

BPM and Intalio

Tuesday, Sept 23 2008

"Having a real, executable BPM 2.0 environment at your fingertips makes you rethink how you architect solutions. It simply gets you wild again. Keep up the good work!"

—Jose Diego de La Cruz, EPFL

"I can do something without lots knowledge about computing."

—Han Jie Davis, Consultant

"Intalio|Designer is a very productive tool for process modeling! You can really lay things out very quickly!"

—Tom Debevoise, BKA, Inc.

Business Process Management

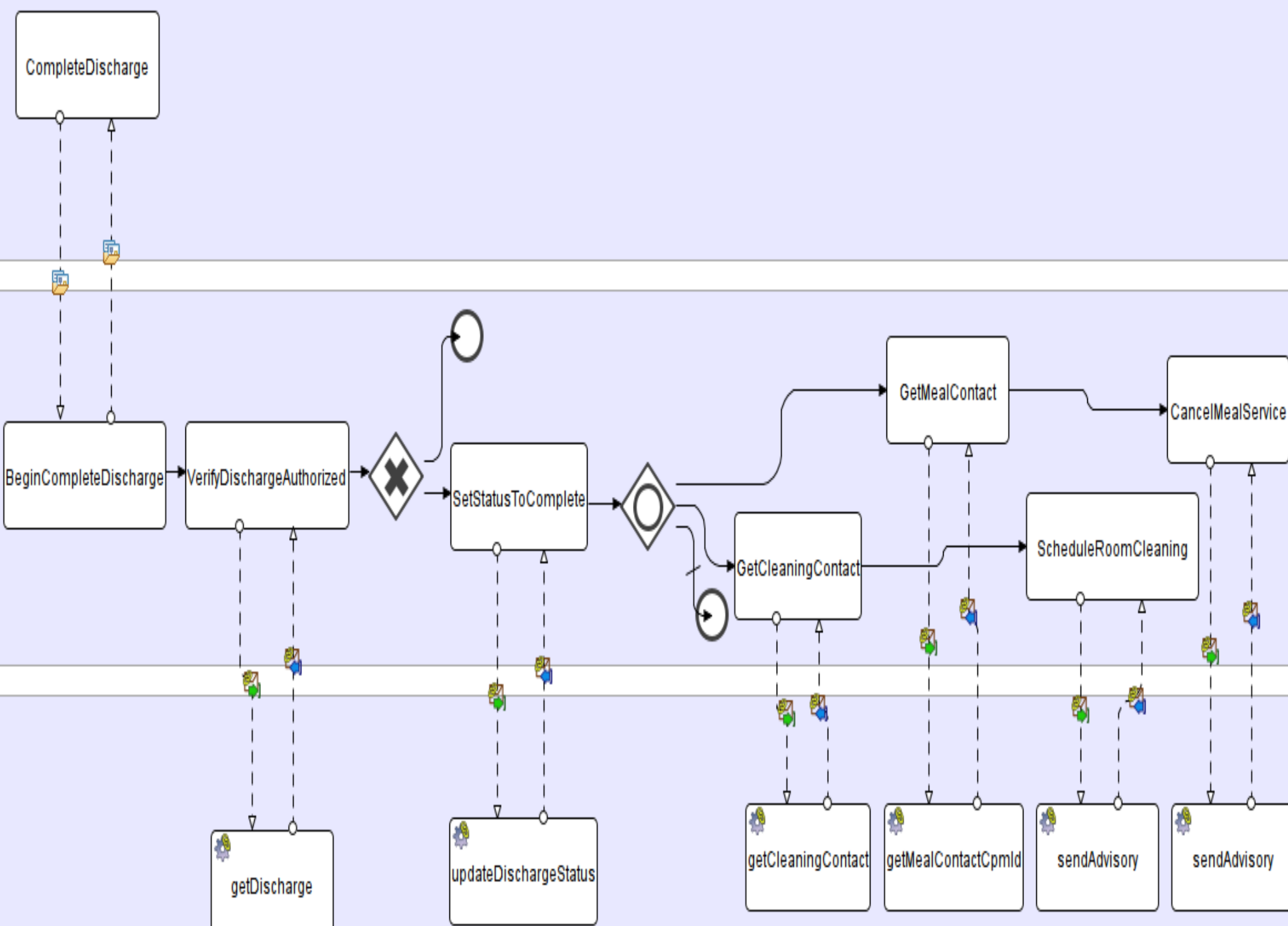
- Graphical notation agreed upon by the Object Management Group to define business processes workflows.
 - Graphically define process, and visually see how data is moving from task to task.
 - Simple notation, can be used by domain experts with little programming knowledge (in theory).
-
-

