

Orchestration and Choreography: Standards, Tools and Technologies for Distributed Workflows

5-7 October, 2005, Second University of Naples, Naples, Italy

Steve Ross-Talbot

CEO Pi4 Technologies

Chair W3C Web Services Activity

Co-chair W3C Web Services Choreography

Agenda

- What are Workflow, Orchestration and Choreography?
- What is SOA?
- How do all of these fit together?
- WS-CDL and bioinformatics
- WS-CDL under the hood
- Formalisms and what they mean
- WS-CDL in practice (lets look at tools)
- What can I do with it
- Summary

What is a workflow?

The automation of business processes, in whole or in part, during which documents, information or tasks are passed from one participant to another for action according to a set of procedural rules

WFMC

What is orchestration?

orchestration [of web service] is a technique to recursively compose and orchestrate web services to provide a new composite web service

WS-BPEL

What is choreography?

A choreography is a global behavioral contract that describes (and therefore can be used to constrain) the valid ordering of messages between services that make up some flow that meets some [business] objective

WS-CDL

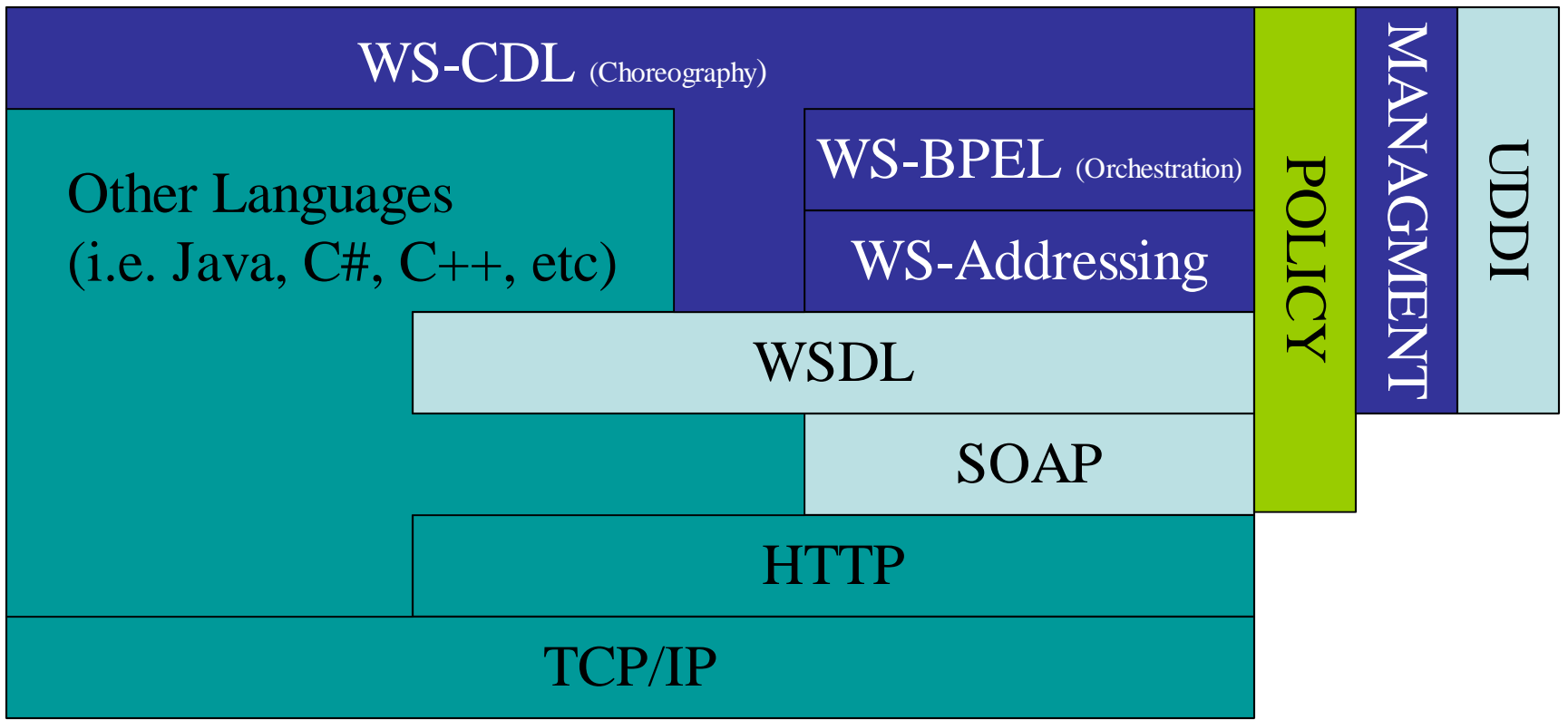
What is SOA?

- Service Oriented Architecture is a way of building distributed systems.
 - A service is a computational process with a WSDL interface.
 - A service may interact with other services through any communication mechanism (HTTP is but one).
 - Services are discovered at runtime.
 - SOA encourages document centric rather/messaging passing than function-centric programming (see WSDL2.0)

How does it fit together?

- An SOA is just a grid of services. Something has to give the grid and the services meaning.
- Meaning can be thought of as some notion of the common behavior of the services in achieving some goal. We might call this a workflow description.
- Distributed (peer to peer) workflows may be described using WS-CDL and executed at the peers using WS-BPEL. Thus orchestration is an execution idiom and choreography a description one.

How does it fit together?

