Introduction

The enterprise architecture framework (EA) is *"a discipline for proactively and holistically leading enterprise responses to disruptive forces by identifying and analyzing the execution of change toward desired business vision and outcomes. EA delivers value by presenting business and IT leaders with signature-ready recommendations for adjusting policies and projects to achieve target business outcomes that capitalize on relevant business disruptions. EA is used to steer decision making toward the evolution of the future state architecture."*[1]

EA had its origins in IBM's Business System Planning in the 1980s and was codified by NIST in 1989:

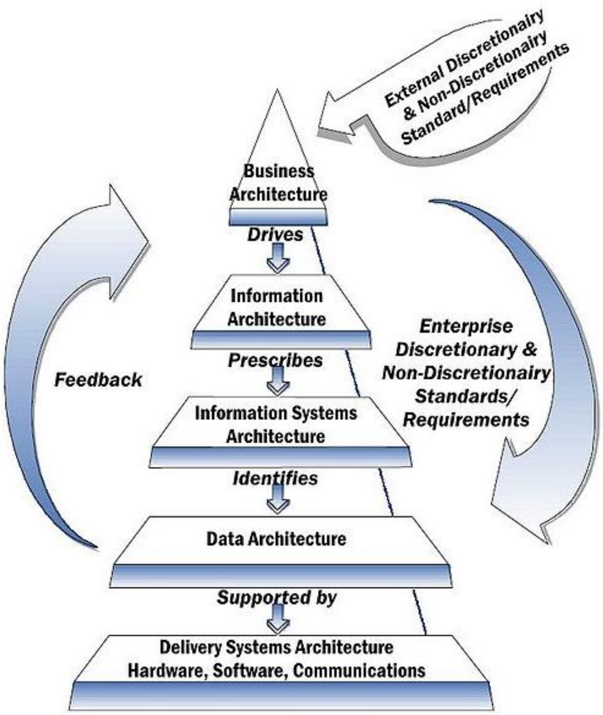


Figure 1: NIST Enterprise Architecture Framework

Since then, EA has evolved but some of its layers remain the same.

EA advocates strong separation of concern and strict decoupling between four architectural layers[2]:

- Business architecture (BA) *"covers all the activities undertaken by a statistical organization, including those undertaken to conceptualize, design, build and*

[1] Gartner IT Glossary, http://www.gartner.com/it-glossary/enterprise-architecture-ea/

[2] Definitions from the Common Statistical Production Architecture (CSPA) glossary, http://www1.unece.org/stat/platform/display/CSPA/Annex+3_+Glossary

maintain information and application assets used in the production of statistical outputs. BA drives the Information, Application and Technology architectures for a statistical organization.”

- Information Architecture (IA) *“classifies the information and knowledge assets gathered, produced and used within the Business Architecture. It also describes the information standards and frameworks that underpin the statistical information. IA facilitates discoverability and accessibility, leading to greater reuse and sharing.”*

- Application Architecture (AA) *“classifies and hosts the individual applications describing their deployment, interactions, and relationships with the business processes of the organization (e.g. estimation, editing and seasonal adjustment tools, etc.). AA facilitates discoverability and accessibility, leading to greater reuse and sharing. Source: Statistical Network BA definition”*.

- Technology architecture (TA) *“describes the IT infrastructure required to support the deployment of applications and IT services, including hardware, middleware, networks, platforms, etc.”*

Furthermore Enterprise information management (EIM) is *defined as “an integrative discipline for describing, organizing, integrating, sharing and governing information assets across organizational and technological boundaries. Its goal is improving business objectives (through increased effectiveness and efficiency), promoting transparency and enabling business insight. Enterprise information is the most valuable information to the business, which is used across business processes and organizational units. In effect, EIM serves to operationalize key principles of enterprise information architecture, elevating enterprise information to the position of a strategic asset that is effectively controlled, leveraged and optimized for significant business value.”*[3]

The DDI standard covers information architecture through its information models but also covers to some extent business architecture via its business model. Information standards such as DDI can be seen as key enablers for EIM.

The DDI Business Model

DDI Lifecycle entails a number of activities within and between the successive stages of the data lifecycle. These activities form pipelines. Recently these pipelines were mapped to Lifecycle and together they form the DDI Generic Longitudinal Business Process Model (GLBPM). The map has two views. Here is the standard view:

[3] See Gartner research note “Hype Cycle for Enterprise Information Management”, 2012, http://www.reassent.com/Portals/0/gartner-hype-cycle-for-eim.pdf

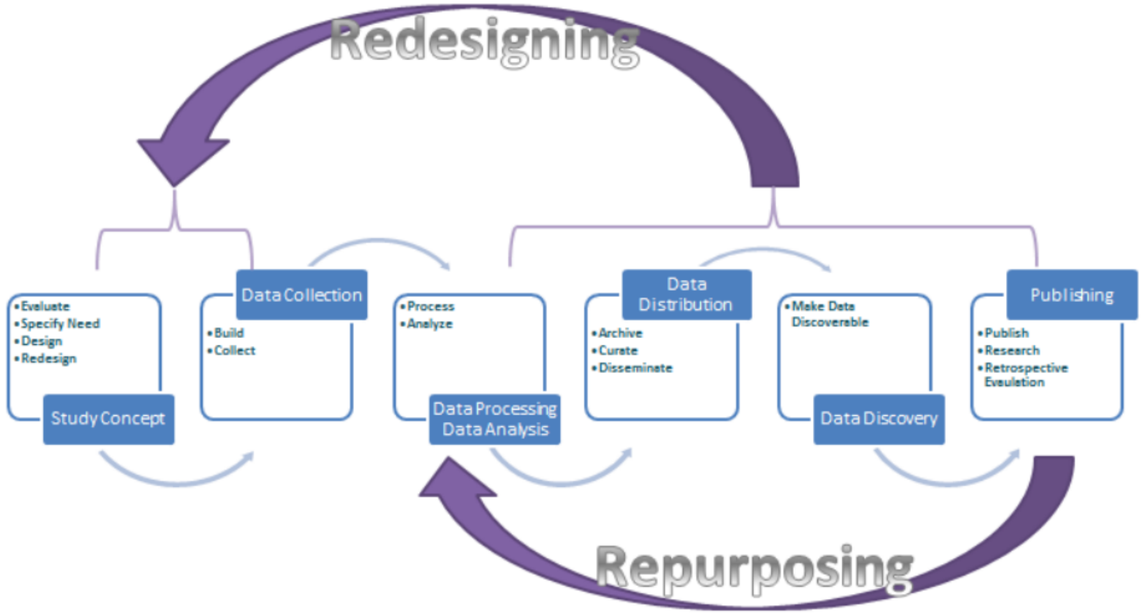


Figure 2: Mapping of GLBPM on DDi Lifecycle - Standard View

Here is the so-called "circle view":

