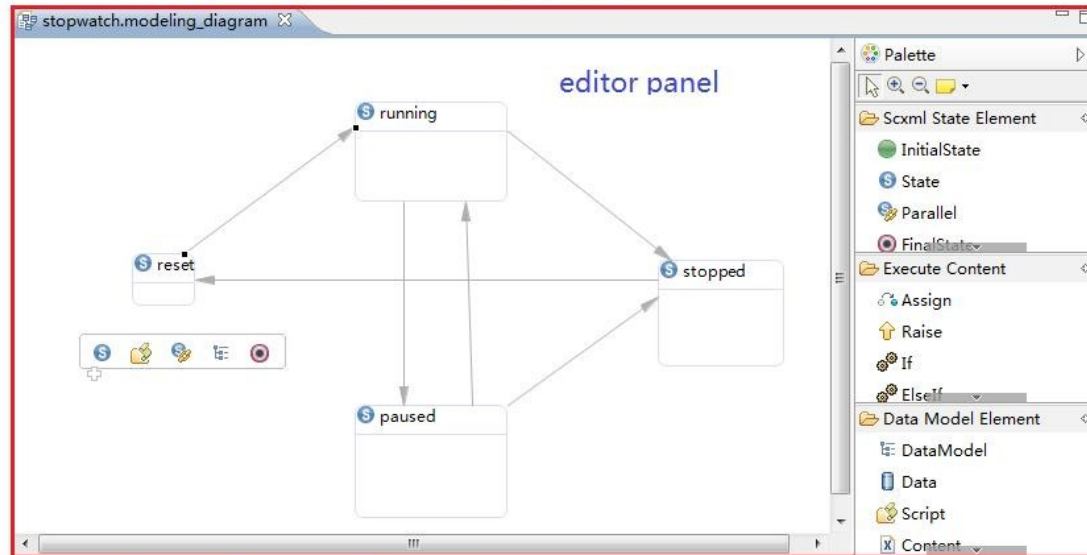
None. This is a [Sandbox](#) component.

# Apache Commons SCXML-Eclipse (Visual Editor)

Visual SCXML - Visual SCXML - How to...

## Edit SCXML diagram content

You can edit SCXML diagram in this editor, use tools in palette tool bar, create states, execute content and data model and other elements in W3C's SCXML recommendation specification:



Problems | Javadoc | Declaration | Properties

### Transition watch.start

property view

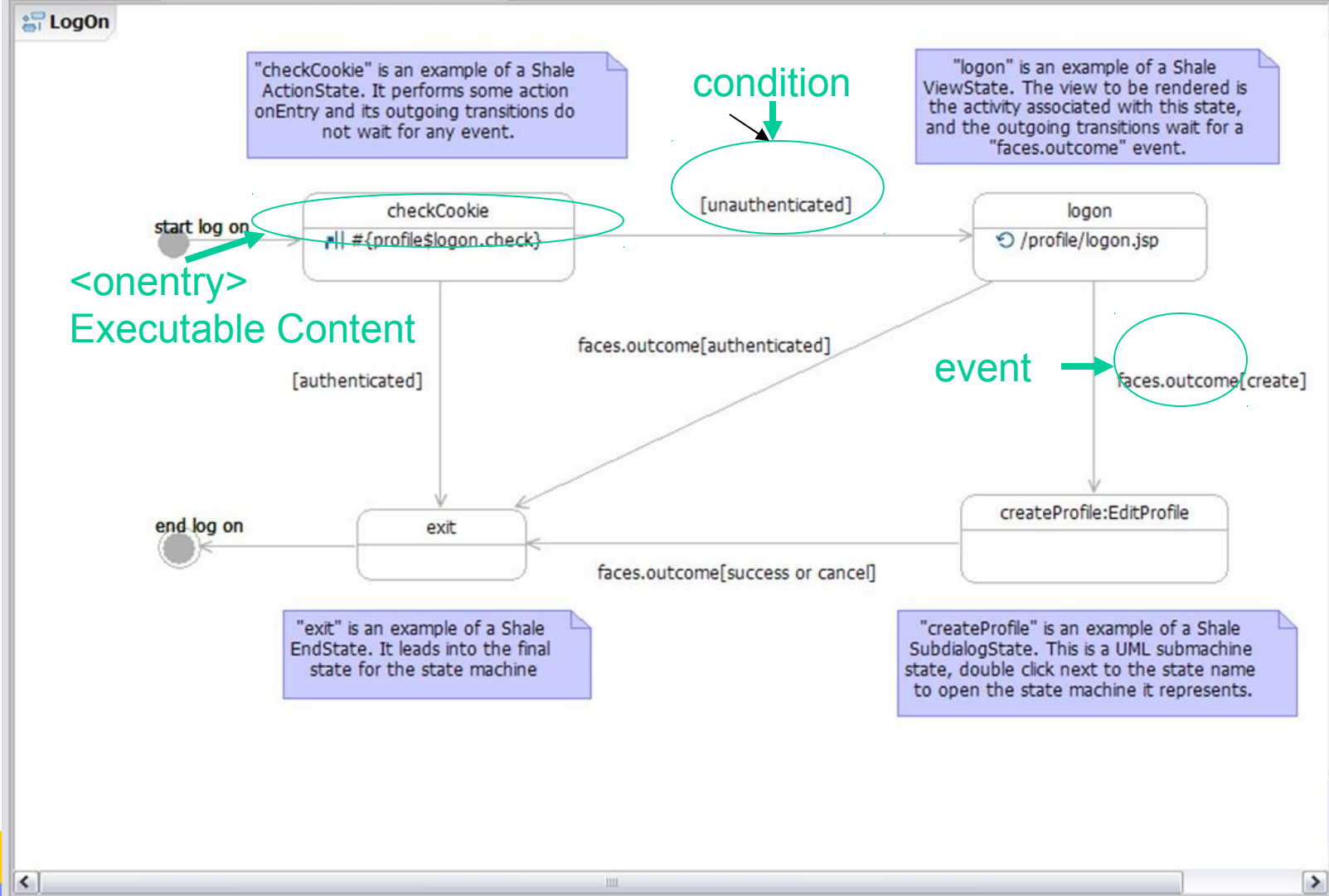
Core	Property	Value
Appearance	Anchor	
	Cond	
	Event	watch.start
	Source Status	
	Target	
	Target Status	State running