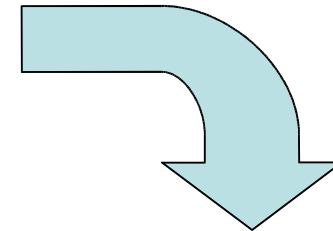


| |
|-----------|
| Legacy |
| Available |
| Nascent |
| Missing |

Relevance to Bioinformatics

- Formalism

| Agents | Communication Events |
|-----------------|---|
| Small molecules | Electron Sharing |
| Proteins | Protein-protein interaction, binding, phosphorylation, methylation, etc |
| Cells | Material consumption, environmental sensing, etc |



| Agents | Communication Events |
|---------------------|----------------------|
| Network service | TCP/IP read/write |
| Email client/server | SMTP read/write |
| Browser server | HTTP read/write |
| WS-CDL service | send/receive |

Common formalisms 

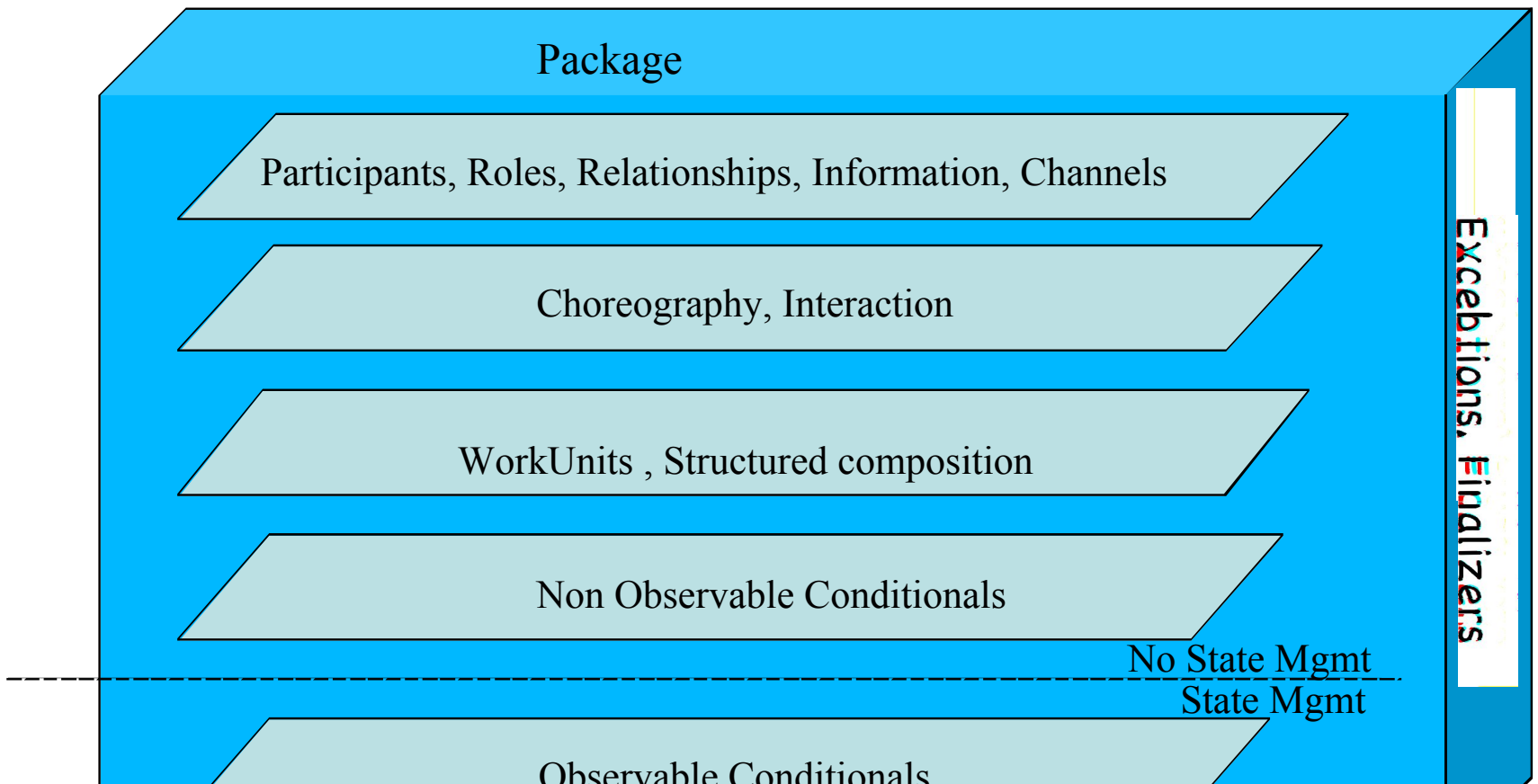
Relevance to Bioinformatics

- SOA
 - Emergence of WSDL described services (KEGG, BLAST, FASTA, Swiss-Prot, ... etc)
 - Componentization for reuse and greater availability
 - Decrease in costs and complexity of data integration
 - WSDL, SOAP, Java, .NET, and the rest of the stack presented in the context of a service grid.