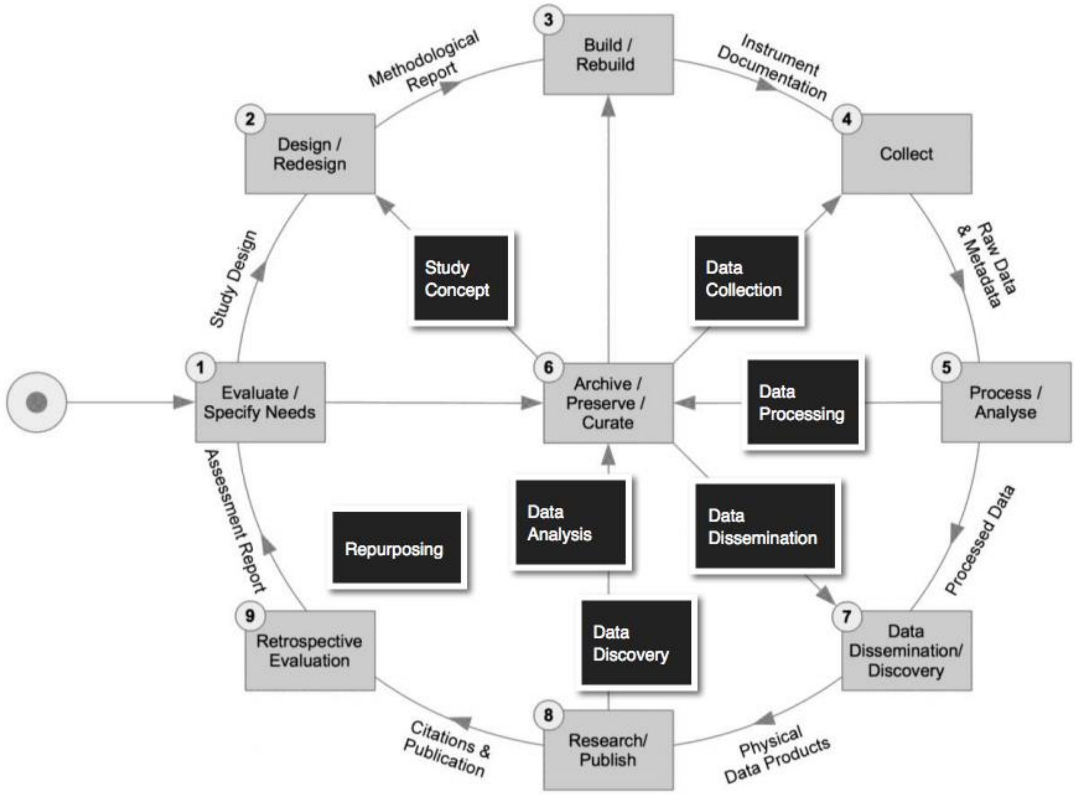


Figure 3: Mapping of GLBPM on DDI Lifecycle - Circle View

GLBPM is the DDI business model and it contains a number of process type pipelines[4]. Here are just some of the process type pipelines that can be found in GLBPM:

- Instrument scheduling including "case management"
- Instrument execution including questionnaire navigation
- The production of official statistics
- Data processing including the production of SIPs, AIPs and DIPs in provenance chains[5]
- The production of business activity monitoring (BAM) metrics to facilitate both near real time and delayed retrospective evaluation

Process type pipelines are abstract and have subtypes. Sometimes we refer to these subtypes as "design patterns". For example, computerized adaptive testing is a design pattern and a subtype of instrument execution.

Process type pipeline subtypes and their design patterns can be both very specialized and core in certain lines of business. For example, recently SAS has identified two core pipelines for data processing – Data Federation and Traditional ETL/ELT. Here SAS offers three use case for Data Federation as opposed to constructing and maintaining a single source of truth:

- 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.

[4] The concept of a process type pipeline was first introduced by IBM in the context of its Sterling Selling and Fulfillment Foundation: "A process type pipeline is a series of transactions and statuses that guide document types, such as a Sales Order, through a predefined process. A pipeline consists of the different statuses a document goes through during fulfillment, negotiation, shipment, or receipt. You can also set up transactions consisting of events, actions, and conditions, as they pertain to the pipeline you are configuring."

[5] SIPs, AIPs and DIPs refer to "moments" in the Open Archival Information System (OAIS) reference model.

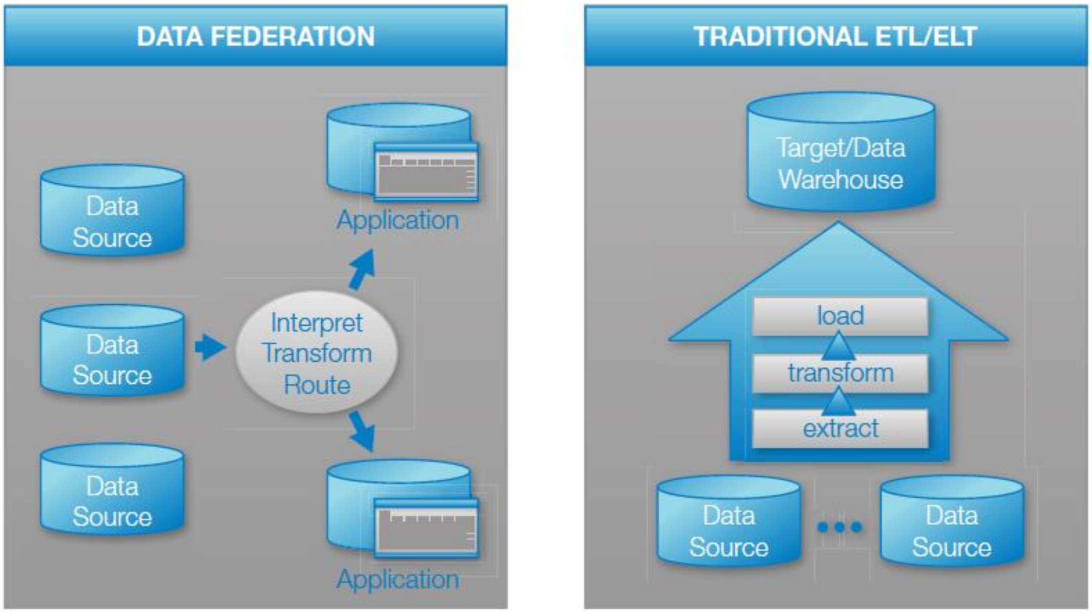


Figure 4: SAS Core Data Processing Pipeline Types

Note that the business case GLBPM makes is broad. It includes many process types. And these process types can be extended into information models that are leading edge in a given business process area. Also, it needs be noted here that GLBPM was based on and extends the General Statistical Business Process Model (GSBPM) with its circle view and additional process pipelines.

By providing building blocks usable in the context of business process modeling GLBPM can be seen as a business architecture standard.

DDI's Process Information Model

DDI has a general purpose lightweight processing model that it has grown in the course of several versions of the DDI specification. Currently, this model is embedded in DDI's ControlConstructScheme where it is used to describe the skip logic that is core to one subtype of instrument execution.

DDI borrowed its control constructs and process model from OWL-S. OWL-S is an ontology built on top of Web Ontology Language (OWL) by the DARPA DAML program. It replaces the former DAML-S ontology. OWL-S has several parts. The part of OWL-S that DDI borrowed and is applicable here is the OWL-S process model.

