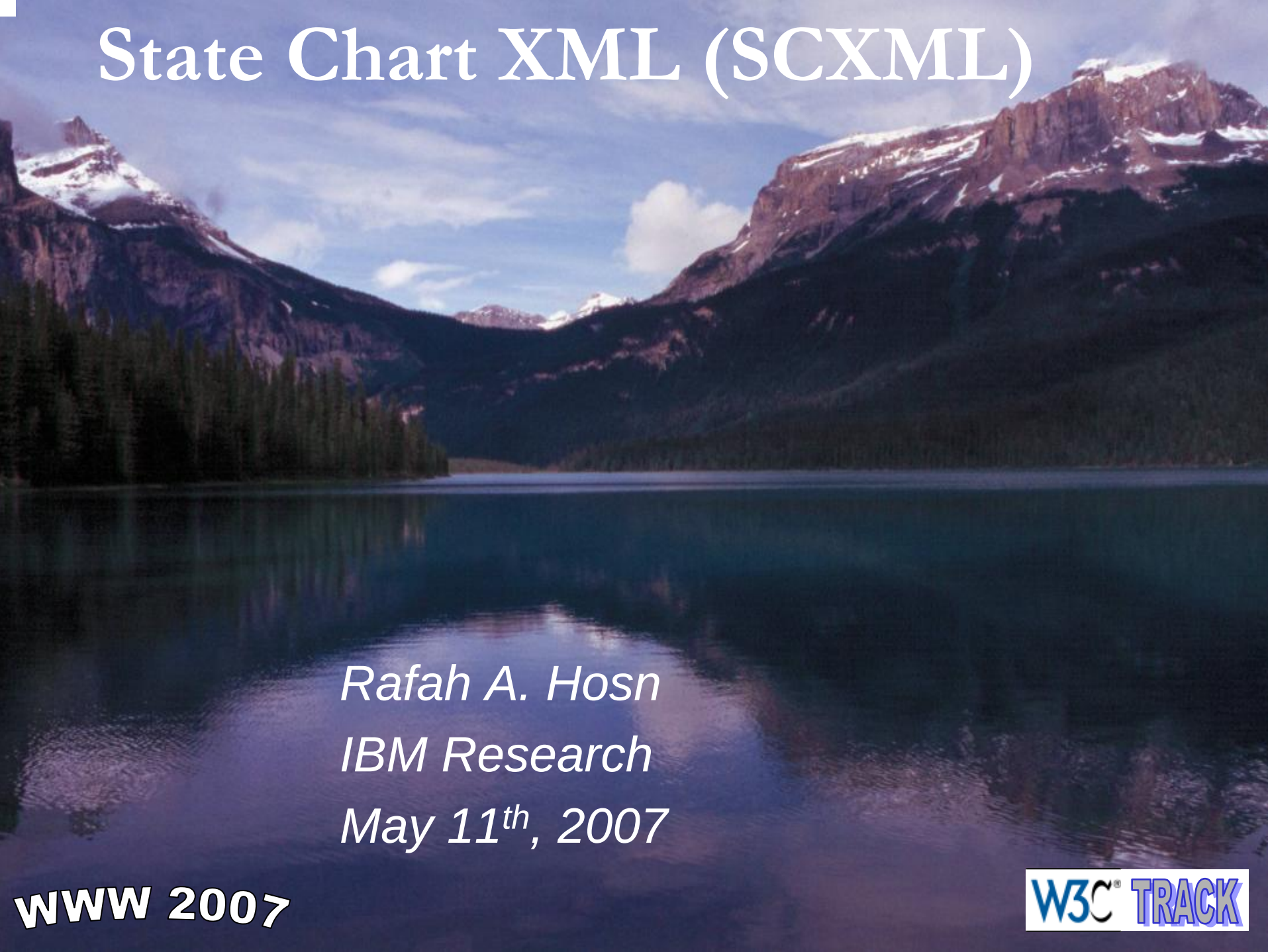
State Chart XML (SCXML)



Rafah A. Hosn

IBM Research

May 11th, 2007

WWW 2007

W3C TRACK

Outline

- SCXML, Language Overview
- SCXML Status & Open Source Implementations
- SCXML Usage Scenarios & Demos

Agenda

- **SCXML, Language Overview**
- SCXML Status & Open Source Implementations
- SCXML Usage Scenarios & Demos

SCXML

- Markup language for state machine definition
  - VBWG
- 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

States and Transitions

- States represent status of system
  - Check Status, Log On, Create a new Profile
- Events are what happens
  - Authentication Done, Profile Created
- Transitions move between states
  - Triggered by events and conditions
  - Checking status to logging on

