Dagstuhl 2015 Report

Table of Contents

Design Principles: Next Steps

These were reviewed by the Technical Committee and brought to the Scientific Board for support (Spring 2016). We have asked various groups to look at metrics surrounding their particular area of responsibility (for example TC is focusing on metrics regarding what is produced in terms of completeness, response to community interests etc.)

The latest version of this document is at:
https://ddi-alliance.atlassian.net/wiki/display/DDI4/Modeling+Guidelines+for+Business+Modelers?preview=%2F37552132%2F37879817%2FDesign_Principles.pdf

Class Library and Versioning

The Modeling Team have been continuing to discuss the issues raised at the sprint and have used this information to inform an overall versioning approach. This area is still in flux but we are starting with a conservative approach (Versioning at the library level and tracking at what point objects change).

We are still in development review stage and so are still looking at options for production review frequency. This topic and issues of backward compatibility are still under discussion.

Production Workflow

Work has progressed on capturing all steps in the production line (PIM, PSM (flattened XMI for each binding), Transformation tools and rules, bindings, documents, images) and organizing this using Bitbucket and Bamboo.

This is held at https://bitbucket.org/ddi-alliance/ddi-views

Patterns

Discussion of possible patterns within DDI continues. Looking at how these are modularized and how they are realized in the model.

Both the Process and Collection Pattern are coming out in the forthcoming release with the focus on reviewing the relevant patterns involved. Methodology has been put forward as a third pattern for this release.

We have differentiated between the “pattern” content which would require the creation of instantiated classes that realize the pattern and those classes which can be used directly. For example: Sampling Design (realizing a Design) directly using Rationale which has a relation to Design but is not a class within the Methodology Pattern and so can be used directly. There has also been discussion of how to reflect "Planned" vs. "Run time" vs. "Retrospective (what was done)" viewpoints. Having a separate Historical Process was not the answer.

Collections: The forthcoming review will have two Functional Views realizing the Collection Pattern; Agent Registry View providing a very simple collection, and a Functional View of Statistical Classification in order to review its relationship to XKOS (goal would be to duplicate this structure)

Process: The forthcoming review will have a Functional View of a Data Capture Instrument realizing some of the more complex features of the Process Pattern.
Methodology: Pending a final decision the Methodology Pattern may be included in the forthcoming release as a model only. An implementation of this pattern is planned in the special release of a Codebook Functional View in 2016.

A longer version of the discussion is at Appendix 3.

Bindings

These are currently being reviewed and revised with the noted ability to provide relationships for Inputs and Outputs emphasised.

General Binding Rules
RDF is still on hold due to lack of resources

A longer version of the discussion is at Appendix 1.

Use Cases

The three use cases were noted by the Modeling Team and will be useful for providing examples and for testing the ability of Functional Views to fulfill their goals.

They are:

- Longitudinal
- Blood Collection
- Data Linkage

Data Capture

The forthcoming release will have a simple questionnaire Functional View and Dagstuhl 2016 will focus on expanding the coverage of Data Capture in terms of both depth and breadth

Data Description

Viewpoint has been incorporated. Terminology issues were raised at the Norway 2016 Sprint and clarified the need for additional documentation of classes at the level of their use in a Functional View. This would provide the ability to provide terminology related to the community of use for the Functional View and provide input to a cross-community glossary of terms in DDI.

A longer version of the discussion is at Appendix 2.

Appendix 1: General Binding Issues

October 23, 2015

Participants

Michel Dumontier, Michelle Edwards, Martin Forsberg, Alejandra Gonzalez-Beltran, Arofan Gregory, Marcel Hebing, Daniella Meeker, Mary Vardigan, Joachim Wackerow

Goal / question

The goal of this session was to bring together the XML and RDF groups to talk about how the bindings work together with the optimal output.

Documentation & notes

Every object in DDI has an identifier. In RDF, you can have a URL with agency, ID, and version, each of which has a URL. There are four URLS that you need to persist to round-trip from XML and back.