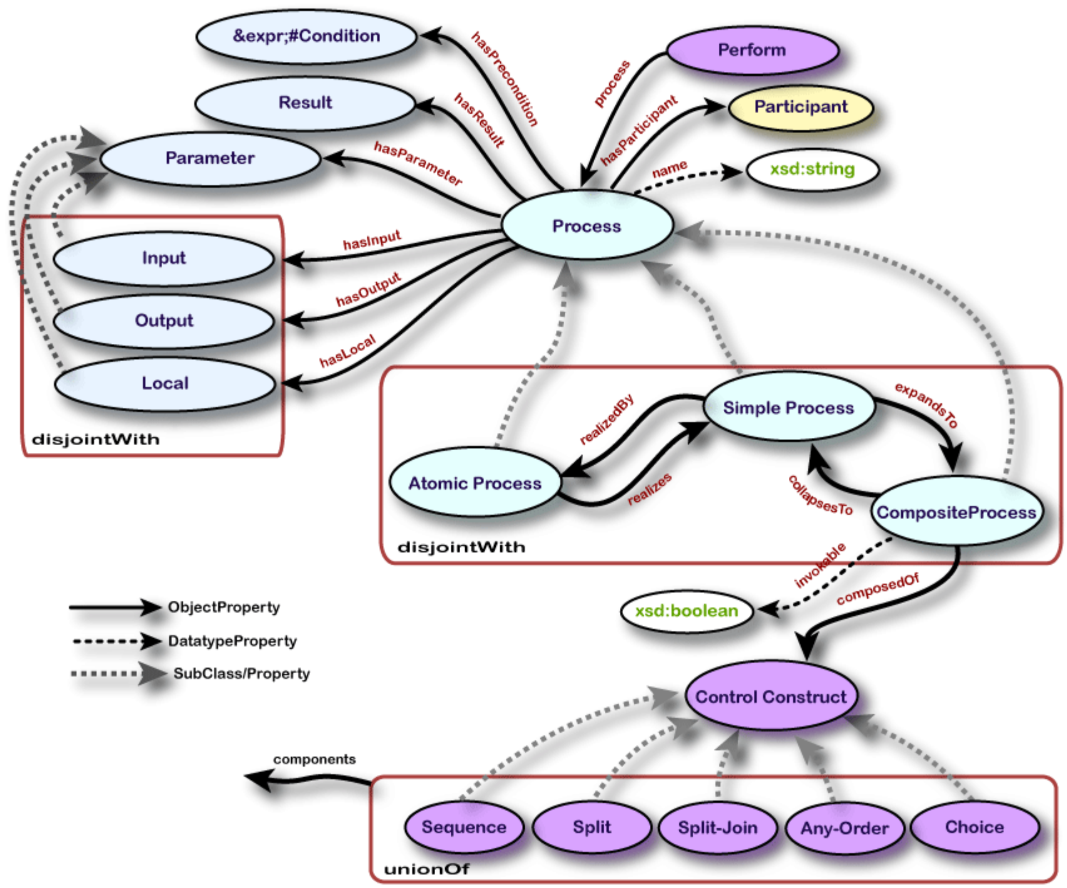


Figure 5: OWL-S Process Model

The DDI business process model borrowed from OWL-S works well with at least certain types of data collection instruments. It remain to demonstrate that it is generic and, as such, capable of modeling all the types of process pipelines enumerated before.

OWL-S, GSIM and Ray Kurzweil Process Pipeline Types

In OWL-S first there are disjoint atomic, simple and composite process types that can over time morph into one another. In this context we consider a "corner case" which perhaps can reveal the capabilities of OWL-S and, by way of comparison those of the Generic Statistical Information Model (GSIM).

Let's call the corner case the "quantum questionnaire": given an input, it is able to score a participant on a construct using the answer to just one question. Let's assume that this score has predictive validity. In OWL-S the quantum questionnaire would be represented by a "simple process" which for now isn't "knowable", so it cannot be represented as a "composite process". Instead, for now, it is represented and realized by the OWL-S "atomic process". Indeed, this type of process may never be knowable in which case it may be an example of a "singularity" as defined by von

Neumann, Kurzweil and others. Recall that a technological singularity is a computer-based intelligence that knows how to solve problems that humans can't.

Once we entertain the possibility of a type of process that is predictable, useful but unknowable, control constructs don't apply. Under these circumstances a process can be performed by a black box that is, of course, "machine readable" but not "human readable".

Sometimes legacy programs that produce official statistics are "magical", but we want to replace this magic with a knowable, "human readable", repeatable process. Sometimes we build automata that can change their own operating instructions, improving themselves in the process. Here we don't dispel magic but are in the business of producing it.

It appears that GSIM can represent these situations much as OWL-S can:

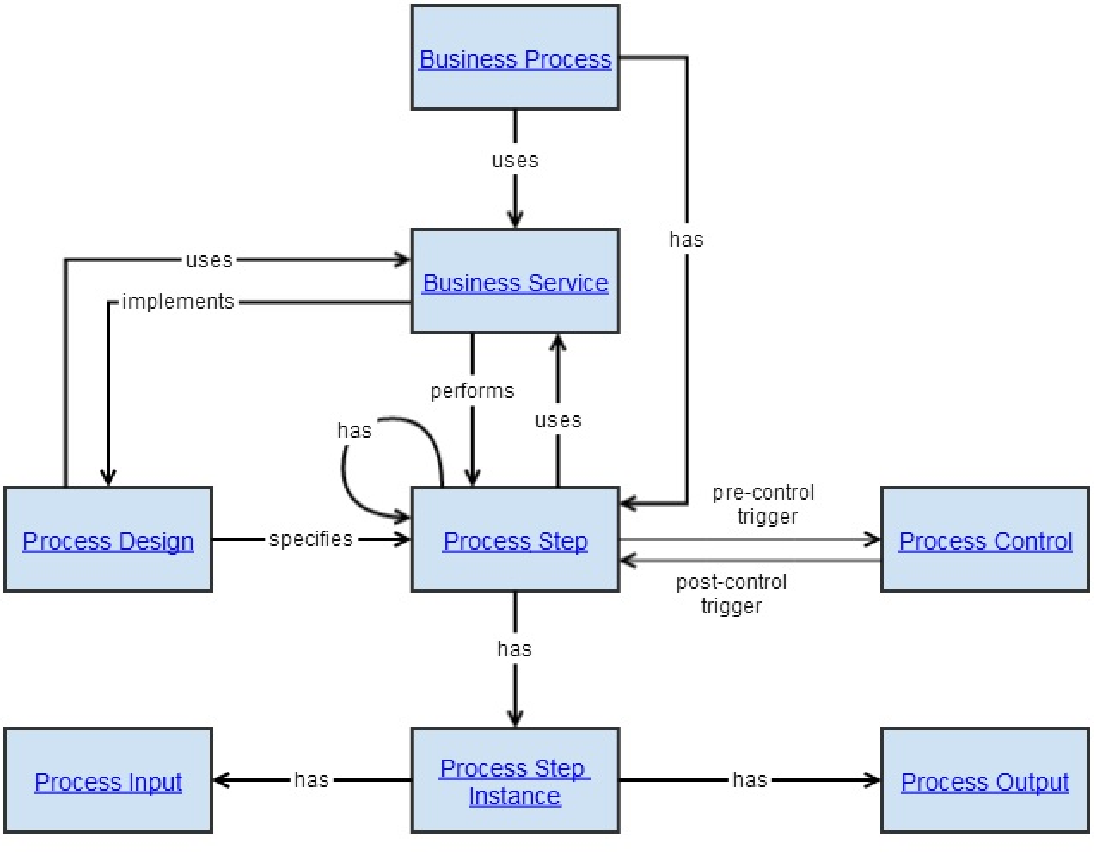


Figure 6: GSIM Process Objects

GSIM has a Process Design process object where, in principle, composite processes, atomic processes, "simple" processes and singularities can be specified. Here note that "the *Process Design* defines and the *Process Control* manages the flow between

Process Steps, even where the flow is 'trivial'. *Process Design* is left to focus entirely on the design of the process itself, not sequencing between steps"[6].