Hierarchal States

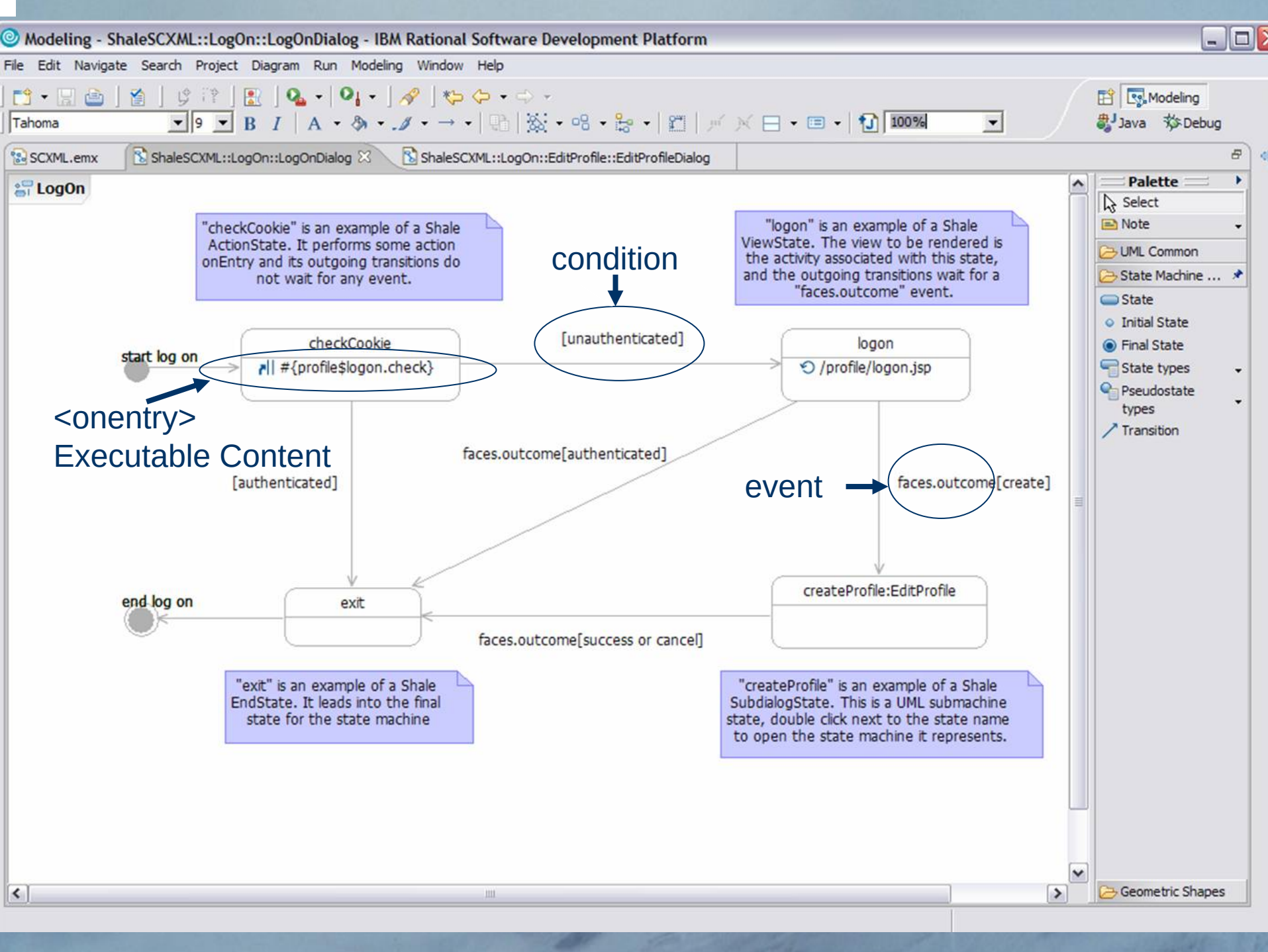
- **Compound state**
  - Parent state decomposed into child states
  - <initial> default initial state
- **Represent task decomposition**
  - Parent state always in a single child state
  - Transitions between child states show progress of the task
- **Allow re-use of sub-machines**