The three fields could be stored as literals. Because the predicates are also URLs, they need to be stored as well. There is also a Local ID to store all of this. The other nuance to this is there are some things that are not full-blown objects but are complex data types. In the RDF, these would become blank nodes.

How will the conversion be achieved technically? Probably in Java, C#, and XSLT. There should be a reference implementation for this. But it is currently questionable whether the DDI Alliance would sponsor this. It is probably more likely that a member like GESIS would do this. Currently the XML goes to OWL but OWL is not a constraint language and so it is not as meaningful as it should be. So SHACL (there is a draft spec at (https://www.w3.org/TR/shacl/) and Shex (see https://www.w3.org/2001/sw/wiki/ShEx), which can validate, should be used to have as much expressivity as the schemas. We would translate the semantics of the XML schema to RDF schema using these other languages.

The bigger issue is the conversion between the XML and RDF instance data. You can do a simple dump of XML into RDF but this will not take advantage of the features of RDF. The treatment of external vocabularies is another issue. We need to know what the expected structure of the transformation is so it's not a naive transformation.
We don't currently have a unit to unit type relationship. Naive: replace the DDI relation with a domain-specific translation. More complex: a set of things is transformed into another set -- this needs rules. But we can't dictate to developers. However, we can say the input looks like this and the output looks like this. You can use RML for these rules or some formal language. There aren't too many cases where these special transformations need to occur. But when there is a meaningful semantic embedded that needs to be represented in RDF.

The model-driven approach doesn't have as much rich information as what the data description group has discussed. Right now the XML is richer.

An example would be helpful. We are responsible for specifying the schemas in RDF and XML and provide guidance to people so they can do the transformation. That guidance should be computable. Every W3C group has the reference implementations. They don't produce a spec without an implementation. This is strongly recommended for DDI. And DDI has a proof of concept requirement before voting on a new spec takes place.

For each vocabulary you target you could have a different structure.

Simplifying the Model

There will be local and global concepts. Things local to a single dataset can stay here. It doesn't have to be in the model.

How do we produce an RDF vocabulary that is much smaller and easy to use?

The model may have too many classes. Good design patterns are key core things and subtypes that are specializations of that. It's a same structural pattern but maybe a domain of different applications. The integration step of modeling has not been done yet.

Michel has SIO standards that have the minimal number of elements to be maximally expressive. How do I treat space and time regardless of what we are measuring?

An empirical evaluation of the highly used concepts would be very helpful.

An example of an RDF document built from an XML schema was produced using turtle. This surfaced some of the verbosity in DDI. SIO (Semanticscience Integrated Ontology) would do it slightly differently.

We should create a repository of everyone's DDI and then understand which elements everyone is using. We also need to do some model integration, which will simplify the model.
Flattening has an effect on the XML and inheritance means we are carrying a lot of information along. From an XML perspective, it would be cleaner to separate the individual from the administrative information about the metadata.

An element that says "administrative" would separate these things.
How we could use graph statements to capture versioning. We can have a record for an individual and the individual is version-less. The trig graph format shows the graph name as the record identifier. "isPrimaryTopicOf" is included. We should look at the model to make sure administrative information is separated.

All objects should be identifiable. In Linked Data, we want to identify EVERYTHING. If you have unaddressable things, there are problems. We could show a view without all of the identifiers. How can we reliably round-trip identifiers across instances? Practice for LD is to provide a mechanism to resolve the IDs. We can keep URNs and then publish a URI that is resolvable. We have a mechanism for storing multiple IDs. Do you have a commitment to provide the resolution for the long term? If we have the capacity, we should do it. We can make assertions based on the URN but when you want to get them, use the URL. This would allow us to get rid of things like AnnotatedIdentifiables.

Identification in RDF

There is an OWL pass key for identification. The challenging thing and the tension is do you want information to be merged (about, say, an individual) to be exposed?

Conclusion

An empirical evaluation of the highly used concepts would be very helpful.

Currently the XML goes to OWL but OWL is not a constraint language and so it is not as meaningful as it should be. So SHACL and Shex, which can validate, should be used to have as much expressivity as the schemas. We would translate the semantics of the XML schema to RDF schema using these other languages.

