Adaptive Protocols and the DDI 4 Process Model

Jay Greenfield
Sophia Kuan
Booz Allen Hamilton

NSF-Census Research Network Meeting Spring 2014

Suitland, MD
Agenda

• Describe several uses of an emerging process model that is part of the future DDI 4 specification

• The uses we will cover today include:
  – The representation and execution of multi-mode contact procedures
  – The representation and execution of data transformation pipelines
  – Putting the two together in a process model able to both represent and facilitate the execution of data driven adaptive protocols
Disclaimer

GSIM 1.1 is a mature specification. So is OWL-S. So is DDI-3 which uses OWL-S.

However, DDI 4 is in the early stages of development. While it integrates with mature specifications, it is very much a work in progress. Many of its details remain to be finalized.
Background: What is DDI 4?

• It is the next major version of the DDI specification
• Instead of being XML-based it is information model (UML) based
• Being model-based enables DDI 4 to more easily:
  – *Interact* with other UML models including GSIM
  – *Modularize* the specification by resolving it into a set of more or less specialized, extensible functional views
  – *Create functional views* that support the use of heterogeneous data sources (including auxiliary data, process data and registries) across the data lifecycle
Background: What is GSIM?

We need consistent information

• Modernization of statistics requires:
  – Reuse and sharing of methods, components, processes and data repositories
  – Definition of a shared “plug-and-play” modular component architecture
• The Generic Statistical Business Process Model (GSBPM) will help determine which components are required
• GSIM will help to specify the interfaces

From What is GSIM? at GSIM 1.1 portal
**Background: What are DDI 4 Functional Views?**

- The primary publications of the DDI Alliance are functional views.
- From these views various artifacts including XML schemas and RDF vocabularies are generated.
Our use case: meet the players…
# GSBPM

<table>
<thead>
<tr>
<th>Specify Needs</th>
<th>Design</th>
<th>Build</th>
<th>Collect</th>
<th>Process</th>
<th>Analyse</th>
<th>Disseminate</th>
<th>Evaluate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.1</td>
<td>2.1</td>
<td>3.1</td>
<td>4.1</td>
<td>5.1</td>
<td>6.1</td>
<td>7.1</td>
</tr>
<tr>
<td>Identify needs</td>
<td>Design outputs</td>
<td>Build collection instrument</td>
<td>Create frame &amp; select sample</td>
<td>Integrate data</td>
<td>Prepare draft outputs</td>
<td>Update output systems</td>
<td>Gather evaluation inputs</td>
</tr>
<tr>
<td></td>
<td>1.2</td>
<td>2.2</td>
<td>3.2</td>
<td>4.2</td>
<td>5.2</td>
<td>6.2</td>
<td>7.2</td>
</tr>
<tr>
<td>Consult &amp; confirm needs</td>
<td>Design variable descriptions</td>
<td>Build or enhance process components</td>
<td>Set up collection</td>
<td>Classify &amp; code</td>
<td>Validate outputs</td>
<td>Produce dissemination products</td>
<td>Conduct evaluation</td>
</tr>
<tr>
<td></td>
<td>1.3</td>
<td>2.3</td>
<td>3.3</td>
<td>4.3</td>
<td>5.3</td>
<td>6.3</td>
<td>7.3</td>
</tr>
<tr>
<td>Establish output objectives</td>
<td>Design collection</td>
<td>Build or enhance dissemination components</td>
<td>Run collection</td>
<td>Review &amp; validate</td>
<td>Interpret &amp; explain outputs</td>
<td>Manage release of dissemination products</td>
<td>Agree an action plan</td>
</tr>
<tr>
<td></td>
<td>1.4</td>
<td>2.4</td>
<td>3.4</td>
<td>4.4</td>
<td>5.4</td>
<td>6.4</td>
<td>7.4</td>
</tr>
<tr>
<td>Identify concepts</td>
<td>Design frame &amp; sample</td>
<td>Configure workflows</td>
<td>Finalise collection</td>
<td>Edit &amp; impute</td>
<td>Apply disclosure control</td>
<td>Promote dissemination products</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.5</td>
<td>2.5</td>
<td>3.5</td>
<td>5.5</td>
<td>5.6</td>
<td>5.7</td>
<td>5.8</td>
</tr>
<tr>
<td>Check data availability</td>
<td>Design processing &amp; analysis</td>
<td>Test production system</td>
<td>Derive new variables &amp; units</td>
<td>Calculate weights</td>
<td>Calculate aggregates</td>
<td>Finalise data files</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1.6</td>
<td>2.6</td>
<td>3.6</td>
<td>6.1</td>
<td>6.2</td>
<td>6.3</td>
<td>6.4</td>
</tr>
<tr>
<td>Prepare business case</td>
<td>Design production systems &amp; workflow</td>
<td>Test statistical business process</td>
<td>Finalise production system</td>
<td>Finalise outputs</td>
<td>Finalise outputs</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Manage user support</td>
<td></td>
</tr>
</tbody>
</table>