One way of thinking about the relationship between ~~DDI/OWL-S and GSIM~~GSIM, OWL-S and DDI is that GSIM is more general and then OWL-S together with DDI are~~DDI/OWL-S is~~ more specific. Both GSIM and OWL-S/DDI are platform independent information models only GSIM is a conceptual information model or, again a *reference model* and ~~DDI/~~OWL-S coupled with DDI is more domain-specific.

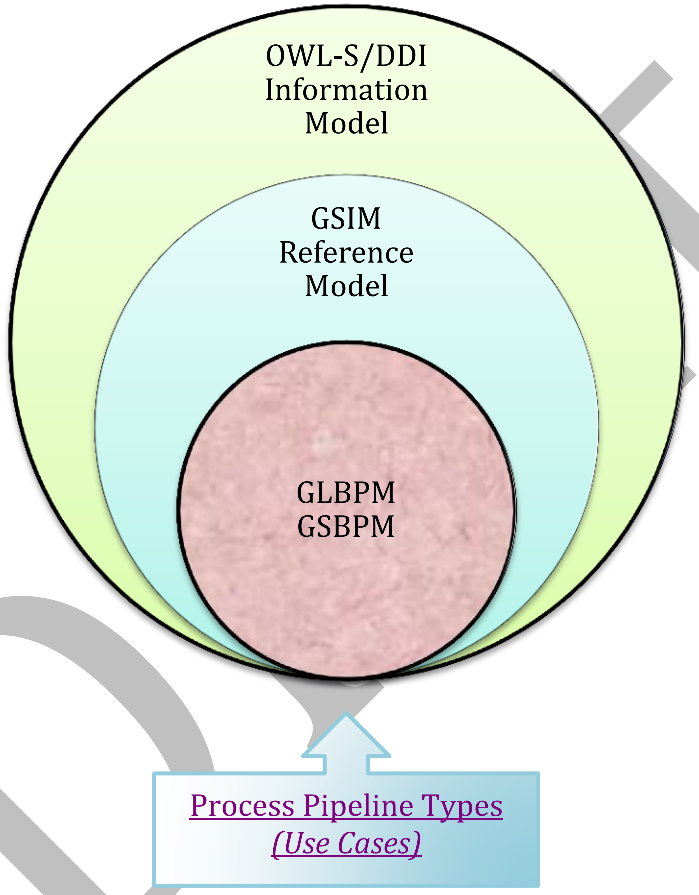


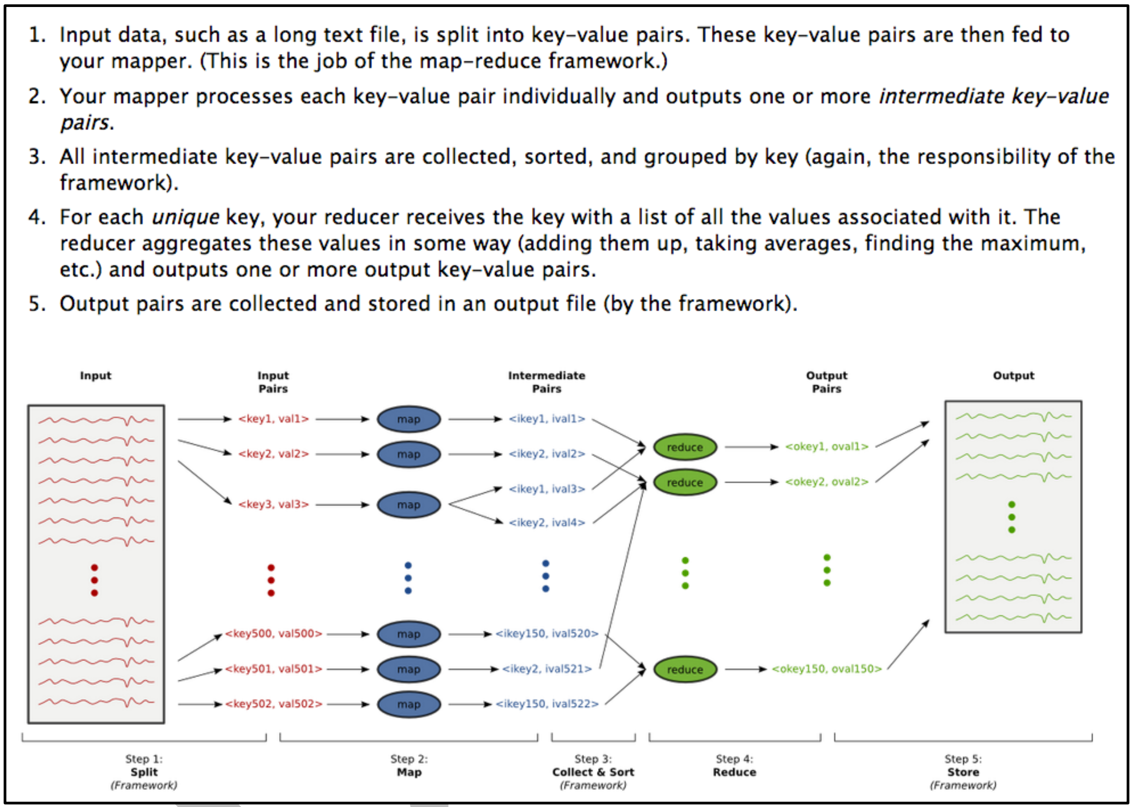
Figure 7: The GSIM/~~DDI/~~OWL-S/DDI Information Model Architecture

In Figure 7 GSIM and OWL-S/DDI together form an Information Model architecture driven by two business process models – GSBPM and GLBPM – which, in turn, are based on several use cases.

What remains to be determined is the adequacy of GSIM/~~DDI/~~OWL-S/DDI ~~vis a vis~~with respect to the process pipeline types (use cases) enumerated earlier. For example, is it possible to model a Process Control that orchestrates a set of control

[6] Personal communication from Therese Lalor.

constructs through which computerized adaptive testing can be modeled? Likewise can we model the SAS data federation data processing pipeline in which data sources are joined just in time and in response to an ad hoc request? Likewise, can we transform a micro dataset that in one way or another integrates several data sources into a macro dataset and the production of official statistics? Likewise, looking ahead now, can GSIM/~~a DDI/~~OWL-S/DDI orchestrate the several MapReduce design patterns? The~~is~~ one in Figure 8 supports the numerical summarization family:



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).

Figure 8: MapReduce Programming Model – Numerical Summarization Design Pattern[7]

One approach to answering these questions is in the abstract. Can it be claimed the DDI/OWL-S is *compositionally complete*? Consider that a process and process steps can be modeled as business services just like GSIM suggests, that BPEL4WS is a "very expressive" and widely used business process development language, that BPEL4WS can be mapped to OWL-S and that in this mapping BPEL4WS was found to be a subset of OWL-S[8]. While it is not entirely provable, under these circumstances OWL-S is arguably compositionally complete.[9]

[7] From WebMapReduce 1.0.2 by Garrity and Yates.

[8] See *From BPEL4WS Process Model to Full OWL-S Ontology* by Aslam, et al.

[9] Many caveats apply here. Perhaps the one that looms largest is the future of parallelism. In addition to "simple parallelism" which OWL-S and BPEL4WS both support with the split/join control construct, there are other forms of parallelism in the pipeline. In "adaptive data parallelism", workflow is dynamically adapted to the current state of the grid execution

DDI Events

Apart from its process model DDI also represents a type of event called the LifecycleEvent. In DDI and, more generally, in ontology processes and events are different[10].