The bigger issue is the conversion between the XML and RDF instance data. You can do a simple dump of XML into RDF but this will not take advantage of the features of RDF. The treatment of external vocabularies is another issue. We need to know what the expected structure of the transformation is so it’s not a naive transformation. You can use RML for these rules or some formal language.

All objects should be identifiable. In Linked Data, we want to identify EVERYTHING. If you have unaddressable things, there are problems. We could show a view without all of the identifiers. How can we reliably round-trip identifiers across instances? Practice for Linked Data is to provide a mechanism to resolve the IDs. We can keep URNs and then publish a URI that is resolvable. We have a mechanism for storing multiple IDs. Do you have a commitment to provide the resolution for the long term? If we have the capacity, we should do it. We can make assertions based on the URN but when you want to get them, use the URL. This would allow us to get rid of things like AnnotatedIdentifiable.

Appendix 2 - Data Description Summary

Participants

David Barraclough, Gary Berg-Cross, Michel Dumontier, Martin Forsberg, Dan Gillman, Alejandra Gonzalez-Beltran, Larry Hoyle, Jon Johnson, Daniella Meeker, Steve McEachern, Eric Prud'hommeaux, Barry Radler, Ørnulf Risnes, Flavio Rizzolo, Dan Smith, Achim Wackerow

Goal / question

A model for data description from the datum up to the data store

Documentation & notes

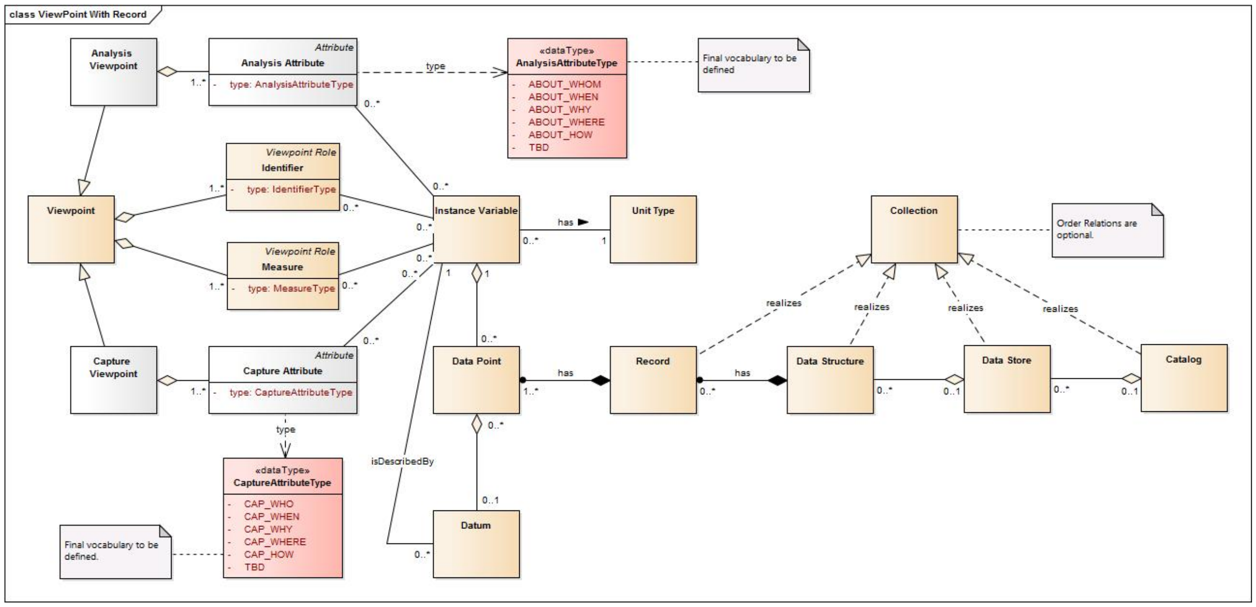
see Data Description Discussion

https://docs.google.com/document/d/1NxelsigRMpMAI73Pes2vrpwrpn4IyMI7CyRA0FbARvM/edit#heading=h.hmttvl1zpv3i

Conclusion - Agreed positions

The group, after much discussion, agreed upon the following model which is being implemented in the Drupal site.