From *Introducing the GSBPM* at the [GSBPM portal](https://www.gsbpm.com)
In the adaptive protocol use case...

• The relationship between Collect and Process is non-linear:
  – We use administrative and other data collected and processed previously together with almost real time processing of current data to guide data collection now.
  – Collect can depend on Process and vice versa.

• Although GSBPM and the business processes that GSBPM represents were not intended to form a linear model, the systems that we Build behave by and large in a linear way.
GLBPM

From: Generic Longitudinal Business Process Model
GLBPM

• Because the **L** in GL**B**PM is *longitudinal*, GLBPM thinks in circles

• And the systems we **Build** to support longitudinal studies are formed around an **Archive**

• Think of the Census as a longitudinal study

• There are many lessons learned from ordering processes around archives and archival services
Meet the Information Model we will use to **design**, **build** and **collect** the adaptive protocol use case...

DDI specializes OWL-S atomic processes with:

- Contact procedures
- Data transformation objects

**OWL-S** implements **GSIM**

**GSIM provides core conceptual model**

- Process
  - Atomic Process
  - Composite Process
    - Atomic Process
- Business Process
  - Process Design
  - Process Control
  - Process Step
Using GSIM, how might we represent an adaptive design?

• Imagine that we want to use the current representativeness of study participants to decide if, for a specific individual, we want to engage in a refusal conversion.

• In such a design we would conditionally attempt to convert based on the calculation of a statistic.

• Let’s first examine how GSIM might represent the calculation of this statistic...
The design of a calculation...

- Following FDA guidelines for adaptive designs we might undertake “analyses of the accumulating study data...at prospectively planned time points within the study”
- This might entail comparisons over time of historical demographic data with current demographic data
- In the process we might want to \textit{integrate data} (GSBPM 5.1) and \textit{calculate aggregates} (GSBPM 5.7)
GSIM Process Step (Top Level)
A tale of two process designs and the case for data federation

From Best Practices in SAS Management for Big Data

Booz | Allen | Hamilton
A tale of two process designs and the case for data federation

• **Data is too sensitive:** Organizations don’t want to provide direct access to data sources.

• **Data is too diverse:** Data is stored in multiple source systems that all have different security models, duplicate users and different permissions.

• **Data is too ad hoc:** When data is changing frequently, constant updates are needed to maintain integration logic. It becomes difficult to make a repeatable integration process, especially if there are many data integration applications that need access to the same data.
Our process design is...

• ...a data federation data lake that collects everything and supports dive in anywhere, flexible access data processing

• The data lake is, in turn, traversed in a series of sub-steps that together form a map reduce algorithm for numerical summarization

• The algorithm both integrates and aggregates...
Our process design is...