Intalio

- One of the most complete implementations of BPMN, as well as BPEL.
 - Details are hidden about how the BPMN gets transformed to BPEL. (possible optimization issues?)
 - Follows BPMN standards pretty well, but cases exist where they are different.
 - Not too much of a problem since the nature of a graphical language makes things intuitive.
 - Uses XSD for message schema's instead of DTD
-
-

Intalio

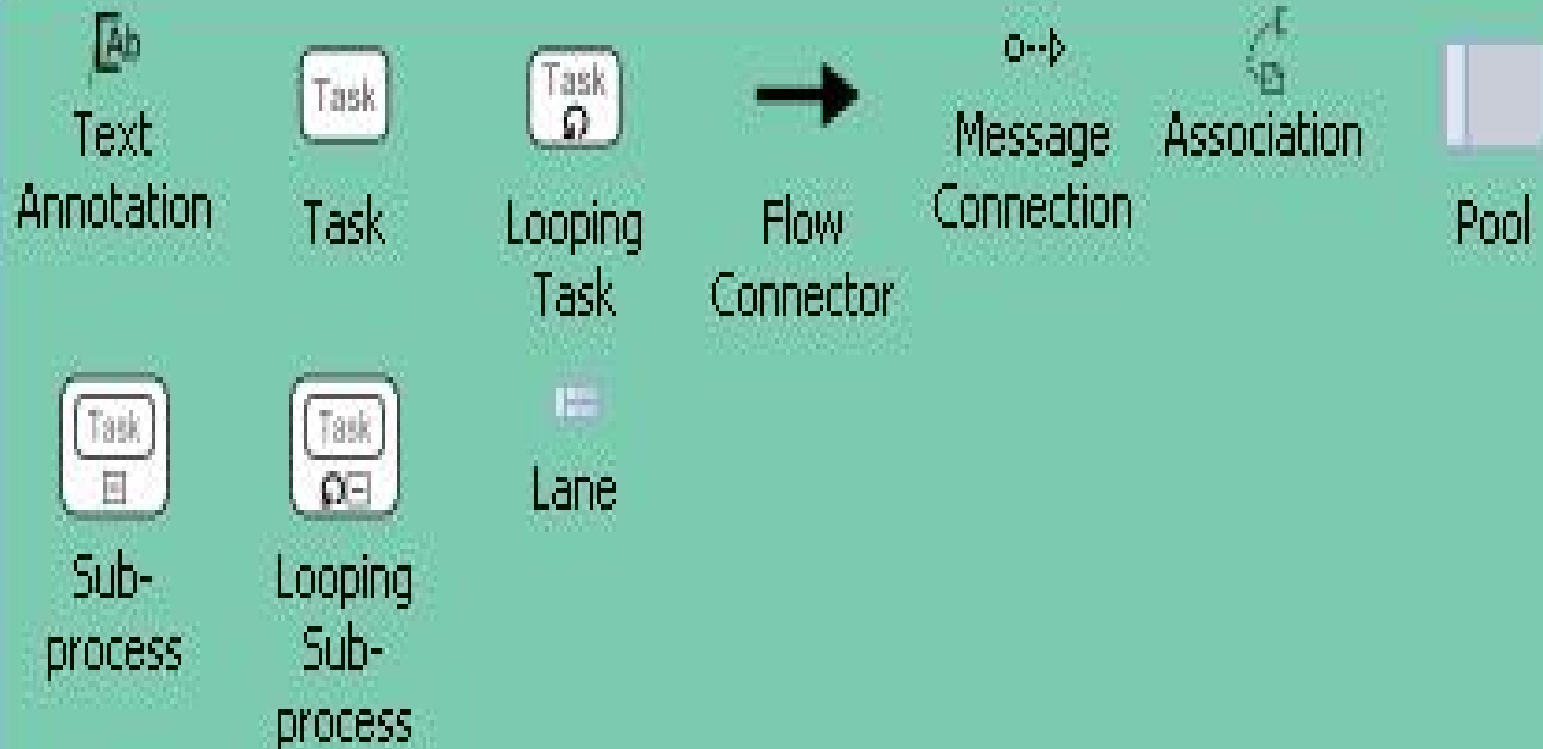
- Broken up into two parts
 - Designer
 - Creates business logic, and data mappings.
 - Server
 - Runs under Geronimo, exposes Intalio WSDL to kickstart an Intalio process (how to do this later).



