**DDI Sprint #3
Vancouver, BC, Canada**

**Wednesday, March 26, 2014**

**Minutes**

Sprint participants:

Therese Lalor (Project Manager and Sprintmeister)

Jay Greenfield, Arofan Gregory, Larry Hoyle, Jon Johnson, Alan Li, Ron Nakao, Wendy Thomas, Joachim Wackerow, Mary Vardigan

**Architecture**

Achim showed a slide on the proposed architecture with the layers of:

* Primitives (string, etc.)
* Extended Primitives (LabelType, etc.)
* Building Blocks (Instance Variable, etc.)
* Composition/Cluster
* Simple or complex questionnaire, for example

In this model, compositions/clusters can be instantiated as, for example, a simple questionnaire or a complex questionnaire. We also have views for publication and an abstract type (in this case, instrument or source).

The group commented that this architecture is really elegant, but that we should test this by doing some modeling. The question was raised of what is the role of composition and do we need that layer in the hierarchy. This prompted a longer discussion about packaging and versioning. It was also discussed that perhaps views should bring in building blocks rather than compositions to make it as easy as possible for new use cases. We need to ground the conversation in examples.

Anyone should be able to make a view, but there are views that are officially endorsed. We have developed a simple syntax to make a view. You can get complex by adding objects or by specializing objects that already exists. View can only add objects.

We need a discussion about versioning and releases before we can make decisions on the architecture.

The point was also raised that we need deliverables. Can we get any specific area modeled during this time? We need to balance the development of the architecture vs. moving to deliverables. When developing 3.0, we didn’t spend enough time on the architecture and moved too quickly to implementation. The major advantage of spending more time on this now is that it will result in a more robust and flexible architecture. Now, since we have DDI 3.2 in existence, we are not under the same time pressure.

**Process Model**

Jay provided some slides showing a possible process model architecture. GSIM provides a core conceptual model, (human-readable), OWL-S implements GSIM (machine-readable), and DDI specializes OWL-S atomic processes.

Jay suggested a protocol execution step that involved control constructs. He will be attending a DDI workshop at the Census Bureau soon, and will be modeling an adaptive design, which Census is interested in. The group thought that making a simple process model available for extension and use is important. Complexity comes in when we chain processes.

In terms of data transformation, eventually you have to describe a chain of transformations that you pull in and you show their relationships with each other in terms of input and outputs. Each process step might have a lifecycle event that monitors the process step

Achim showed a framework of the process to develop the model.



Content is captured in Drupal, which exports to XML (XMI) to import into Enterprise Architect. XMI can also be exported out of EA to generate the bindings. Documentation can be pulled into the bindings. This picture has changed since we added views. We have added an XML description of the view. Documentation is combined with the view. All the things are processed at the same time with views.

In Drupal we have people defining views not packages. How do we organize the views in Drupal? We need to export machine-readable version of views. Is the view any part of the UML model? Drupal will need a new packaging method. Views should now be lists of references.

We need to capture content from working groups using Drupal, which is only an input mechanism to the overall modeling process. It is too large a burden for the content groups to determine what is going into the model and what is a view. The modeling group needs to build the library and based on the content capture the views are defined. Capturing content and defining views are two different steps.

Groups still need to look at objects in the model to pull in. Drupal was created to group into packages but we need the capability in Drupal of defining views. The other thing we need to do is group into packages. We will need to ask for additional functionality in Drupal. We will need Drupal developers to do the Drupal work. We will need to export views and have all content versioned and archived.

How is the library built? It’s not enough to bring in classes or objects downstream. We will also have to bring in profiles or the specifications for the views and understand what views are connected in terms of simple and advanced. This is a strong argument to have information about how views are organized and packages are managed in the same place.

Content people would create objects they needed for their views. It would be easier to take views and subset them for the library. We need a new diagram of the workflow.

This is trying to represent a workflow but it also gets into what is not resolved yet about the architecture. It’s more efficient to filter and then transform.