And as you see in the diagram above, you can edit elements' properties in Eclipse property view.

## Export a SCXML document

Right click the SCXML "modeling" file which you would like to export, select "Export SCXML document", and fill a filename, click "Finish" to finish export operation.





**Palette**

- Select
- Note
- UML Common
- State Machine ...
- State
  - Initial State
  - Final State
- State types
- Pseudostate types
- Transition

Geometric Shapes



# Synergy SCXML Web Laboratory

SCXML Source

```
<?xml version="1.0"?>
<scxml initialstate="s0">

  <state id="s0">
    <onentry>
      <log expr="'Hello world!'" />
    </onentry>
    <transition target="s" />
  </state>

  <final id="s" />

</scxml>
```

More Examples... About

## SCXML Web Laboratory

SCXML Web Laboratory is a web interface to a prototype implementation of State Chart XML (SCXML). For an immediate demonstration of what the program does, click the Run button in the frame to the left. Then perhaps try some of the other examples that appear in this frame when you click the More Examples button on the top. Note that you may actually edit the SCXML sources and/or the Input Events field before you click Run. This is the main purpose of SCXML Web Laboratory: To allow people to get a feeling for what SCXML is all about, and to elicit feedback in order that we can improve our implementation!

Never heard about SCXML? Unfortunately, this is not the place to introduce SCXML as such. Instead, we refer the interested visitor to the documents listed in [the bibliography](#).

## Synergy SCXML v. 0.7.5

Synergy is the name of a dialogue system research platform that we are developing in a project of ours, and Synergy SCXML is an explorative implementation of SCXML in [the Oz programming language](#) - the language that we tend to use a lot in our projects.

We hope that our implementation of SCXML will provide us with flexible means to explore the [Data-Flow-Presentation \(DFP\) framework](#) recently proposed by the W3C. The choice of Oz as implementation language means that we can keep the codebase conveniently small (version 0.7.5 consists of less than 1,500 lines of code half of which is reusable library code). A small codebase together with what Oz has to offer in the form of dataflow concurrency, logic programming, constraint-programming, network transparent distribution. etc., means we will be able to explore advanced concepts very quickly.

Also, since we use Oz as our datamodel access/scripting language our SCXML developers will have at their disposal something far more powerful than Javascript or anything like it.

powered by

Run Timelimit: 5 s. Report:  Trace  Log  Events

Input Events

<http://www.ling.gu.se/~lager/Labs/SCXML-Lab/>

Thanks