Processes occur. They may be durative or punctual. However, once they occur and pass into history, apart from outputs and outcomes, they produce new information not in connection with the thing that has been processed but about the process itself. General speaking and in the case of the DDI LifecycleEvent, an event captures in large part *dynamic* information about a process:

- When did the process occur?
- Who initiated the process and under what circumstances?
- How long did the process take?
- Apart from its outcome, what was the quality of the process? How practical was the process? How acceptable was it?[11]
- Other key performance indicators (KPI).

Practically speaking, events often convey information to business activity monitors (BAM). Also, practically speaking, when a BAM is getting real time or near real time information, it may figure into and be used by Process Control. When this occurs, Business Activity Monitoring is a subtype of Process Control.

It needs to be noted that the distinction between processes and events in DDI is not always so clear. Consider the DDI ProcessingEvent and ProcessingInstruction. Arguably these objects are about data processing and can be subsumed by one or more of the various GSIM process objects and/or the OWL-S representation of process specifics.

The GSIM/~~DDI/~~OWL-S/DDI Information Model

Introduction

The model presented here has several features:

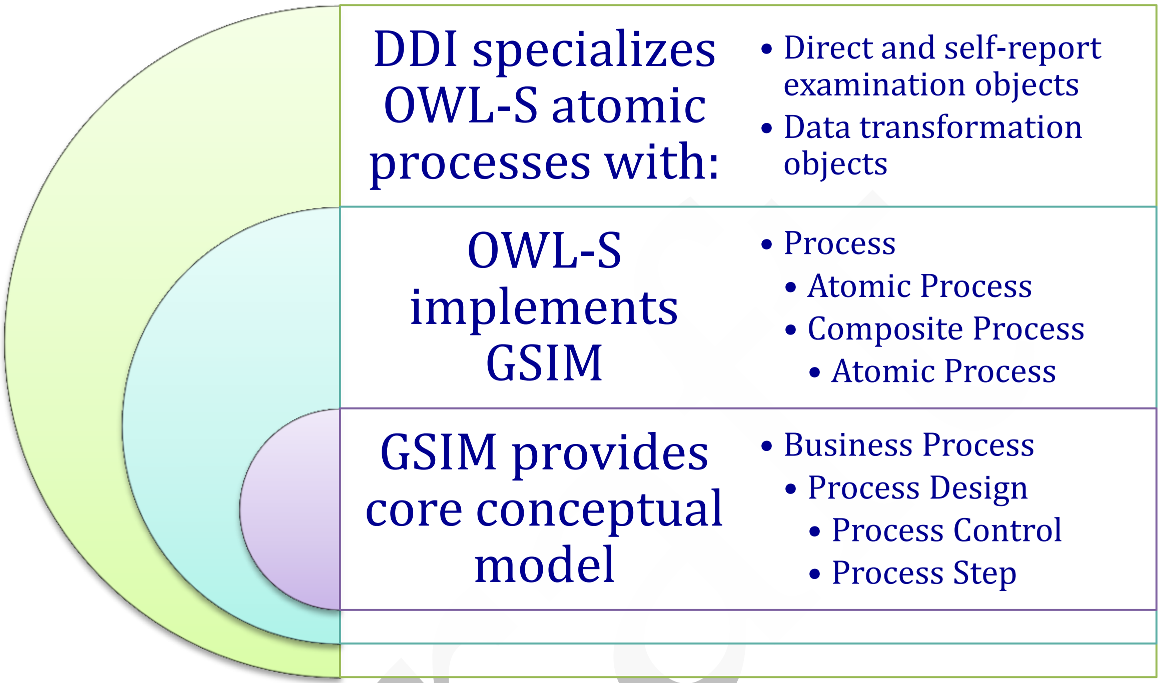
- ~~DDI/~~OWL-S/DDI information objects *implement* GSIM information objects.

environment. In adaptive data parallelism, execution entails some form of *reflection* on grid resources. And this reflection needs to be modeled together with the parallelism. See *Parallel Computing Patterns for Grid Workflows*.

[10] Generally speaking, ontologists agree that a distinction needs to be maintained between things like processes and information about things. In line with this distinction see *On What Goes On: An Ontology of Processes and Events* by Antony Galton.

[11] In DDI 3.x QualityStatement is a reusable object with links to many study objects including ProcessingEvent. See the quality statements in Quality Descriptions by way of example.

- OWL-S composite processes decompose into other composite and non-composite (including atomic) processes
- DDI lends these atomic processes domain specificity drawing from the many process objects sprinkled throughout DDI 3.x.



DDI specializes OWL-S atomic processes with:

- Direct and self-report examination objects
- Data transformation objects

OWL-S implements GSIM

- Process
  - Atomic Process
  - Composite Process
    - Atomic Process

GSIM provides core conceptual model

- Business Process
  - Process Design
    - Process Control
    - Process Step

Figure 9: Introduction to the GSIM/DDI/OWL-S Information Model

The model spreads over many figures:

- Figure 10 features GSIM and OWL-S. In Figure 10 OWL-S implements GSIM.
- Figure 11 presents the OWL-S Composite Process Control Constructs in detail.
- Figure 12 presents the DDI specializations of the OWL-S atomic process. It organizes these specializations using concepts from the Ontology of Biomedical Investigations.
- Figure 13 features a single DDI specialization – the ComputationItem. ComputationItem is a DDI 3.x item that was added to OWL-S to hold a variable and the code executed that gives the variable its value. It has been extended here so that, in addition to code, the variable can take an expression. In the future there might be an expression language like the proposed SDMX EXL expression language that integrates with GSIM and DDI.[12] For now though we have extended ComputationItem to include a simple model-based algebraic expression language.

[12] EXL is the "EXpression Langue" developed and used by Bank of Italy. The SDMX Technical Working Group (TWG) includes members from both the DDI and GSIM standard groups.

In viewing the model, note the following:

- Namespaces are used to distinguish the source of each process object. In this connection OWL-S objects that DDI over time has borrowed get the "owls" namespace. Also, with DDI-specific objects for the most part, no version is indicated, and the namespace is "ddi". There is one exception where a new DDI-specific process object has been proposed – "ddi4". Finally, Ontology of Biomedical Investigations objects introduced to organize the DDI specializations of OWL-S get the "obi" namespace.
- UML class models as a rule don't cross pages. Here for convenience we added page connectors like those used in a flow diagram that spans multiple pages.