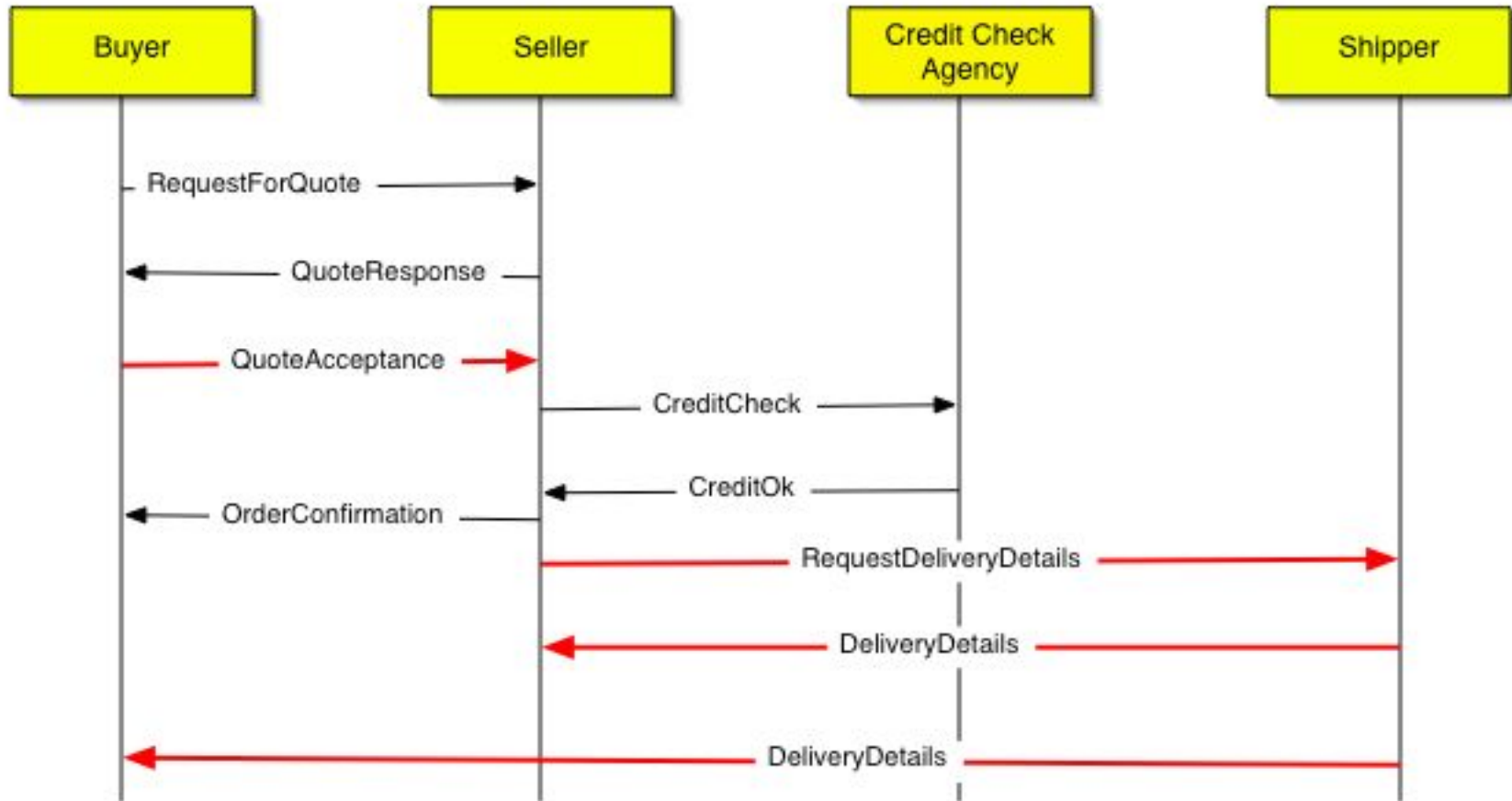
Relevance to Bioinformatics

- Process volatility
 - In-silico experimentation as workflows across a peer to peer SOA
 - Contractual descriptions for clinical trial protocols (Vioxx) that can be proven to have been followed
 - Outsourcing and yet remaining in control
 - WS-BPEL and WS-CDL

WS-CDL under the hood



WS-CDL under the hood



Normal Collaboration

WS-CDL under the hood

```

<interaction name="
    " operation="
    channelVariable="
">
  <description type="
    ">
    </description>
  <participateRelationshipType
    " fromRole="
    " toRole="
    " />
    <exchange name="
    " informationType"
    " action="
    "></exchange>
</interaction>

<interaction name="
    " operation="
    channelVariable="
">
  <description type="
    ">
    </description>
  <participateRelationshipType
    " fromRole="
    " toRole="
    " />
    <exchange name="
    " channelType="
    " action="
    ">
      <sendVariable="
    " />
      <receiveVariable="
    " />
    </exchange>
</interaction>

<interaction name="
    " operation="
    channelVariable="
">
  <description type="
    ">
    </description>
  <participateRelationshipType
    " fromRole="
    " toRole="
    " />
    <exchange name="
    " informationType"
    " action="
    "></exchange>
    <exchange name="
    " informationType"
    " action="
    "></exchange>
</interaction>

<interaction name="
    " operation="
    channelVariable="
">
  <description type="
    ">
    </description>
  <participateRelationshipType
    " fromRole="
    " toRole="
    " />
    <exchange name="
    " channelType="
    " action="
    ">
      <sendVariable="
    " />
      <receiveVariable="
    " />
    </exchange>
</interaction>

<interaction name="
    " operation="
    channelVariable="
">
  <description type="
    ">
    </description>
  <participateRelationshipType
    " fromRole="
    " toRole="
    " />
    <exchange name="
    " informationType"
    " action="
    "></exchange>
</interaction>
    
```

WS-CDL under the hood

```

<?xml version="1.0" encoding="UTF-8" ?>
<package name=" " author=" "
version=" " targetNamespace=" "
xmlns="http://www.w3.org/2004/12/ws-chor/cdl"
xmlns:bs="http://www.pi4tech.com/cdl/BuyerSellerExample-1">
<description type=" " >
</description>
<sequence>
<interaction name=" " operation=" " channelVariable=" " initialize=" " >
<description type=" " >
</description>
<participateRelationshipType fromRole=" " toRole=" " />
<exchange name=" " informationType=" " action=" " >
<description type=" " >
</description>
</exchange>
<exchange name=" " informationType=" " action=" " >
<description type=" " >
</description>
</exchange>
</interaction>
</sequence>
</choreography>
</package>
    
```

WS-CDL under the hood

```

<workunit name="                                " repeat="                                ">
  <choice>
    <silentAction roleType="                                ">
      <description type="                                ">                                </description>
    </silentAction>
    <sequence>
      <interaction name="                                " operation="                                " channelVariable="                                ">
        <description type="                                ">                                </description>
        <participateRelationshipType="                                " fromRole="                                " toRole="                                " />
        <exchange name="                                " informationType="                                " action="                                ">
          </exchange>
        </interaction>
        <interaction name="                                " operation="                                " channelVariable="                                ">
          <description type="                                ">                                </description>
          <participateRelationshipType="                                " fromRole="                                " toRole="                                " />
          <exchange name="                                " channelType="                                " action="                                ">
            <send variable="                                " />
            <receive variable="                                " />
          </exchange>
        </interaction>
        <assignRoleType="                                ">
          <copy name="                                ">
            <sourceExpression="                                " />
            <targetVariable="                                " />
          </copy>
        </assign>
      </sequence>
    </choice>
  </workunit>
  <sequence>
    <interaction name="                                " operation="                                " channelVariable="                                ">
      <description type="                                ">                                </description>
      <participateRelationshipType="                                " fromRole="                                " toRole="                                " />
      <exchange name="                                " informationType="                                " action="                                ">
        </exchange>
      </interaction>
      <exchange name="                                " informationType="                                " action="                                ">
        <description type="                                ">                                </description>
      </exchange>
    </sequence>
  </workunit>