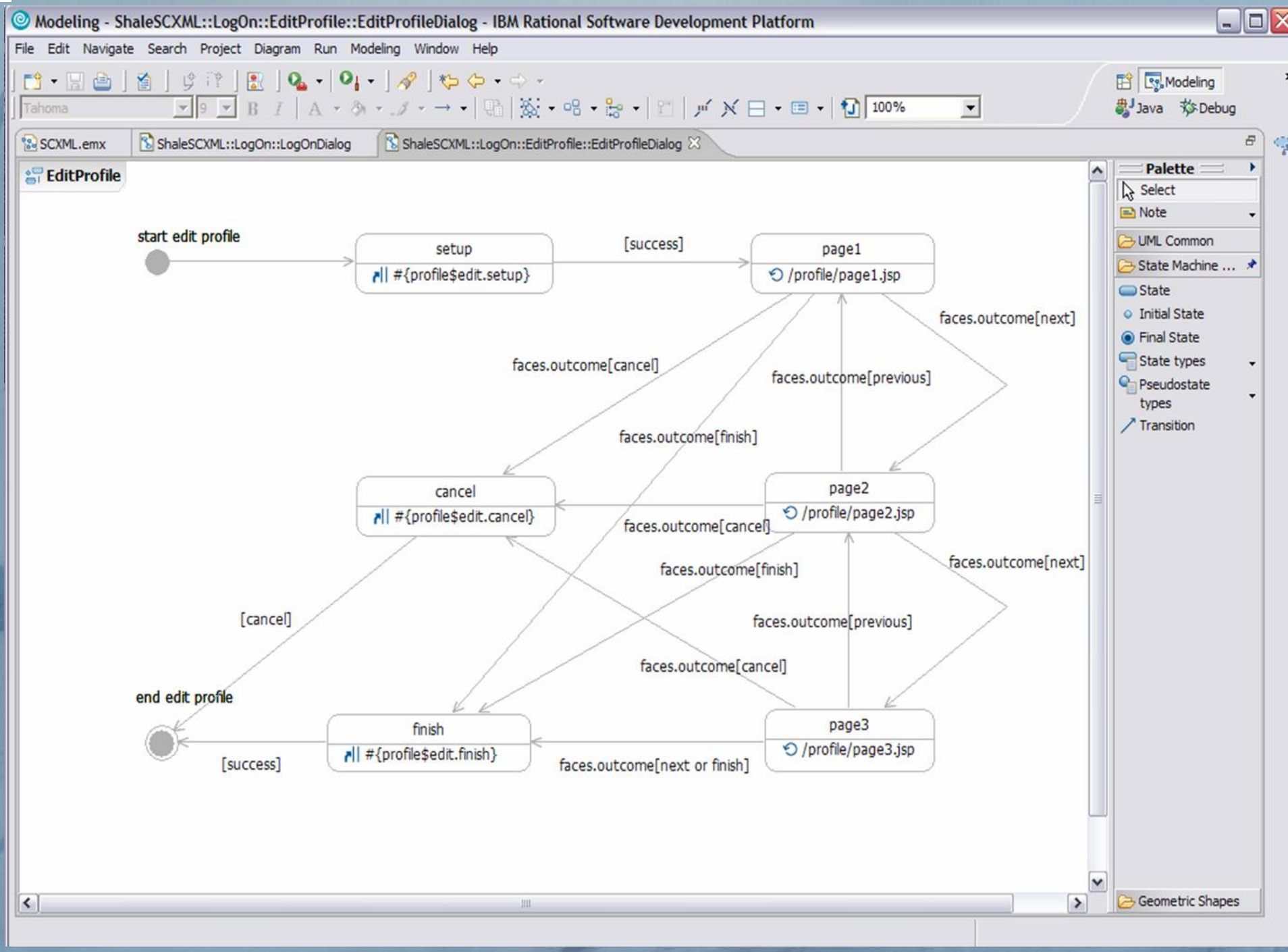
Data Model*

- XML data model
  - Visible to all states
  - Local to SCXML interpreter
- <datamodel>
  - element with one or more <data> elements
- <data>
  - element whose value is an XML tree
  - XML tree can be sourced from any URI or in-lined

Executable Content

- **Actions**
  - In <onentry>, <onexit> and <transition>
- **Basic operations**
  - <send> asynchronous external event to arbitrary URL
  - <event> asynchronous event internal to the state machine
  - <assign> updates data model
  - <validate> validates data model
- **Extensions**
  - By profile or platform-specific


Modeling - ShaleSCXML::LogOn::LogOnDialog - IBM Rational Software Development Platform
File Edit Navigate Search Project Diagram Run Modeling Window Help
SCXML.emx
ShaleSCXML::LogOn::LogOnDialog
ShaleSCXML::LogOn::EditProfile::EditProfileDialog
LogOn
"checkCookie" is an example of a Shale ActionState. It performs some action onEntry and its outgoing transitions do not wait for any event.
condition
"logon" is an example of a Shale ViewState. The view to be rendered is the activity associated with this state, and the outgoing transitions wait for a "faces.outcome" event.
start log on
checkCookie
#{profile$logon.check}
[unauthenticated]
logon
/profile/logon.jsp
<onentry>
Executable Content
[authenticated]
faces.outcome[authenticated]
event
faces.outcome[create]
end log on
exit
createProfile:EditProfile
faces.outcome[success or cancel]
"exit" is an example of a Shale EndState. It leads into the final state for the state machine
"createProfile" is an example of a Shale SubdialogState. This is a UML submachine state, double click next to the state name to open the state machine it represents.
Palette
Select
Note
UML Common
State Machine ...
State
Initial State
Final State
State types
Pseudostate types
Transition
Geometric Shapes


Modeling - ShaleSCXML::LogOn::EditProfile::EditProfileDialog - IBM Rational Software Development Platform
File Edit Navigate Search Project Diagram Run Modeling Window Help
Modeling
Java
Debug
SCXML.emx
ShaleSCXML::LogOn::LogOnDialog
ShaleSCXML::LogOn::EditProfile::EditProfileDialog
EditProfile
Palette
Select
Note
UML Common
State Machine ...
State
Initial State
Final State
State types
Pseudostate types
Transition
Geometric Shapes
start edit profile
setup
#{profile$edit.setup}
[success]
page1
/profile/page1.jsp
faces.outcome[next]
faces.outcome[cancel]
faces.outcome[previous]
faces.outcome[finish]
cancel
#{profile$edit.cancel}
page2
/profile/page2.jsp
faces.outcome[cancel]
faces.outcome[finish]
faces.outcome[next]
[cancel]
faces.outcome[previous]
faces.outcome[cancel]
end edit profile
finish
#{profile$edit.finish}
page3
/profile/page3.jsp
[success]
faces.outcome[next or finish]