Model Diagram



Definitions

A Viewpoint is an assignment of the Roles Identifier, Attribute, and Measure to DataPoints in a Record. In a ViewPoint each DataPoint has a single Role. This seems to correspond to a Data Structure Definition in SDMX.

- An Identifier is a Role of a Datapoint used to identify a Record under some ViewPoint.
- An Attribute is a Role of a Datapoint describing other characteristics of a Record under some ViewPoint.
- A Measure is a Role of a Datapoint containing a Datum of interest under some ViewPoint.
- A Role designates the function an InstanceVariable performs in the context of the ViewPoint. (Identifier, Attribute, or Measure of interest).
- A DataPoint is a container for a Datum.
- A Datum is the designation of a value.
- A Value is a concept with a notion of equality defined.
- A Record is a Collection of DataPoints with an optional OrderRelation.
- A DataStructure is a Collection of Records with an optional OrderRelation.
- A DataStore is a Collection of DataStructures.

An overview documentation describing the above model and it’s application to a case of the storage of a blood pressure capture is linked HERE.
The attributes that that are used to describe the association of a DataPoint with a Viewpoint were inspired by the W5H framework. These are as follows.

The W5H Questions

- What references a Variable that describes the Value – Its concept, value domain etc. When data atoms are arranged into a table this would relate to a column of the table.
- Who contains a reference to the Unit associated with the value. When data atoms are arranged into a table this relates to the rows of the table.
- Value contains the representation of the value of the data atom.
- Where points to spatial information related to the value.
- Why relates to the purpose of the value
- When points to temporal information related to the value
- How describes the manner in which the value came into being – was it via observation or a transformation of other data?
- Annotation. The data atom may also have a reference to an annotation.

Non agreed positions

What is the exact nature of a DataStore and how will users interact with it? A review of the possible functionality of the DataStore was developed by Ørnulf Risnes and is linked here for consideration (see HERE).

There was discussion about whether the Datum should be a separate class. Not all participants had an opportunity to fully discuss this. A new issue (DMT-14) has been added to the Modelling Team's issue list on JIRA for consideration.

There was discussion about whether a Datum can be versioned. Not all participants had an opportunity to fully discuss this. A new issue (DMT-15) has been added to the Modelling Team's issue list on JIRA for consideration.

There was some discussion of the relationship between DataCapture and DataDescription, particularly the set of objects that would required to execute a DataCapture and store a Datum.

Flavio noted that this would likely include the following:

- InstanceVariable
- DataPoint
- Datum
- Record

However, this has not yet been evaluated through the completion of an example. This exercise should be conducted using one of the established UseCases as soon as possible.
There was no opportunity for a review of the final version of the model with regard to the implications for the expression of the model in either RDF or XML. This exercise should be completed as part of the review of the package.

Recommendations

- Implement these model changes in the DDI4 model (in Drupal).
- Test the model against several use cases. Ideally one of the use cases developed at this sprint, e.g. the data linkage use case.
- Describe specific examples of populating a Rectangular Structure and a Cube. (See initial discussions of this in the Thursday afternoon section of the Data Description Discussion document.

Still open / further discussion

The use of the Process model and Methodology to capture a Datum and populate a DataPoint has not been assessed. The DataCapture and DataDescription group should complete this work following the workshop.

One possible “W” attribute is “whether” inclusion of access control attributes could allow control down to the Datum level. This might be useful in a DataLake or other situations where data access needs to be conditional.

There was some discussion of the need to capture the provenance of both data and metadata in the model. Some discussion of both the PROV W3C standard and the W5H model was considered throughout the week, and there was some consideration of how this might be achieved through the use of the Viewpoint object. However this discussion was only preliminary only and needs to be further considered with respect to both the relevant provenance vocabularies/standards/etc. and the implementation in the model.