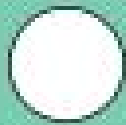
Opening an Intalio Business Logic

- Follow the steps listed below to import the business logic into Intalio:
 - First, open the Intalio designer.
 - Select the File->Import menu, which will cause a popup screen to appear.
 - From the popup, select 'Existing Projects into workspace' and then click the 'Next' button.
 - Select the 'Select achieve file' radio button and browse to the location of the BPEL logic zip file.
 - Finally, click 'Finish' and all the files required to modify Intalio will be loaded.
-
-

Basic BPMN Shapes



Start Events



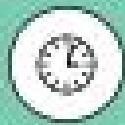
Empty Start



Message Start



Rule Start



Timer Start

Intermediary Events



Empty Intermediate



Message Intermediate



Timer Intermediate



Error Intermediate



Compensation Intermediate



Rule Intermediate



Cancel intermediate

End Events



Empty End



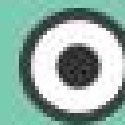
Message End



Error End



Compensation End



Terminate

Gateway Shapes



Exclusive
Data-based



Exclusive
Event-
based



Inclusive
Data-based



Parallel

See more detail at:

<http://bpms.intalio.com/reference-guides/intalio-bpms-designer-bpmn-flow-objects.html>



Adding a new WSDL

- In the 'Project Explorer' window, select the folder for the Business Logic. Right click this directory, select New->Other->File, and name the file appropriately.
 - Copy the WSDL into this new file and save the project.
 - Drag the file name for the WSDL onto the 'Business Project Schema Pool' where you want to add the web service. The name of the WSDL should now appear in the schema, which means the WSDL is successfully part of the schema.
 - Create message connections to and from the WSDL from another task. This will specify the function the WSDL will call and provide the information format for the data it returns.
 - Select the WSDL in the 'Project Navigator' window and expand it until all of the functions are displayed.
 - Select the function to be added to the message connector and then drag and drop it onto the message connector itself. Drag the 'get' functions onto the arrow pointing towards the WSDL and the 'response' onto the arrow pointing away from the WSDL.
-
-