```


WS-CDL under the hood

```

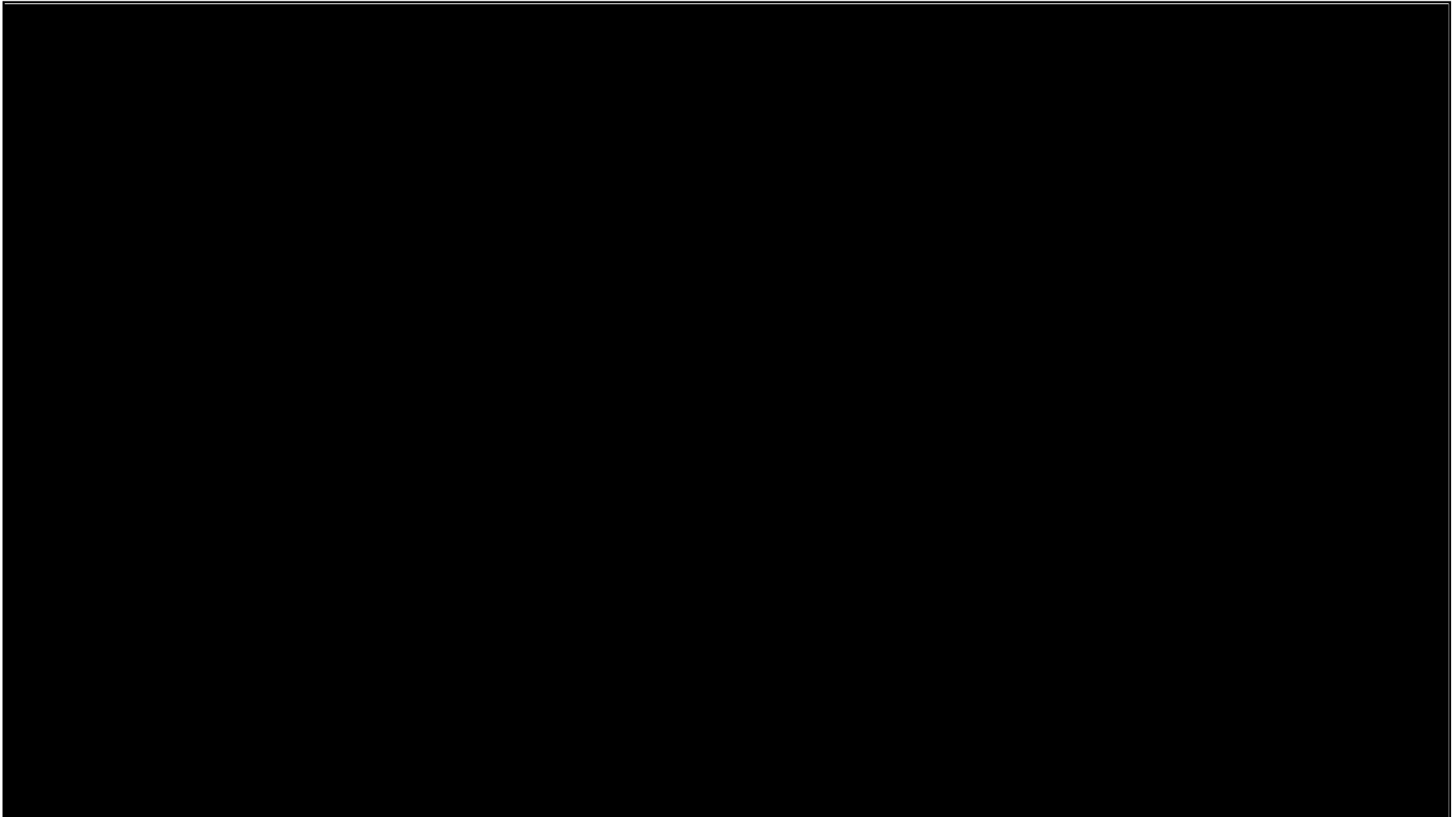
<parallel>
  <workunit name=" " operation=" " channelVariable=" ">
    <sequence>
      <interaction name=" " operation=" " channelVariable=" ">
        <description type=" " />
        <participation relationshipType=" " fromRole=" " toRole=" " />
        <exchange name=" " informationType=" " action=" "></exchange>
      </interaction>

      <choice>
        <sequence>
          <interaction name=" " operation=" " channelVariable=" ">
            <description type=" " />
            <participation relationshipType=" " fromRole=" " toRole=" " />
            <exchange name=" " informationType=" " action=" "></exchange>
          </interaction>
          <assignment type=" " >
            <copy name=" " >