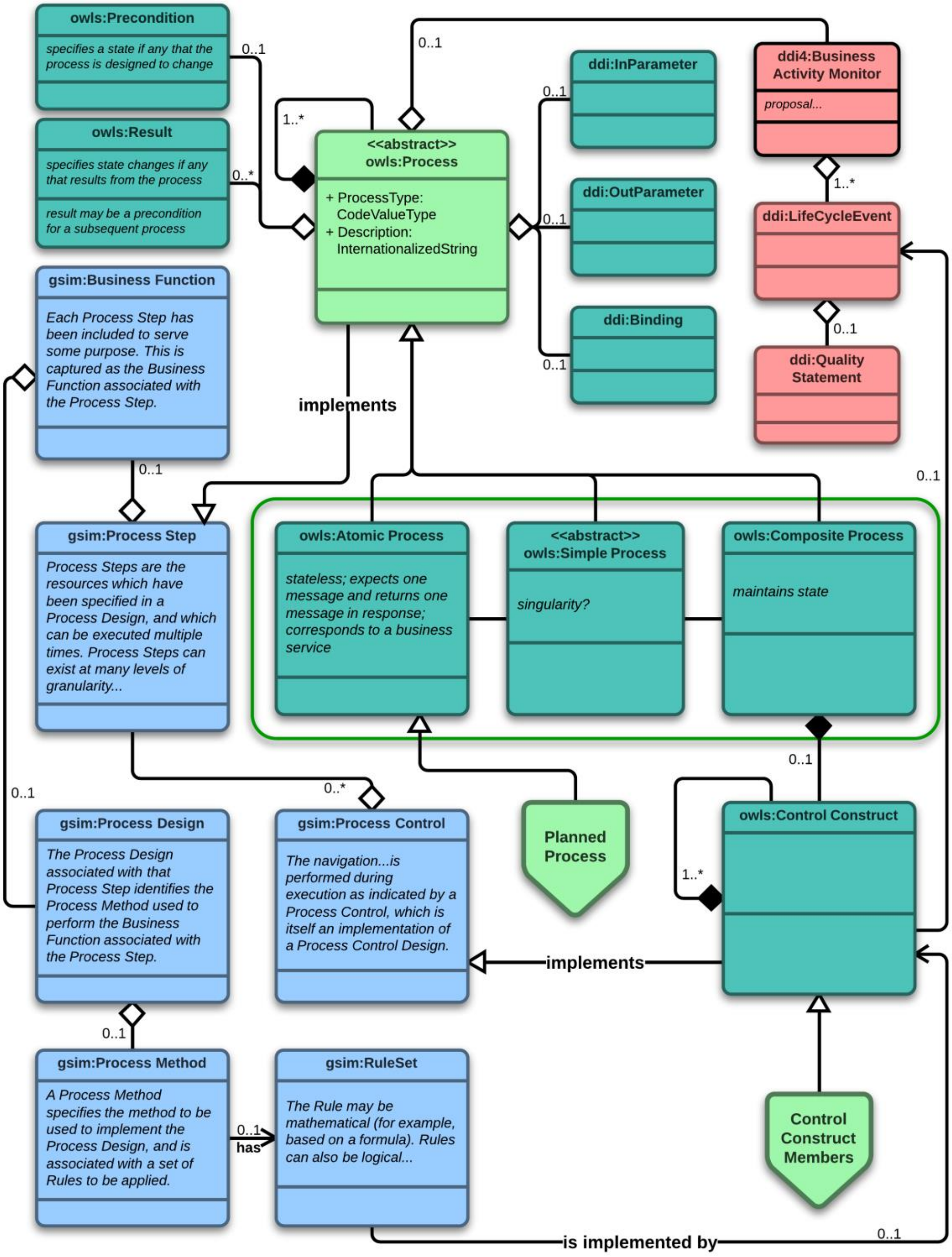
This process model isn't small. Up until now with other packages that have been developed, an approach was taken in which complexity was managed by providing the user with simple and advanced models. Formally, that has led to certain issues with respect to the relationship between a simple and an advanced model. Oliver Hopt discussed these issues in a presentation at EDDI 2013 in Paris.

Also in Paris in a subsequent sprint the idea of a document type that supported views was endorsed. Views are an alternative way to handle complexity. Instead of proliferating objects with copies, a view includes and excludes objects by reference. In this approach a package is defined which contains the whole. Then views slice and dice the package, tailoring it for targeted user profiles. Here we are taking the view approach. So first we will present the model as a whole. And then we will indicate how it might be tailored to create several views. Right now we are thinking three views – one for statisticians describing the production of statistics, a second for questionnaire authors and curators and a third for archivists intent on describing provenance chains.

The SDMX TWG uses EXL as a major input for the development of the Validation and Transformation Language (VTL). Currently, VTL is a work in progress. Perhaps it will become in part an information model and in part a data process application. Whatever the eventual boundaries of VTL, the GSIM/OWL-S/DDI information architecture will integrate with VTL in the future.

The Information Model in Detail

See Figures 10 through 13:



owls:Precondition
specifies a state if any that the process is designed to change
owls:Result
specifies state changes if any that results from the process
result may be a precondition for a subsequent process
0..1
0..*
1..*
0..1
<<abstract>>
owls:Process
+ ProcessType: CodeValueType
+ Description: InternationalizedString
ddi:InParameter
ddi:OutParameter
ddi:Binding
0..1
0..1
0..1
ddi4:Business Activity Monitor
proposal...
1..*
ddi:LifeCycleEvent
0..1
ddi:Quality Statement
gsim:Business Function
Each Process Step has been included to serve some purpose. This is captured as the Business Function associated with the Process Step.
implements
0..1
0..1
gsim:Process Step
Process Steps are the resources which have been specified in a Process Design, and which can be executed multiple times. Process Steps can exist at many levels of granularity...
owls:Atomic Process
stateless; expects one message and returns one message in response; corresponds to a business service
<<abstract>>
owls:Simple Process
singularity?
owls:Composite Process
maintains state
0..1
0..1
0..*
gsim:Process Design
The Process Design associated with that Process Step identifies the Process Method used to perform the Business Function associated with the Process Step.
gsim:Process Control
The navigation...is performed during execution as indicated by a Process Control, which is itself an implementation of a Process Control Design.
Planned Process
owls:Control Construct
1..*
implements
0..1
gsim:Process Method
A Process Method specifies the method to be used to implement the Process Design, and is associated with a set of Rules to be applied.
0..1
has
gsim:RuleSet
The Rule may be mathematical (for example, based on a formula). Rules can also be logical...
Control Construct Members
is implemented by
0..1