Appendix 3 - Patterns

20 October 2015 | session 1 & 2

Participants

Gary Berg-Cross, Arofan Gregory, Marcel Hebing, Eric Prud'hommeaux, Flavio Rizzolo, Wendy Thomas, and Joachim Wackerow

Goal

1. Review of the process and collection model patterns, with the idea that they could be more modular
2. Rationalization and formalization of the model and its relationships

Documentation & notes

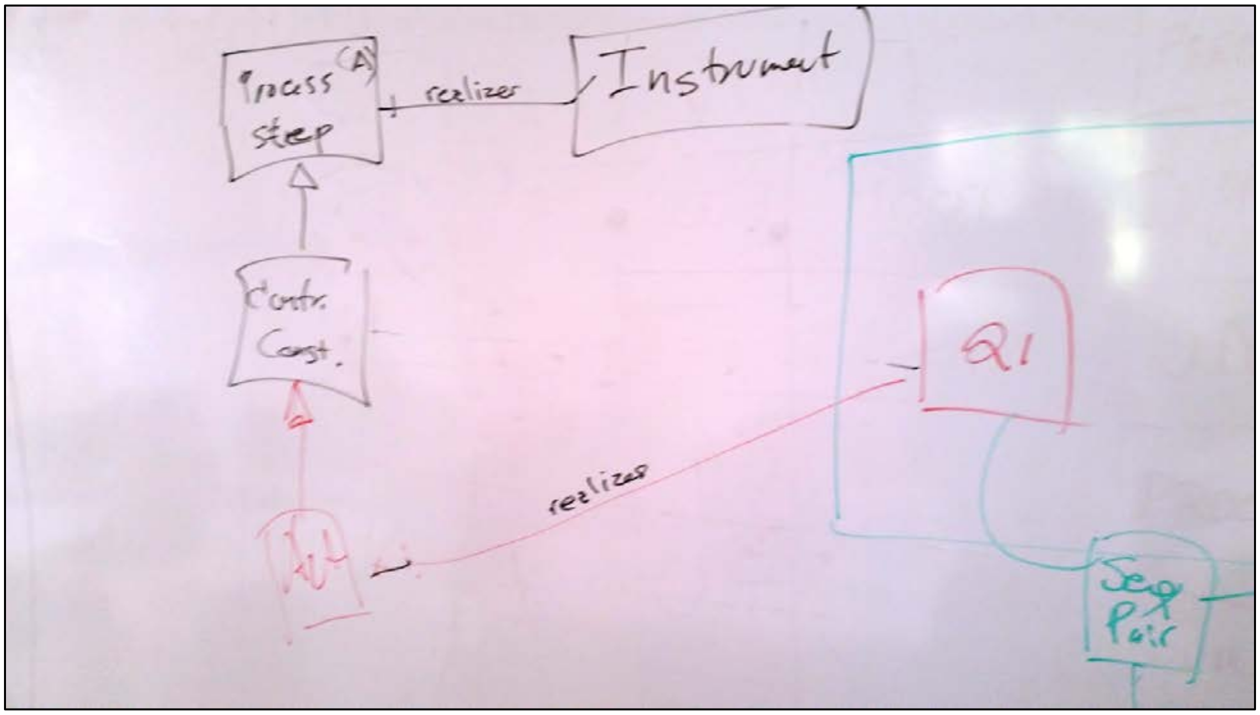
Relevant links and documents

- Modelling team on the DDI wiki
- piratepad.net/sio-dagstuhl
- Models (PDF): Process and Binding
- Act, Service (in Drupal)
- BPMN: example, Wikipedia