1. Input data, such as a long text file, is split into key-value pairs. These key-value pairs are then fed to your mapper. (This is the job of the map-reduce framework.)
2. Your mapper processes each key-value pair individually and outputs one or more intermediate key-value pairs.
3. All intermediate key-value pairs are collected, sorted, and grouped by key (again, the responsibility of the framework).
4. For each unique key, your reducer receives the key with a list of all the values associated with it. The reducer aggregates these values in some way (adding them up, taking averages, finding the maximum, etc.) and outputs one or more output key-value pairs.
5. Output pairs are collected and stored in an output file (by the framework).
GSIM Process Step (Top Level)

GSBPM 5.1, 5.7

Data Lake

Numerical summarization map reduce algorithm

FDA Adaptive Design Guidelines

GSIM Process

Step

(Top Level)
Enter OWL-S…

From *OWL-S: Semantic Markup for Web Services*
Enter OWL-S...

- Previously, in DDI 3, OWL-S information objects were utilized to specify control constructs and the skip logic of questionnaires.
- OWL-S did this with such specificity that software connectors were developed by the Australian Bureau of Statistics and others that take DDI 3 OWL-S as input and code complex instruments in Blaise as well as other survey systems.
Enter OWL-S...

• Now, in DDI 4, OWL-S information objects are about to be put into play alongside GSIM that can specify both **parallel processing** and **sequential processing**

• OWL-S describes **parallel processing** in enough detail that, in the adaptive design use case, it will be able to assist in the **automatic production** of map reduce algorithms for constructing numerical summarizations
Enter OWL-S...

• Also, OWL-S describes *sequential processing* in enough detail that, in the adaptive design use case, it will be able to assist in the *automatic production* of *schedulers* that conditionally determine and produce next actions – such as refusal conversion – in multi-mode contact procedures.
Enter OWL-S...

• Finally, using OWL-S, it is possible to define a **service profile** for each process.
• In conjunction with a **software agent**, the service profile enables:
  – Automatic web service discovery
  – Automatic web service invocation
  – Automatic web service composition and interoperation
Enter OWL-S…

- In this context a software agent might be a **program** that takes an OWL-S composite process as input and orchestrates a series of **Web Service Business Process Execution Language** (WSBPEL) calls.
In summary in DDI 4...

- Application Layer
- Information Model Layer
- Business Process Layer

WSBPEL
Others

GSIM
OWL-S

GLBPM
GSBPM
"That's all Folks!"
Appendix

BIG DATA PRIMER
Big Data Primer (1)

From *The Data Lake: Turning Big Data Into Opportunity*
Big Data Primer (1)

- **MapReduce** work in conjunction with a **file system**
- The **data lake** is this file system
  - One metaphor for the data lake might be a giant collection grid, like a **spreadsheet** – one with billions of rows and billions of columns available to hold data
  - Each cell of the grid contains a piece of data
  - Cells might contain names, photographs, incident reports, Twitter feeds, **OAIS submission information packages** – anything and everything
The image of the grid helps describe the difference between **data mining** and the **data lake**.

With the data lake there is definitely an **ingest process**:

- During the ingest process certain details called "**metadata**" are added so that the basic information can be quickly located and identified.
— These metadata tags serve the same purpose as old-style card catalogues which allow a reader to find a book by searching the author, title or subject

• A MapReduce job uses these metadata tags first to retrieve the appropriate data from the data lake

• Next MapReduce maps each piece of information to a processor, assigning the piece of information a key – K1 – one for each processor
Next the processor runs a user-defined program on the many pieces of information it receives.

- During processing it very likely *shreds* each piece of information into other units, producing for each initial key/value pair one or more additional key/value pairs.

- So the map process looks like this: \( \text{map}(k_1, v_1) \rightarrow \text{list}(k_2, v_2) \)

Finally the *reduce() function* is applied to all the lists, aggregating them, and return the value \( v_3 \).
Data Lake MapReduce Pipeline

- Ingest into Data Lake
- MapReduce
- MapReduce retrieves data from data lake
- Map() assigns data to a processor
- Processor processes data
- Reduce() aggregates data across processors
- Metadata tag for retrieval
Big Data Primer References

• The Data Lake: Turning Big Data Into Opportunity

• The Data Lake: Taking Big Data Beyond the Cloud

• NoSQL Data Modeling Techniques

Note that this reference is of interest for those who want to work through the problem of how data structures might be described and annotated using metadata tags in a data lake environment where on the surface the lake consists of just a big bag of key/value pairs