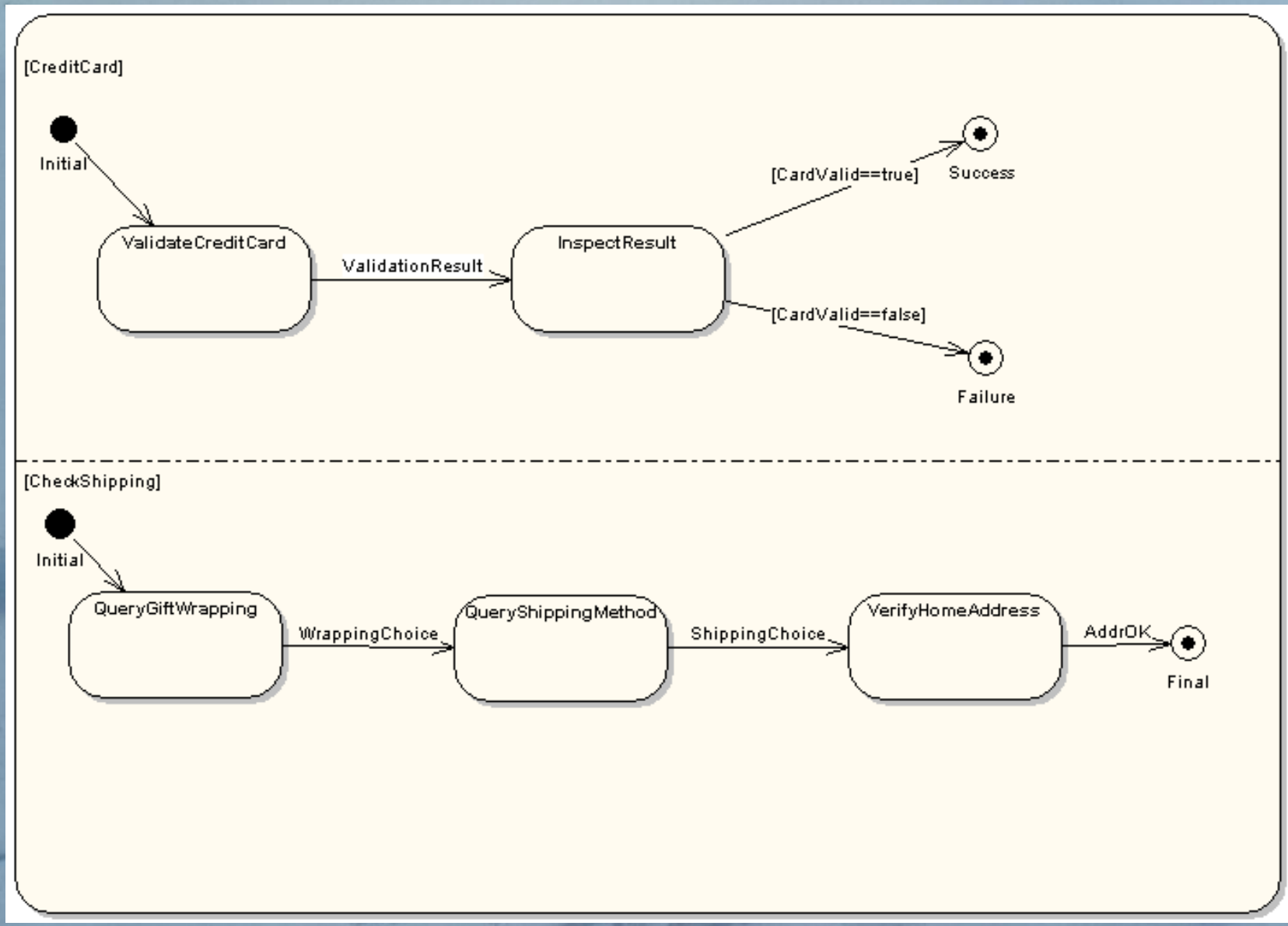
SCXML Equivalent

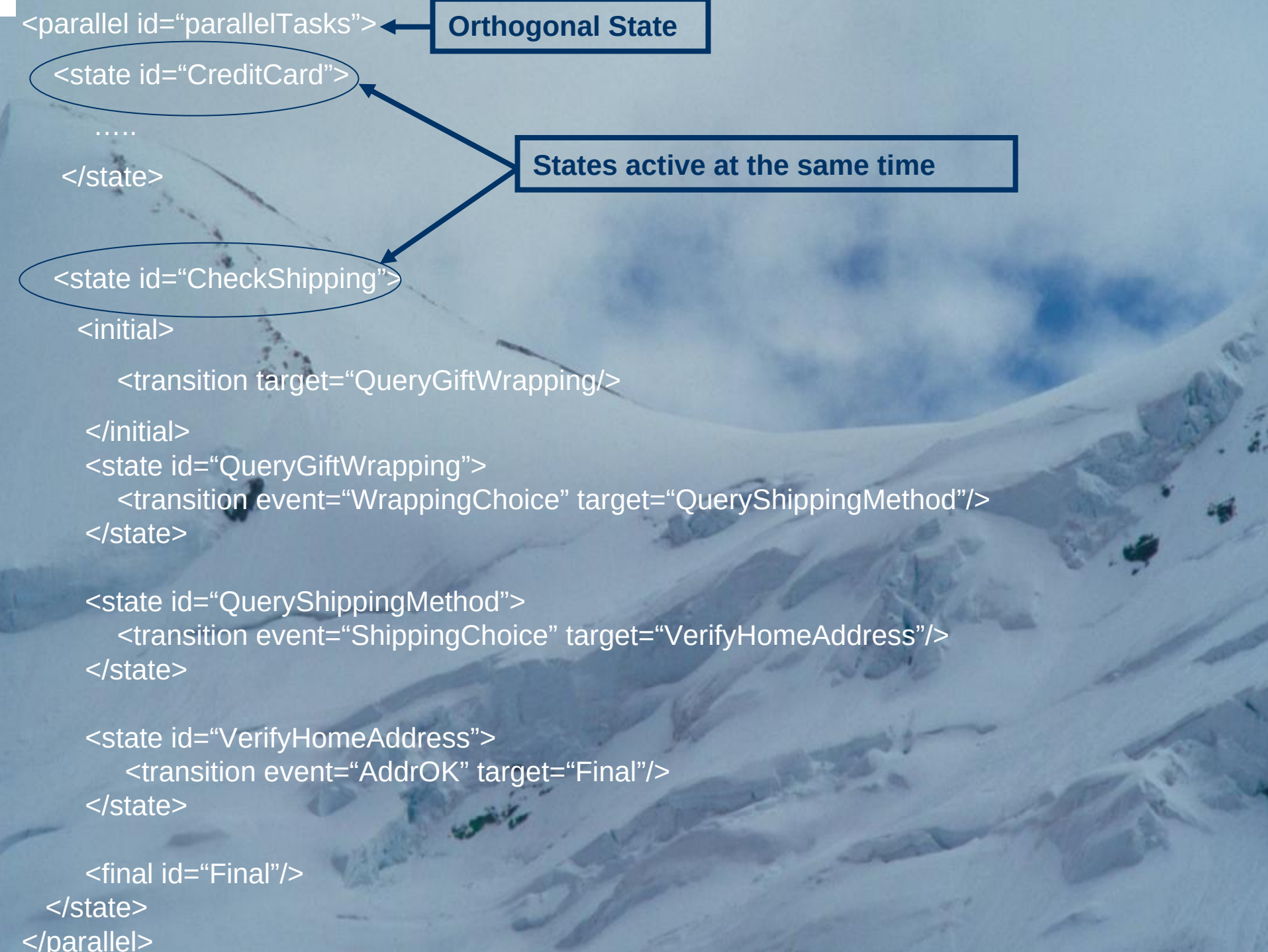
```
<scxml xmlns="http://www.w3.org/2005/07/SCXML" version="1.0“
    initialstate="checkCookie">
 <state  id="checkCookie">
     <onentry>
        <var   name="cookieOutcome"   expr="#{profile$logon.check}" />
     </onentry>
     <transition   cond="${cookieOutcome eq 'authenticated'}“  target="exit"/>
    <transition    cond="${cookieOutcome eq 'unauthenticated‘}” target="logon"/>
 </state>
 <state   id="logon">
   <transition      event="faces.outcome"   cond="${outcome eq 'authenticated'}“
                    target="exit"/>
   <transition      event="faces.outcome"    cond="${outcome eq 'create'}”
                    target="createProfile"/>
 </state>
 <state    id="createProfile"   src="edit-profile-config.xml" >
   <transition    event="createProfile.done"
                  cond="${outcome eq 'success' or outcome eq 'cancel'}” target="exit“/>
 </state>
 <final  id="exit"/>
</scxml>
```

Compound State

Orthogonal States

- **Parallel state**
  - A state that has multiple active children
  - Parallel children operate independently
- **Parallel represents fork/join logic**
- **Parallel state finishes**
  - When each sub-state reaches a final state
  - When a transition is taken out of the parallel


[CreditCard]
Initial
ValidateCreditCard
ValidationResult
InspectResult
[CardValid==true]
Success
[CardValid==false]
Failure
[CheckShipping]
Initial
QueryGiftWrapping
WrappingChoice
QueryShippingMethod
ShippingChoice
VerifyHomeAddress
AddrOK
Final


<parallel id="parallelTasks">
Orthogonal State
<state id="CreditCard">
.....
</state>
States active at the same time
<state id="CheckShipping">
<initial>
<transition target="QueryGiftWrapping/>
</initial>
<state id="QueryGiftWrapping">
<transition event="WrappingChoice" target="QueryShippingMethod"/>
</state>
<state id="QueryShippingMethod">
<transition event="ShippingChoice" target="VerifyHomeAddress"/>
</state>
<state id="VerifyHomeAddress">
<transition event="AddrOK" target="Final"/>
</state>
<final id="Final"/>
</state>
</parallel>