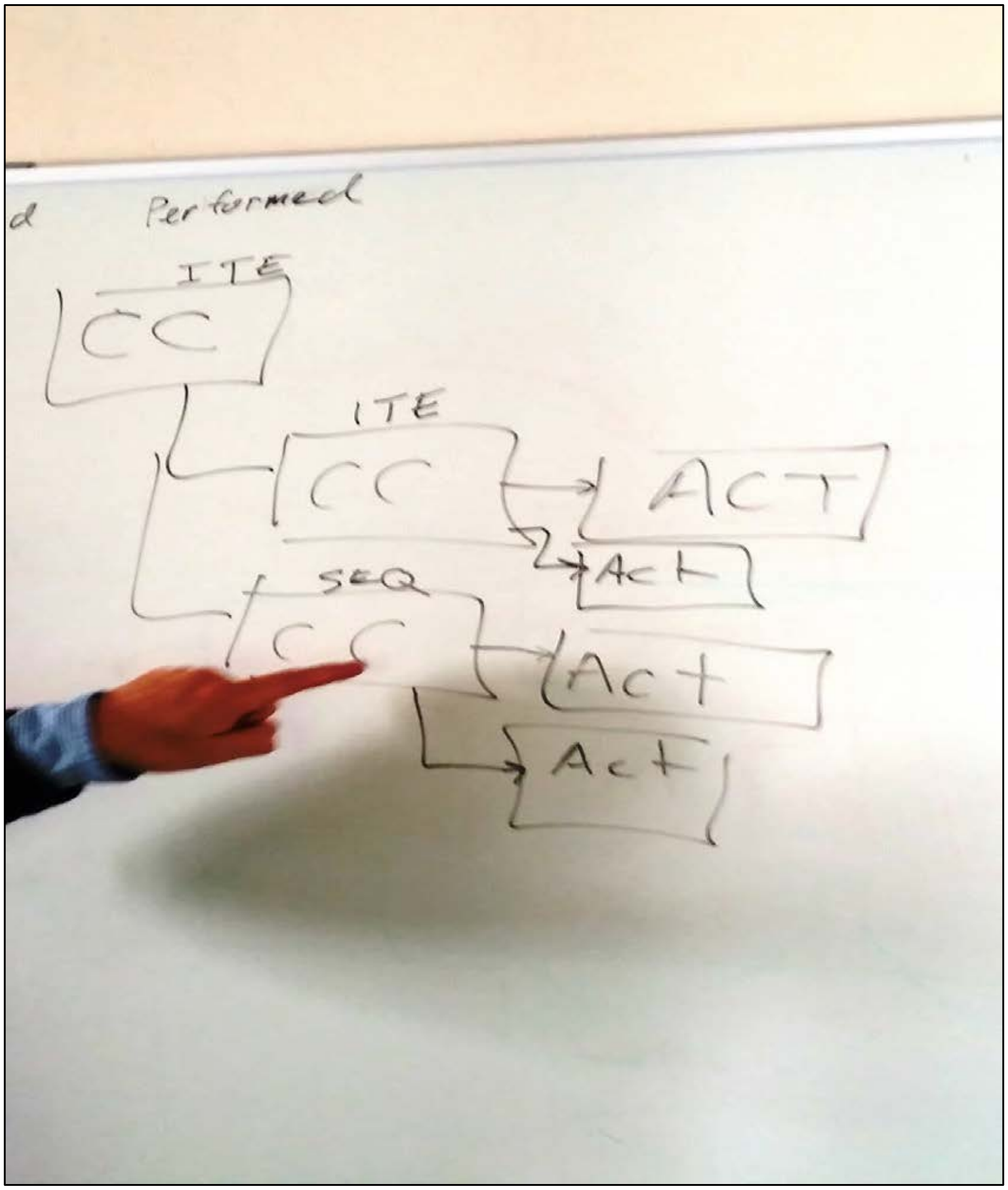
Quick notes

1. Our "realizes"-relationship is itself a pattern. The discussion takes an "Instrument" as an example, which realizes a "Process Step".