Figure 10: OWL-S Implements GSIM

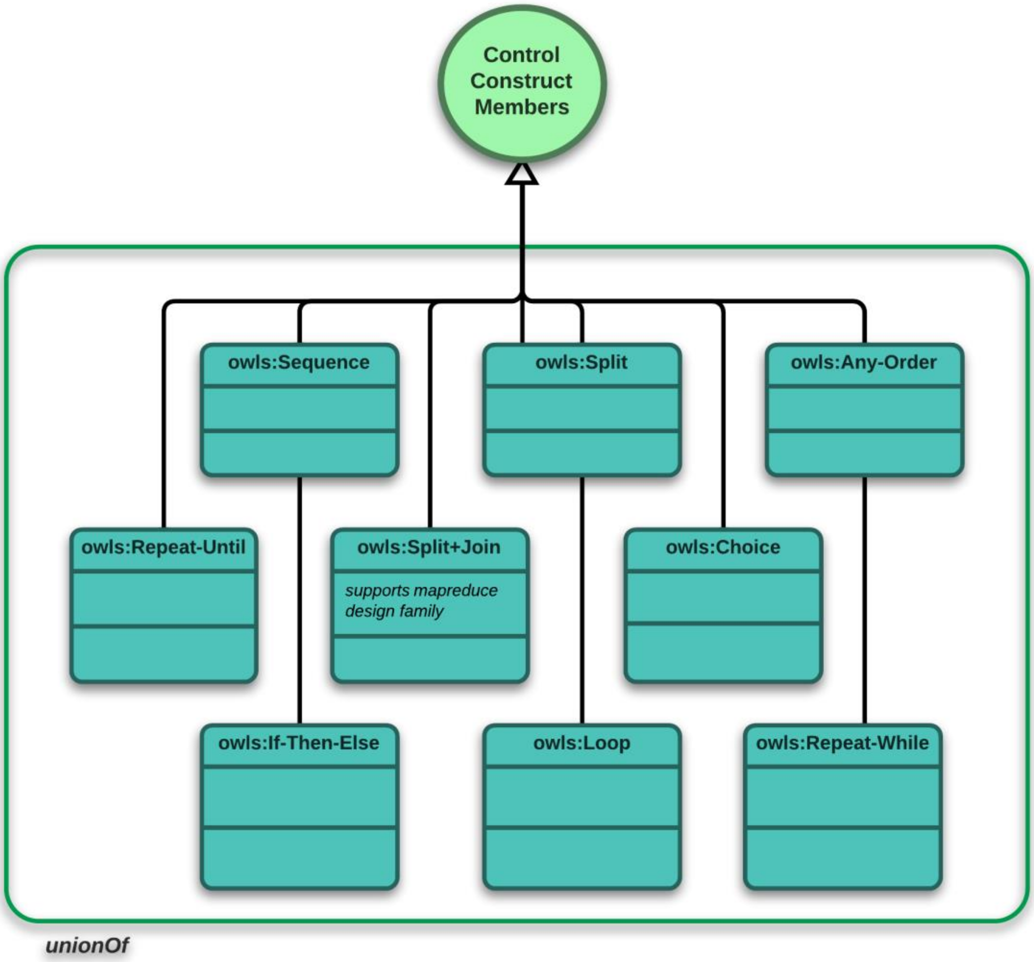


Figure 11: OWL-S Composite Process Control Construct Details

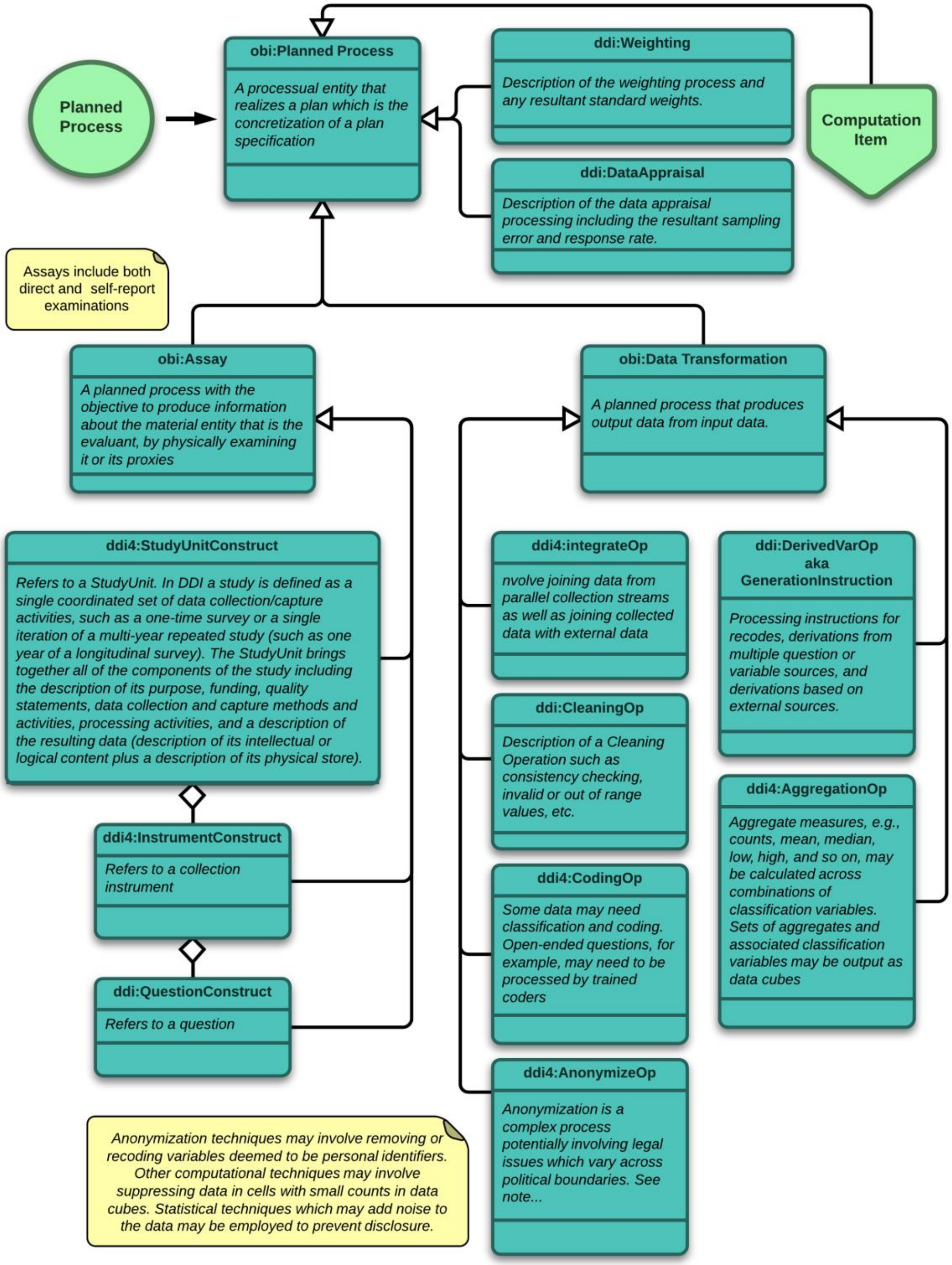

Planned Process
obi:Planned Process
A processual entity that realizes a plan which is the concretization of a plan specification
ddi:Weighting
Description of the weighting process and any resultant standard weights.
ddi:DataAppraisal
Description of the data appraisal processing including the resultant sampling error and response rate.
Computation Item
Assays include both direct and self-report examinations
obi:Assay
A planned process with the objective to produce information about the material entity that is the evaluant, by physically examining it or its proxies
obi:Data Transformation
A planned process that produces output data from input data.
ddi4:StudyUnitConstruct
Refers to a StudyUnit. In DDI a study is defined as a single coordinated set of data collection/capture activities, such as a one-time survey or a single iteration of a multi-year repeated study (such as one year of a longitudinal survey). The StudyUnit brings together all of the components of the study including the description of its purpose, funding, quality statements, data collection and capture methods and activities, processing activities, and a description of the resulting data (description of its intellectual or logical content plus a description of its physical store).
ddi4:InstrumentConstruct
Refers to a collection instrument
ddi:QuestionConstruct
Refers to a question
ddi4:integrateOp
nvolve joining data from parallel collection streams as well as joining collected data with external data
ddi:CleaningOp
Description of a Cleaning Operation such as consistency checking, invalid or out of range values, etc.
ddi4:CodingOp
Some data may need classification and coding. Open-ended questions, for example, may need to be processed by trained coders
ddi4:AnonymizeOp
Anonymization is a complex process potentially involving legal issues which vary across political boundaries. See note...
ddi:DerivedVarOp aka GenerationInstruction
Processing instructions for recodes, derivations from multiple question or variable sources, and derivations based on external sources.
ddi4:AggregationOp
Aggregate measures, e.g., counts, mean, median, low, high, and so on, may be calculated across combinations of classification variables. Sets of aggregates and associated classification variables may be output as data cubes
Anonymization techniques may involve removing or recoding variables deemed to be personal identifiers. Other computational techniques may involve suppressing data in cells with small counts in data cubes. Statistical techniques which may add noise to the data may be employed to prevent disclosure.