              <source expression=" " />
              <target variable=" " />
            </copy>
          </assignment>
        </sequence>

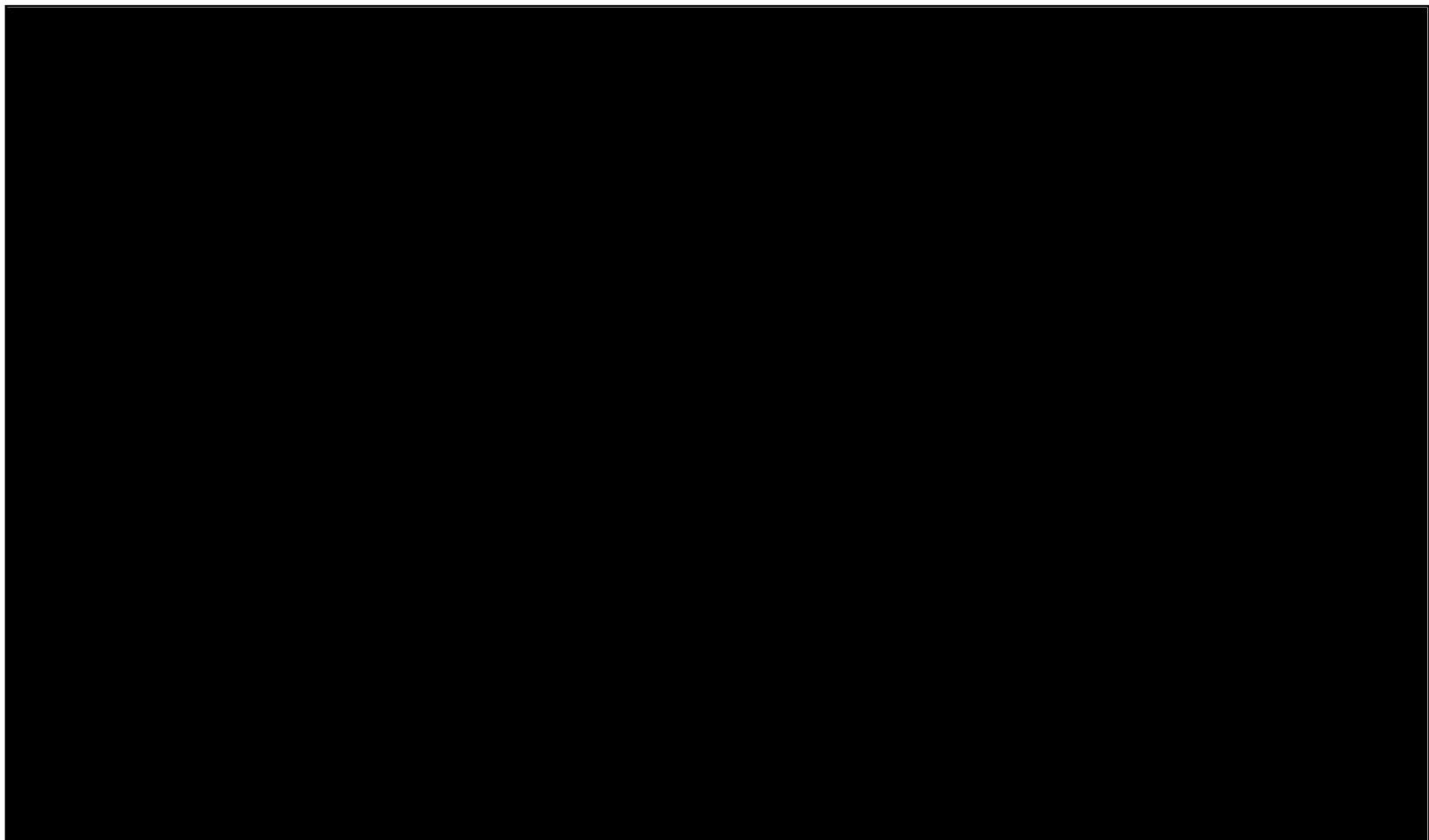
        <sequence>
          <interaction name=" " operation=" " channelVariable=" ">
            <description type=" " />
            <participation relationshipType=" " fromRole=" " toRole=" " />
            <exchange name=" " informationType=" " action=" "></exchange>
          </interaction>
          <assignment type=" " >
            <copy name=" " >
              <source expression=" " />
              <target variable=" " />
            </copy>
          </assignment>
        </sequence>
      </choice>
    </sequence>
  </workunit>

  <workunit name=" " guard=" " blocking=" ">
    <workunit>
  </parallel>
    
```