<history> and <anchor>

- <history> pseudo-state
  - Allows re-entry into a compound state at point it was last exited
  - Represents pause and resume logic
  - Transition must specify ID of the <history> state
- <anchor> tag is more dynamic
  - <transition anchor="foo"/> goes to last state to define "foo"
  - Do not need to know the name of the state
  - Can rollback data model as well
  - Not in Harel

<invoke> and <finalize>

- <invoke> calls external service
- <finalize> preprocesses the results
  - Used to update the data model
- <state> with <invoke> represents activity of the external service
- Not in Harel

Agenda

- SCXML, Language Overview
- SCXML Status & Open Source Implementations
- SCXML Usage Scenarios & Demos

SCXML Status & Open Source Implementations

- **Public Draft available:**
  - **http://www.w3.org/TR/scxml/**
- **Open Source Apache SCXML engine**
  - **Available: http://jakarta.apache.org/commons/scxml/**
  - **Shale "Dialog Manager": http://shale.apache.org/shale-dialog-scxml/**
- **SCXML Mozilla Plugin & RSA Modeling Transfomer available:**
  - **http://www.alphaworks.ibm.com/tech/scxml**

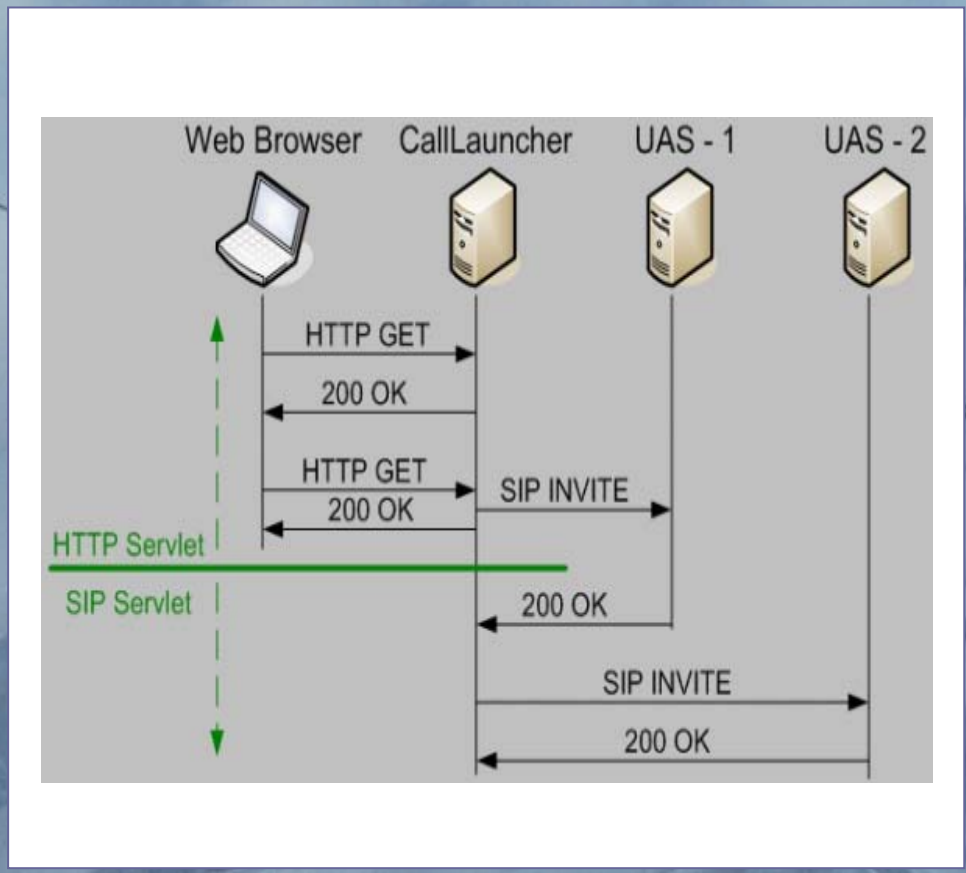
Agenda

- SCXML, Language Overview
- SCXML Status & Open Source Implementations
- **SCXML Usage Scenarios & Demos**