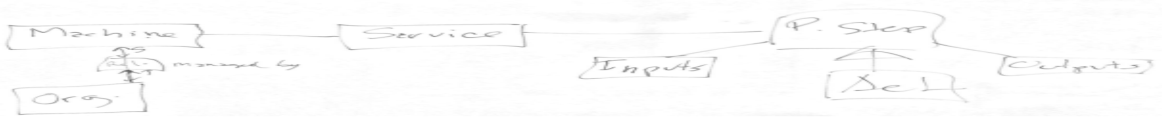
2. For future discussion: Is "Act" an abstract object or something to implement? – First conclusion: it probably should be abstract.
3. Eric proposes leveraging ideas from a clinical trials model as useful here with Instrument: defined → planned → scheduled → performed
    a. Performing a survey is a different type of step from the others. We capture different types of metadata for this, such as how the Instrument was carried out, conformance etc.
    b. Note there are loops back, so that an Instrument is tested and we go back to improvement and a new plan.

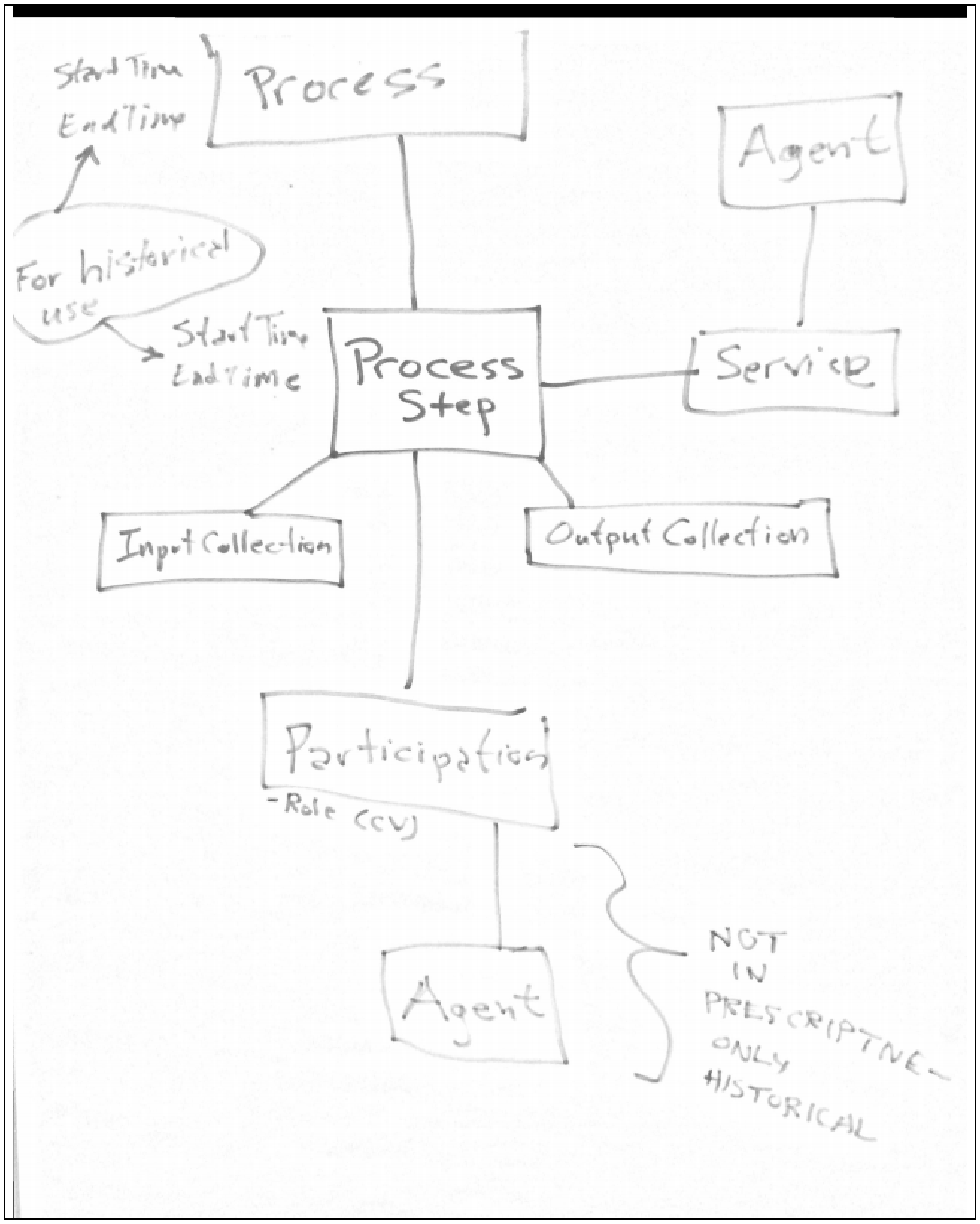


1. We might need to re-model "Act" to contain "Control Constructs"—to get to a level of detail that gets real.



1. In reality, you might start with an "Act" (e.g., an R script) and later on replace it by a "Control Construct" to add more detail. But is it a good idea to replace one Class by another (and losing identifiers)?
2. Do we need a role in addition to a Service to prescriptively describe processes?


Machine
managed by
Org.
Service
P. Step
Inputs
Del.
Outputs



1. We should add StartTime and EndTime to Process and ProcessStep (for historical use).
2. Add design time. InputCollections and OutputCollections are Types of objects. In the historical description of a process, the InputCollection and OutputCollection are instances.
3. A question is, “What types of ontologies/ODPs might exist that we can leverage?”