WS-CDL under the hood



WS-CDL under the hood



Formalisms

- Pi-calculus
 - Algebraic encoding of behavior
 - Reduction rules to show progress
- Static checking for liveness (livelock and deadlock)
- Composition
- Channel/port passing (dynamic topologies)
- Higher order calculii
 - Includes stochastic pi-calculus used for in-silco experiments to simulate interaction between molecules

Formalisms

| Model | Completeness | Compositionality | Parallelism | Resources |
|-----------------|-------------------------------------|-------------------------------------|-------------------------------------|-------------------------------------|
| Turing Machines | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Lambda | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Petri Nets | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| CCS | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| π | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |

Formalisms

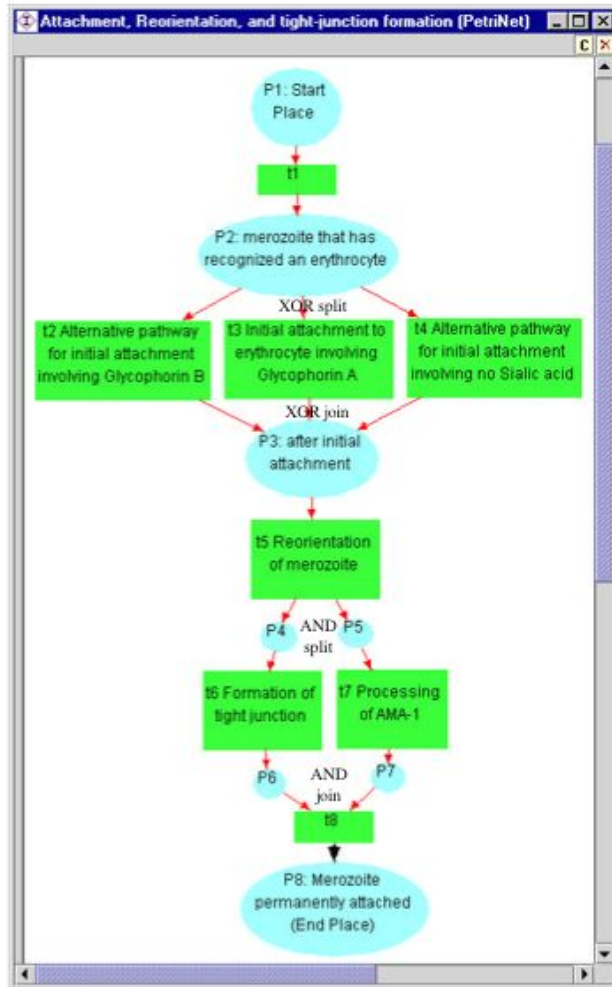
| Operation | Notation | Meaning |
|-------------|--|--------------------|
| Prefix | $^1 P$ | Sequence |
| Action | $\mathbf{x}(y), x(y)$ | Communication |
| Summation | $a.P + b.Q$ $\cdot \sum_{i=1..n} P_i$ | Distributed Choice |
| Recursion | $P = \{ \dot{E}.. \} P$ | Repetition |
| Replication | $!P$ | Repetition |
| Composition | $P Q$ | Concurrency |
| Restriction | $(\nu x)P$ | Encapsulation |
| Nothing | 0 | Do nothing |