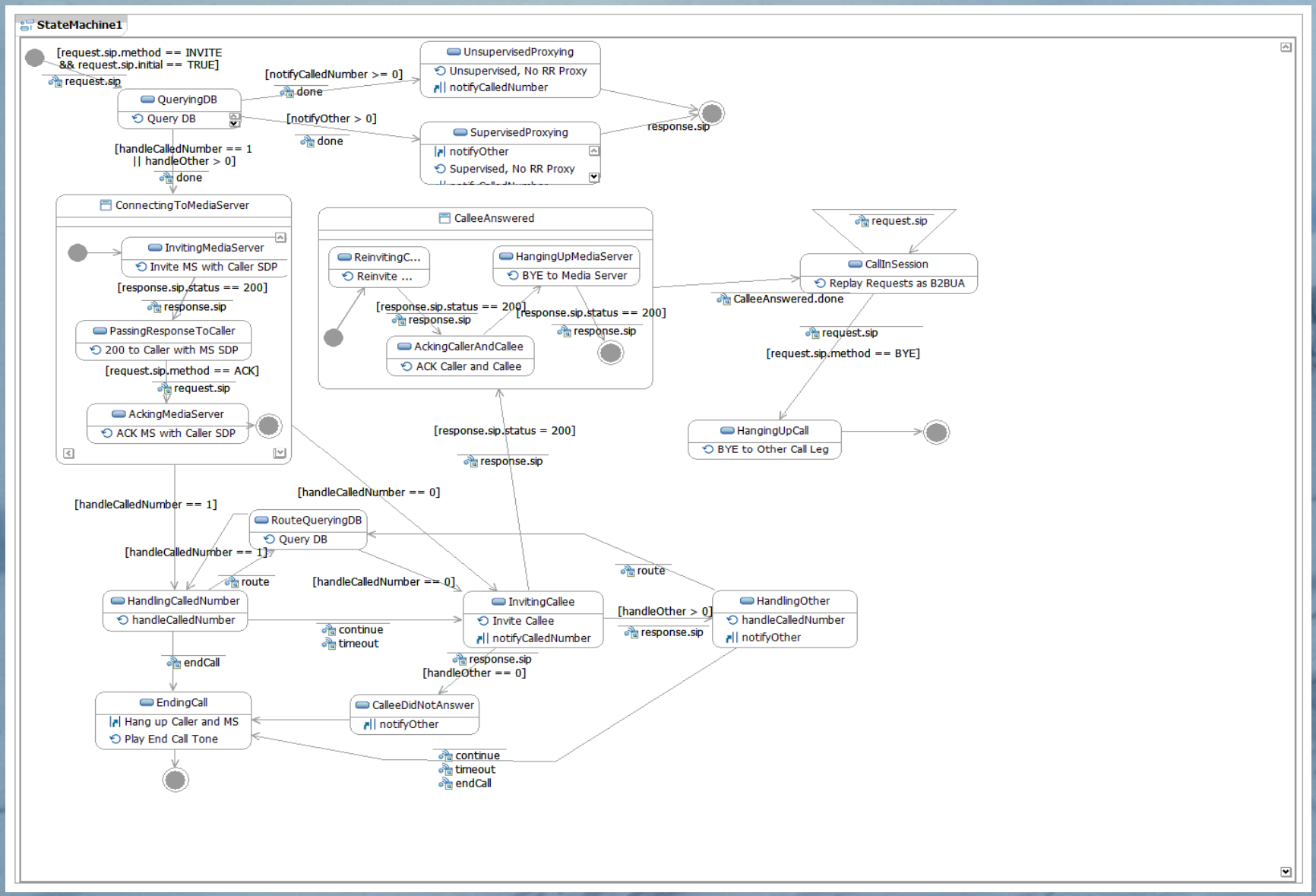
Use Case: SCXML as SIP Controller

- **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

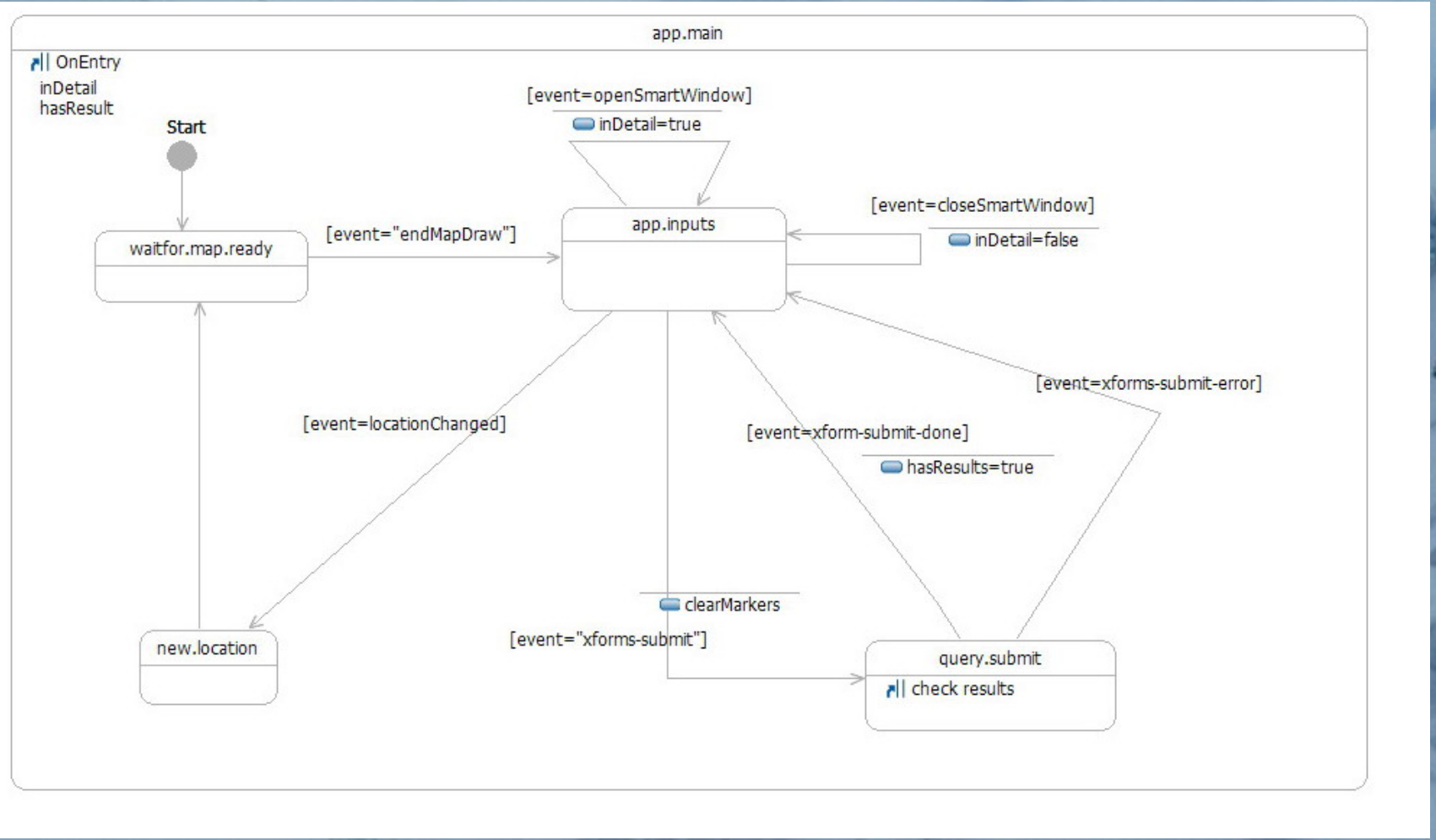
Example Call/Invite SIP event flow



SCXML SIP Controller for Parlay Call Direction

