NOTES On Nimbus Control Model -> XPDL Import.

# 2012 November Release Goals

The following are the stage goals for initial release of Nimbus importer.

(grey means not done yet)

1. Stage 1 - Import lines and objects from simple diagram as BPMN process
   1. Standard Studio for designers import wizard (Simple import using XSLT).
   2. Validate that the incoming import is Control model **Simplified** export.
   3. Additional post-import Java handling auto-layout.
   4. Equivalences …

| **Nimbus** | **Studio** |
| --- | --- |
| **Process diagram** | * **Business Process** per diagram |
|  |  |
| **Activity** | * Placed in lane appropriate to selected resource. If no resource selected OR multiple resources selected then is added to default lane. * Task Type:   + Create sub-process task for drill-down activity     - Parameter mappings?     - Link up sub-processes when import is from different file (find process with same id and add parent package).   + Else if non-drill-down activity has resource then User Task   + If activity has no resource then task type none (or service?) |
| **Activity Notes** | * **(notes bubble in Nimbus) become text annotation attached to task.** |
| **Activity Commentary** | * Start of Task Description. |
| **Statement Links** | * (normally things like statements of require compliance with standards and so on) are appended to task description. |
| **Activity URL** | * Link to Nimbus Control documentation for activity. * Possibly add new control to launch url in studio web viewer on property sheet and activity tooltip |
| **Activity Resource** | * One **package** **participant** per unique named resource. * One “Lane” for all activities.   + Have made this configurable in xslt (*not intended to be user selectable)* to lane-per-resource plus default lane for non-resourced/multi-resource activities. * Lowest level task activities (i.e. not sub-process) are assigned participant(s) for the activity resource(s). |
|  |  |
| **Connections** | * Connections can mean several different things in Nimbus process diagrams (incoming/outgoing data, trigger for action and so on) and depends only on a customer’s chosen use or methodology. * As there is likely to be a suggested methodology for the level of Nimbus diagram in which the interface between high-level documentation-only parts of the process and lower level to-be-automated parts we will simply have to choose one semantic meaning for the connections in order to make things simple and understandable for users. * We have chosen to treat connections **both** as routing-between-activities and as “incoming / outgoing data”. * Therefore multiple connections in the same direction, between the same Nimbus activities will be collapsed into **a** **single** **connection** and parameters / data fields and activity data associations will be created for connections. * All start connections (connection without source object) are connected to a single start event. (Note: There was a choice between this and multiple start events (one per start connection). We decided to have a single start event implying that all data is passed and all connections happen at start of process). * All end connections (connection without target object) are all connected to a single end event. |
| **Start / End Connection**  **Label** | * Each unique start/end connection label will become a **formal parameter.**   + The ‘uniqueness’ is case insensitive and white-space normalised (so “Connection Label” and “ Connection LABEL “ are treated as being the same and will therefore be treated as same process data). * The formal parameter **mode** is set according to whether the label appears on a start connection (*In*) or end connection (*Out*) or both (*In-Out*). * All formal parameters will be associated in the start-event activity interface data. |
| **Connection Between Activities**  **Label** | * Each unique label on connections between activities *that is not also used on a start/end connection* (see above) will become a **data field.**   + The ‘uniqueness’ is case insensitive and white-space normalised (so “Connection Label” and “ Connection LABEL “ are treated as being the same and will therefore be treated as same process data). |
| **All Connection Labels** | * As the connection labels are treated as process data, interface data associations are added to user tasks for the data identified by its incoming / outgoing connections. * The interface data **mode** is set according to whether the label appears on an incoming connection (*In*) or outgoing connection (*Out*) or both (*In-Out*). |
| **Connection Commentary** | * Added to **formal parameter / data field description**   + For multiple connections with equivalent labels, all commentaries are added. * And **sequence flow description**.   + For multiple connections between same activities all commentaries are added. |
|  |  |
|  |  |

1. Stage 2: ???