System $= (!Client | !IdleServer)$
 $Client(o, c, req, rsp) = o.req_1.rsp1.req_2.rsp2.c.Client(o, c, res, rsp)$
 $IdleServer(o, req, rsp, c) = o.BusyServer(o, req, rsp, close)$
 $BusyServer(o, req, rsp, c) = req.rep.BusyServer(o, req, rsp, c) + c.IdleServer(o, req, rsp, c)$

Formalisms

| WS-CDL | The π-calculus |
|------------------|---|
| The channels | a pair of "ports" in the π -calculus |
| An interaction | message exchange that occurs between paired ports |
| message | a polyadic message |
| type | sorts |
| Guarded workunit | Process pattern |

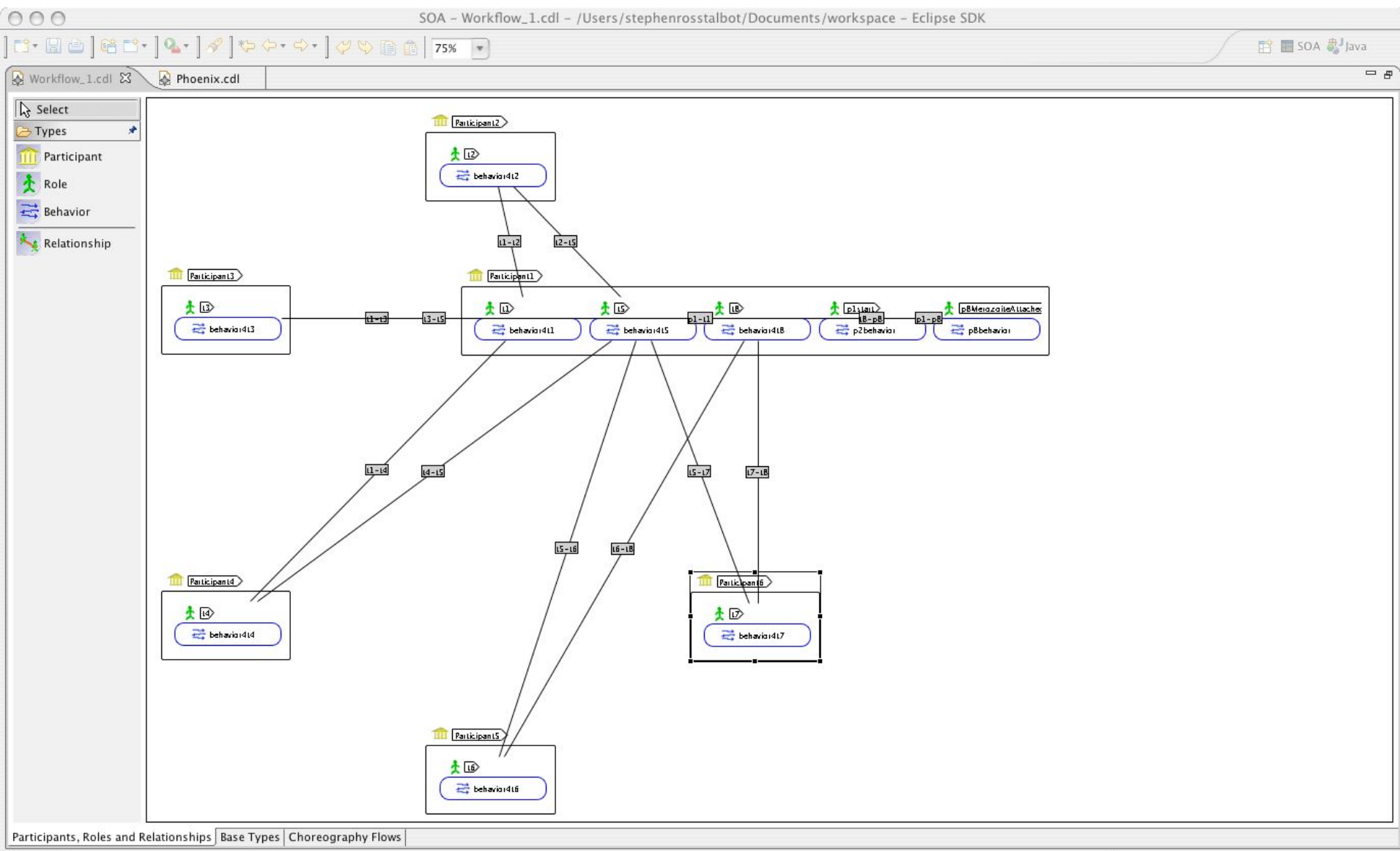
WS-CDL in practice



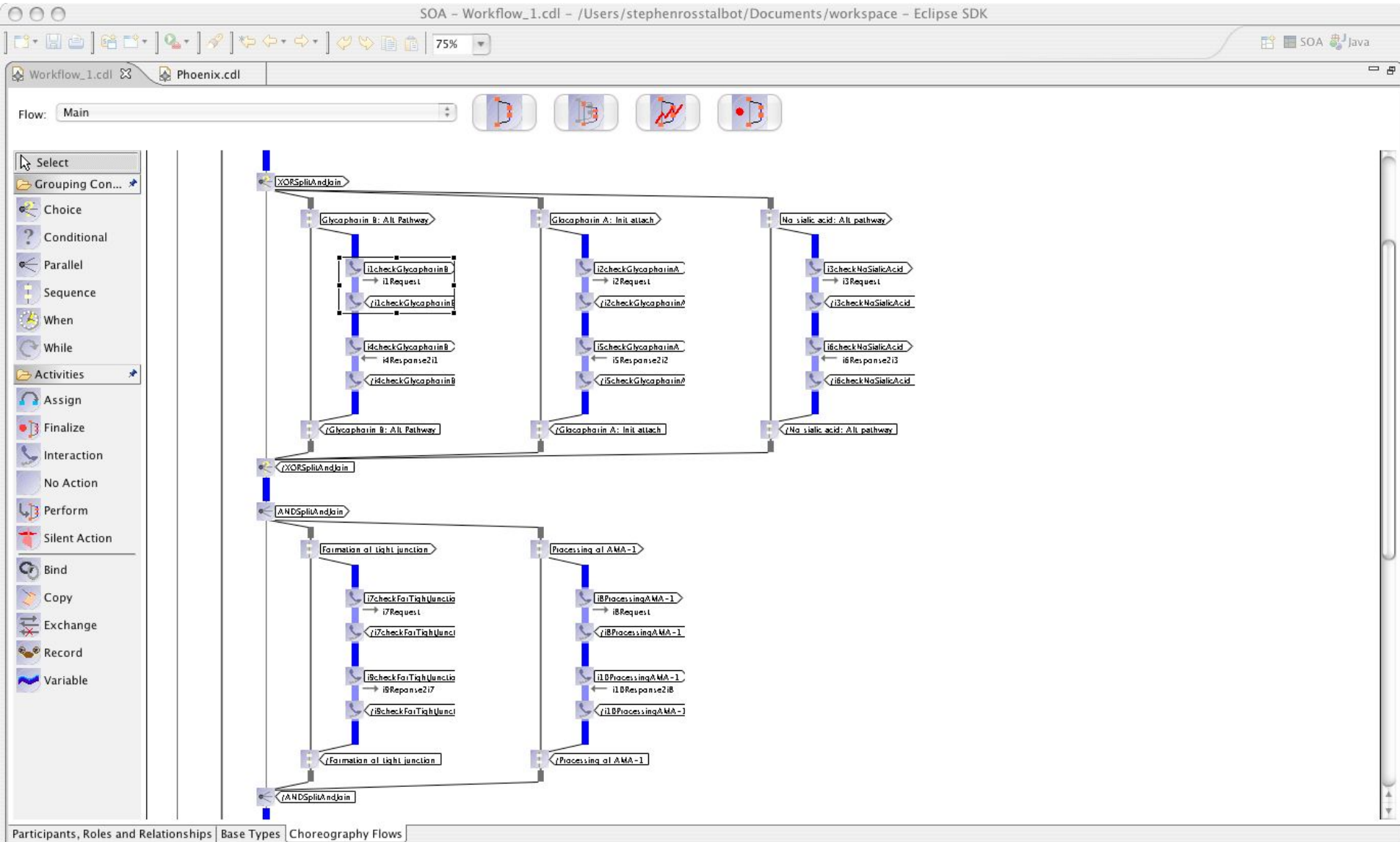
Modeling biological processes using Workflow and Petri Net models
 Mor Peleg, Iwei Yeh, Russ B. Altman
 Stanford Medical Informations, Stanford University, Stanford CA
 94305, USA

Figure 5. A Petri Net model of the invasion process, corresponding to the Workflow model shown in Figure 2. Places are shown as circles, and transitions as rectangles, and are labeled as t_1 .. t_8 . The first and last places in the Petri Net are also labeled (as P_1 and P_8). Implicit XOR split and joins are marked as "XOR split" and "XOR join", respectively. AND split and joins are also marked.