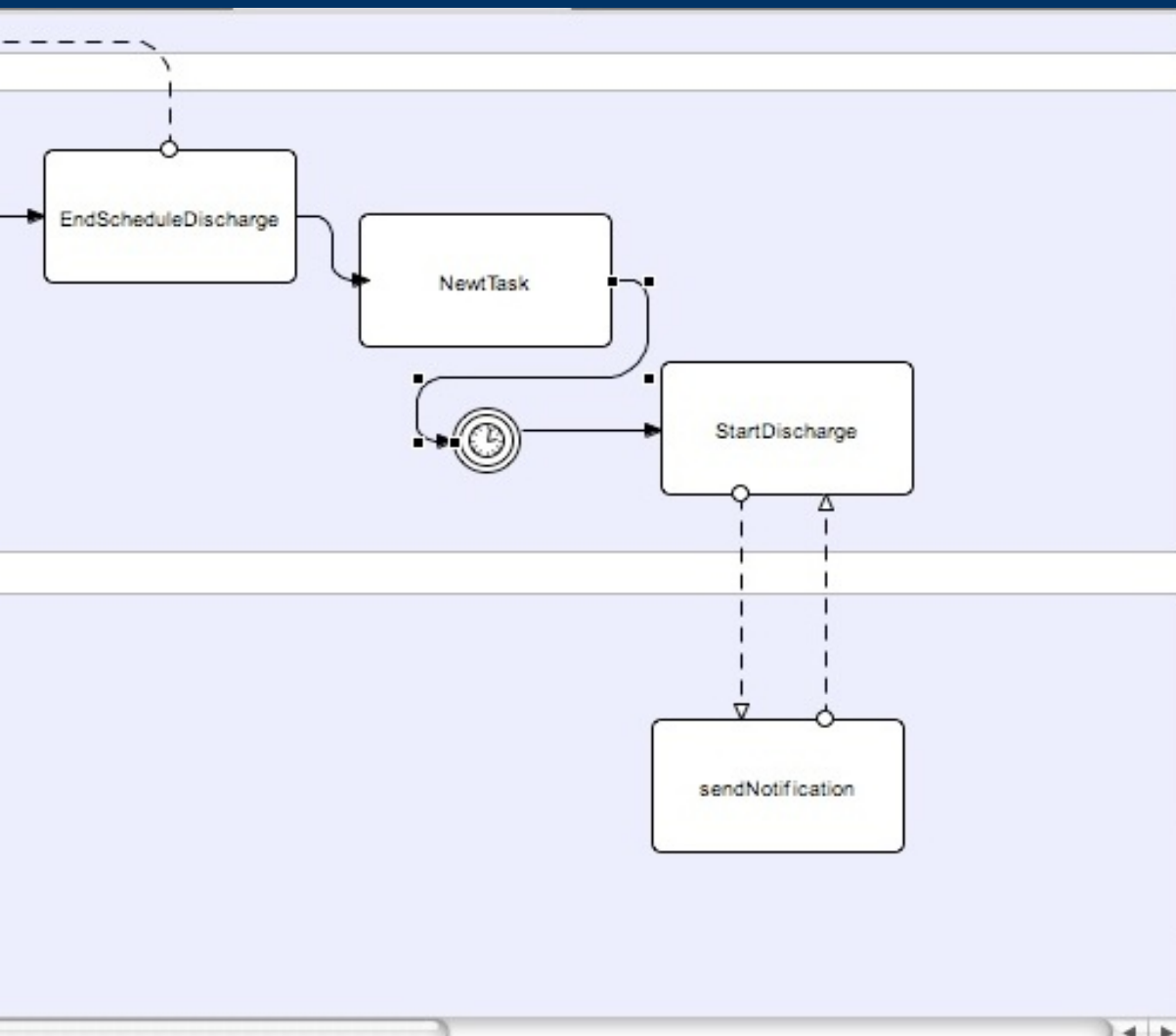
Mapping results of WSDL's

- Select Window->Show View->Mapper to open the 'Mapper'.
 - Select a task, one of the boxes in the schema, and the 'Mapper' will show all the variables that can be mapped on the left hand side and all of the variables that can be mapped to on the right hand side.
 - Delete the existing arrow on the mapped to side.
 - Select the variable of the response name from the WSDL added on the left, then click it and drag the arrow that is added to the variable on the right side where this data is wished to be mapped to.
-
-

Modifying Business Logic

- Open the schema.
 - Select Window->Show View->Palette, which will show all of the different business logic functions that can be added to the schema.
 - To modify the current logic, drag and drop a task from the palette onto the schema.
 - Add flow connector arrows from the palette to and from where you want the new task to go in relation to the new business logic.
-
-

Intalio Palette



Palette

- Select
- Insert space

Basic BPMN Shapes

- Text Annotation
- Task
- Looping Task
- Flow Connector
- Message Connection
- Association
- Pool
- Sub-process
- Looping Sub-process
- Lane

Start Events

- Empty Start
- Message Start
- Rule Start
- Timer Start

Intermediary Events

- Empty Intermediate
- Message Intermediate
- Timer Intermediate
- Error Intermediate
- Compensation
- Rule
- Cancel

Invoking a web service from Intalio

- Generating a web service that Intalio understands can be tricky. There are several important issues to consider.
 - Make sure any objects that need to be passed to and from the web service are Java Beans. Beans must contain a default no-argument constructor, contain appropriately named getter and setter functions for each stored value, and implement the 'Serializable' interface.
 - Once the classes are correctly created, the service can be created by compiling the classes into the WEB-INF/classes folder or into a jar in the WEB-INF/lib folder. To expose the service, add a <service> section into the WEB-INF/server-config.wsdd file:

```
<service name="MyWebService" provider="java:RPC" style="wrapped" use="literal">
  <parameter name="allowedMethods" value="*" /><parameter name="className"
value="name.of.class.to.expose.as.a.Webservice" /><namespace>http://namespace.for.this.service</namespace>
<beanMapping languageSpecificType="java:name.of.parameter.Object" qname="ns:MyObject"
xmlns:ns="http://namespace.for.this.service"/><beanMapping
languageSpecificType="java:another.param.or.return.Object" qname="ns:ReturnObject"
xmlns:ns="http://namespace.for.this.service"/></service>
```
- Intalio requires that any web service it will consume should be of the “wrapped” and “literal” styles. Additionally, to avoid problems using the web service from Intalio, all members of the service should be located inside the same namespace
 - Intalio has problems understanding XML <include> directives.

Exposing an Intalio Project

- Generally, we will want to invoke the Intalio Process from a JSP page or Java code. The easiest way to accomplish this is to use the WSDL2Java tool to generate stub Java classes from the Intalio WSDL.
- The first step is to locate the WSDL associated with the Intalio Process. Navigating to <http://my.intalio.server:8080/ode> and clicking “services” will list each Process deployed to the Intalio Server. Locate the appropriate service and click the provided link to be taken to the WSDL. Save this file to disk, as it may need to be edited to work with WSDL2Java.
- Invoke WSDL2Java:
 - `java org.apache.axis.wsdl.WSDL2Java TargetService.wsdl -a -pmy.stubs.ns`

Exposing an Intalio Process

- If WSDL2Java fails with an “unable to find specified file” error, the specified WSDL references an XML schema document with a relative path that is incorrect. For example, if your WSDL contains:
 - `<xs:import namespace="urn://keypoint/schedule-discharge" schemaLocation="schedule-discharge/ScheduleDischarge/LogicPool/InterfacePool?xsd=xsd0"/>`
- The schemaLocation element needs to be changed to an absolute path so that WSDL2Java can locate it. The corrected path for the provided example might be:
 - `<xs:import namespace="urn://keypoint/schedule-discharge" schemaLocation="http://keypoint.cs.colorado.edu:8080/ode/processes/schedule-discharge/ScheduleDischarge/LogicPool/InterfacePool?xsd=xsd0"/>`
- Once these changes have been made, WSDL2Java should produce the required stub classes in the namespace specified by the -p argument. These stub classes can now be used to invoke the Intalio web service in the same way as any other web service.