a. There are things in DOLCE and its derived patterns that may be leveraged. These include the Participation pattern mentioned in Gary's presentation, but also the Path ODP that has been used in several areas and is well formalized. It may be used to describe the related segments in a sequence. See also GeoLink Core Ontology Design Patterns

b. Process ontologies also exist, such as the formalization of PSL.

c. PROV O is another candidate as is Semanticscience Integrated Ontology SIO mentioned by Michel.

d. Also some use of the HL7 RIM may be of value.

Conclusion

Process: Several issues were identified regarding the process pattern. These pertained to the way in which roles were related to process steps, and how time was related to the process model. It was suggested that we look at the "path" (semantic trajectory) pattern from VOCAMPS to add more rigour to our modelling. PROV-O might need to be revisited.

Bindings: Bindings were discussed, with an explanation of the collections pattern within the process model. Again, the VOCAMPS path pattern might be usefully applied.

Collections: We also looked at collections, but did not identify any major issues. We confirmed the value of XKOS.

Further links:

- VOCAMP (vocamp.org)
- PROV-O (w3.org)

Still open / further discussion

- There are still open questions regarding granularity.
- Should we add Participation/Role to the process model?
- We still need to create the binding relationships for Inputs and Outputs to refer to Types and Instances of data/metadata objects.

- Do we build a single pattern for both prescriptive and historical process instances? How do we manage time in relation to process?

3.3.2 Alignment with Agent Role pattern

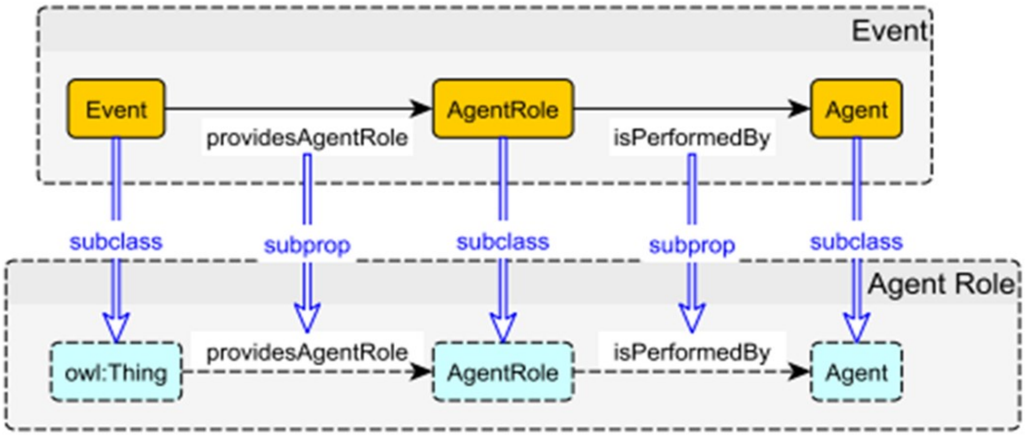


Figure 3.3: Event aligned to Agent Role