WS-CDL in practice



WS-CDL in practice



WS-CDL in practice

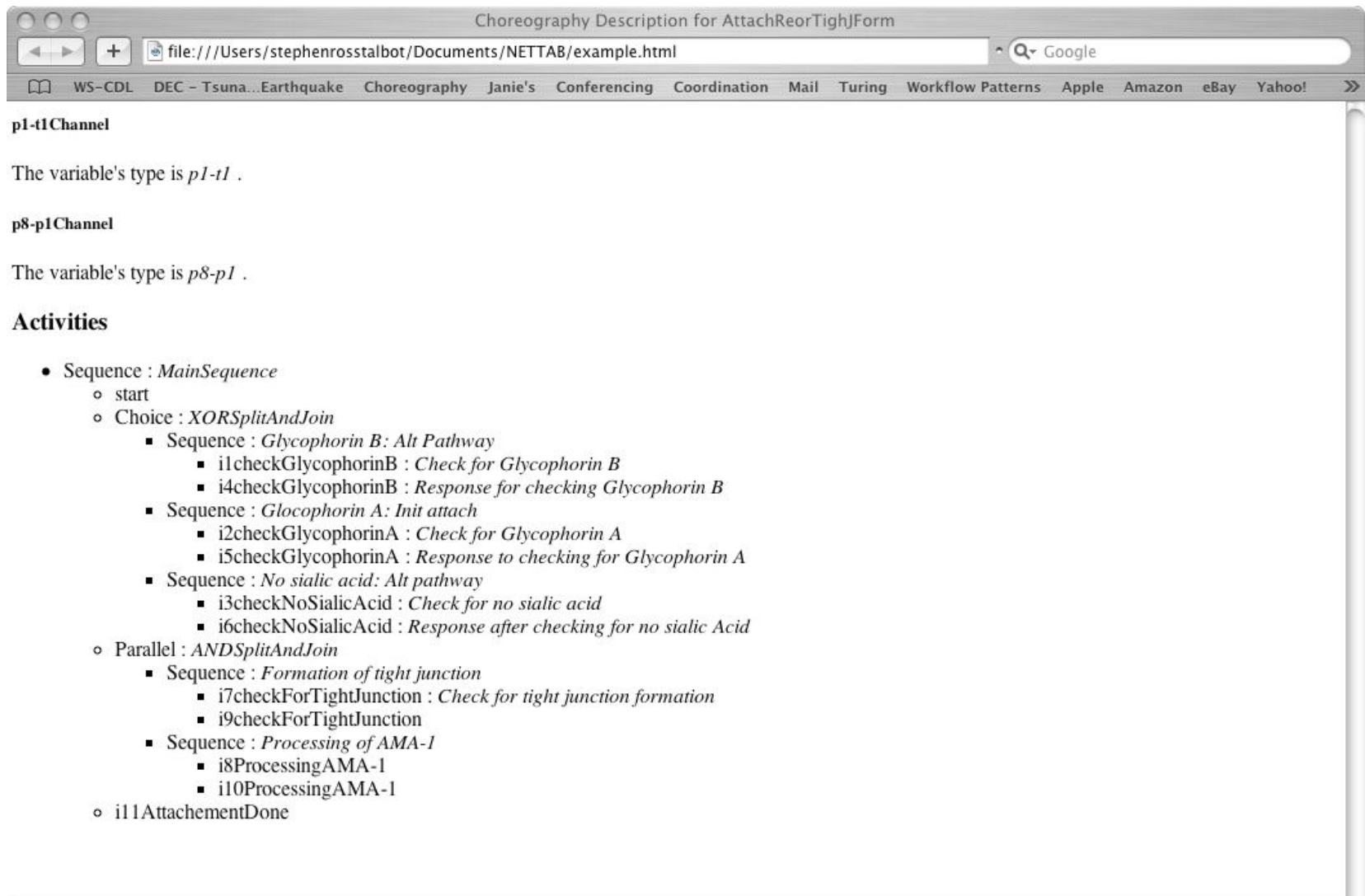
```

+ <variableDefinitions>
- <sequence>
  <description type="documentation">MainSequence</description>
+ <interaction name="start" operation="start" channelVariable="p1-t1Channel">
- <choice>
  <description type="documentation">XORSplitAndJoin</description>
- <sequence>
  <description type="documentation">Glycophorin B: Alt Pathway</description>
+ <interaction name="i1checkGlycophorinB" operation="checkGlycophorinB" channelVariable="t1-t2Channel">
+ <interaction name="i4checkGlycophorinB" operation="checkGlycophorinB" channelVariable="t2-t5Channel">
</sequence>
- <sequence>
  <description type="documentation">Glocophorin A: Init attach</description>
+ <interaction name="i2checkGlycophorinA" operation="checkGlycophorinA" channelVariable="t1-t3Channel">
+ <interaction name="i5checkGlycophorinA" operation="checkGlycophorinA" channelVariable="t3-t5Channel">
</sequence>
- <sequence>
  <description type="documentation">No sialic acid: Alt pathway</description>
+ <interaction name="i3checkNoSialicAcid" operation="checkNoSialicAcid" channelVariable="t1-t4Channel">
+ <interaction name="i6checkNoSialicAcid" operation="checkNoSialicAcid" channelVariable="t4-t5Channel">
</sequence>
</choice>
- <parallel>
  <description type="documentation">ANDSplitAndJoin</description>
- <sequence>
  <description type="documentation">Formation of tight junction</description>
+ <interaction name="i7checkForTightJunction" operation="checkForTightJunction" channelVariable="t5-t6Channel">
+ <interaction name="i9checkForTightJunction" operation="checkForTightJunction" channelVariable="t6-t8Channel">
</sequence>
- <sequence>
  <description type="documentation">Processing of AMA-1</description>
+ <interaction name="i8ProcessingAMA-1" operation="processAMA-1" channelVariable="t5-t7Channel">
+ <interaction name="i10ProcessingAMA-1" operation="processingAMA-1" channelVariable="t7-t8Channel">
</sequence>
</parallel>
+ <interaction name="i11AttachementDone" operation="start" channelVariable="p8-p1Channel">
</sequence>
</choreography>
</package>
    
```

What can I do with it?

- Simulation:
 - I can test it (simulate the message exchanges)
- Generation:
 - I can generate Java or BPEL code, deploy and execute or run it as a set of peer services (no single point of control)
- Documentation:
 - I can produce documentation (html) to describe it
- Execution:
 - I can monitor the threads of execution through it

What can I do with it?



Choreography Description for AttachReorTighJForm

file:///Users/stephenrosstalbot/Documents/NETTAB/example.html

WS-CDL DEC - Tsuna... Earthquake Choreography Janie's Conferencing Coordination Mail Turing Workflow Patterns Apple Amazon eBay Yahoo!

p1-t1 Channel

The variable's type is *p1-t1* .

p8-p1 Channel

The variable's type is *p8-p1* .

Activities

- Sequence : *MainSequence*
 - start
 - Choice : *XORSplitAndJoin*
 - Sequence : *Glycophorin B: Alt Pathway*
 - i1checkGlycophorinB : *Check for Glycophorin B*
 - i4checkGlycophorinB : *Response for checking Glycophorin B*
 - Sequence : *Glocophorin A: Init attach*
 - i2checkGlycophorinA : *Check for Glycophorin A*
 - i5checkGlycophorinA : *Response to checking for Glycophorin A*
 - Sequence : *No sialic acid: Alt pathway*
 - i3checkNoSialicAcid : *Check for no sialic acid*
 - i6checkNoSialicAcid : *Response after checking for no sialic Acid*
 - Parallel : *ANDSplitAndJoin*
 - Sequence : *Formation of tight junction*
 - i7checkForTightJunction : *Check for tight junction formation*
 - i9checkForTightJunction
 - Sequence : *Processing of AMA-1*
 - i8ProcessingAMA-1
 - i10ProcessingAMA-1
 - i11AttachementDone

Summary

- WS-CDL is the Web Services Choreography Description Language (CDL for short)
- Common formalism (the pi-calculus) between SOA/WS-CDL and bioinformatics
- WS-CDL to describe workflow, Orchestration to execute workflow
- WS-CDL description for compliance

Acknowledgments

- Greg Meredith of Djinnisys
- Maria Mirto (for some example and early access to papers)
- Dr Gary Brown (pi4tech)

Grazie
Thank You

Q & A