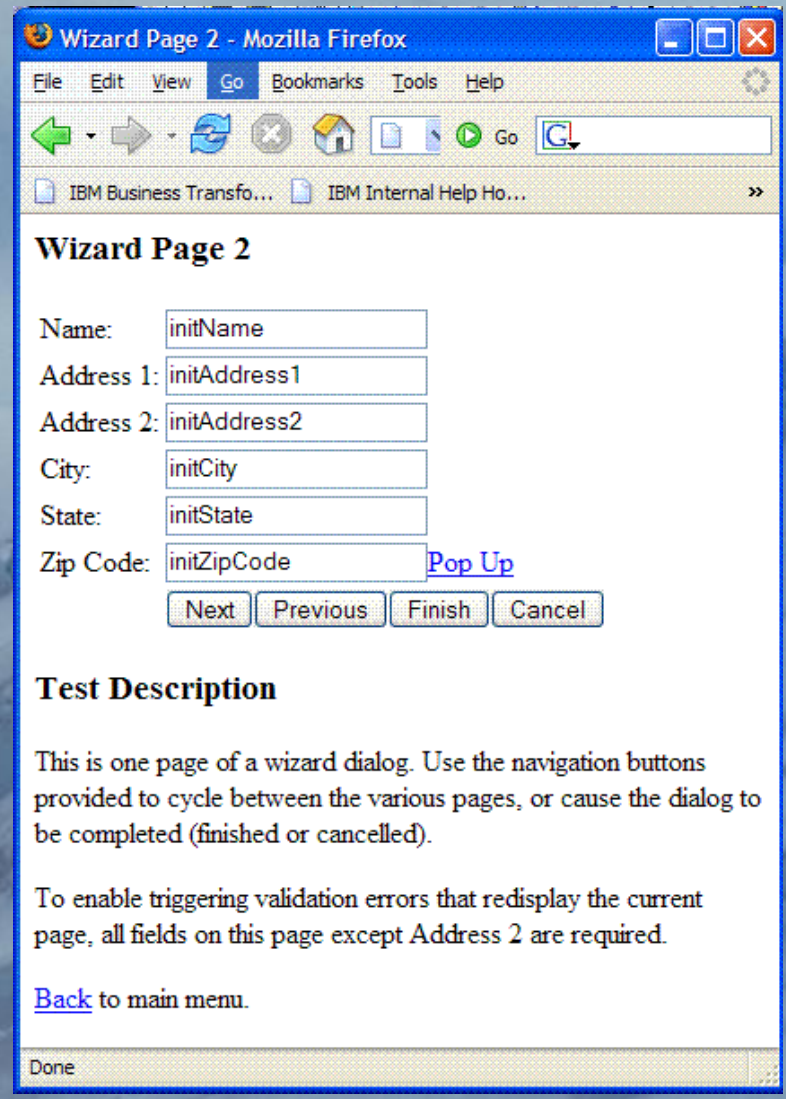


Use Case: SCXML Mash-up controller in Mozilla



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




Thank You !
Pictures courtesy of: www.math.unc.edu/Faculty/petersen/Pics/Banff03/
WWW 2007
W3C TRACK