Figure 12: DDI Specializations of the OWL-S Atomic Process

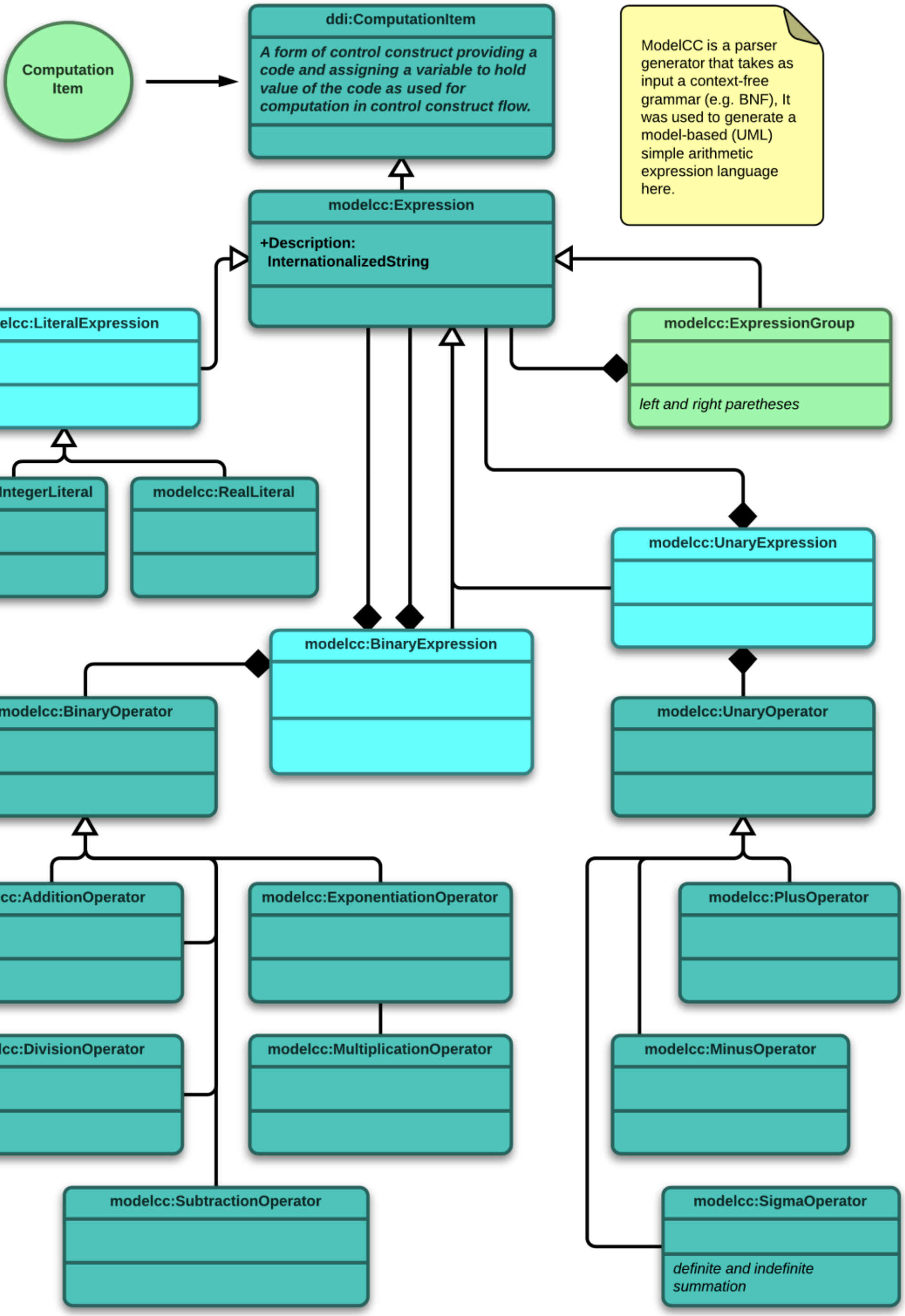

Computation Item
ddi:ComputationItem
A form of control construct providing a code and assigning a variable to hold value of the code as used for computation in control construct flow.
ModelCC is a parser generator that takes as input a context-free grammar (e.g. BNF), It was used to generate a model-based (UML) simple arithmetic expression language here.
modelcc:Expression
+Description: InternationalizedString
modelcc:LiteralExpression
modelcc:ExpressionGroup
left and right paretheses
modelcc:IntegerLiteral
modelcc:RealLiteral
modelcc:UnaryExpression
modelcc:BinaryExpression
modelcc:BinaryOperator
modelcc:UnaryOperator
modelcc:AdditionOperator
modelcc:ExponentiationOperator
modelcc:PlusOperator
modelcc:DivisionOperator
modelcc:MultiplicationOperator
modelcc:MinusOperator
modelcc:SubtractionOperator
modelcc:SigmaOperator
definite and indefinite summation

Figure 13: DDI ComputationItem with Mathematical Expression Extension

Three Sequence Diagrams

Here three sequence diagrams are presented to demonstrate certain paths a user might take through the process model, depending on the user's task:

- Protocol Execution Pipeline
- Provenance Chain
- Statistics Production Pipeline

Note that the Statistics Production Pipeline builds on the Provenance Chain. Also, note that in each case the path is a “happy” or, again, canonical path.

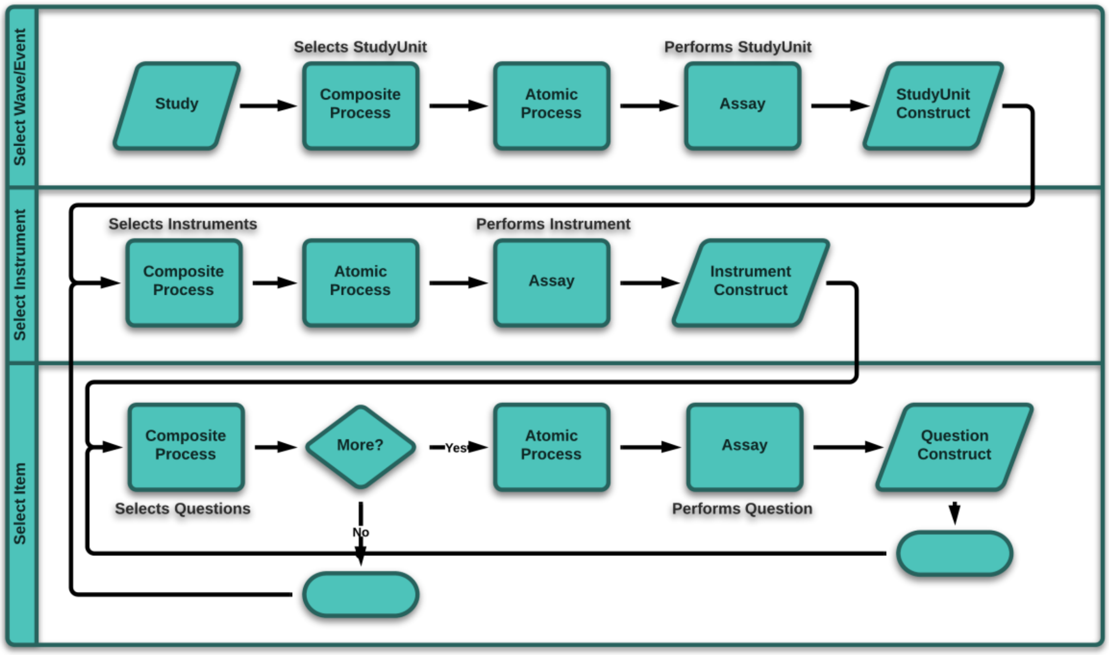


Figure 14: Protocol Execution Pipeline

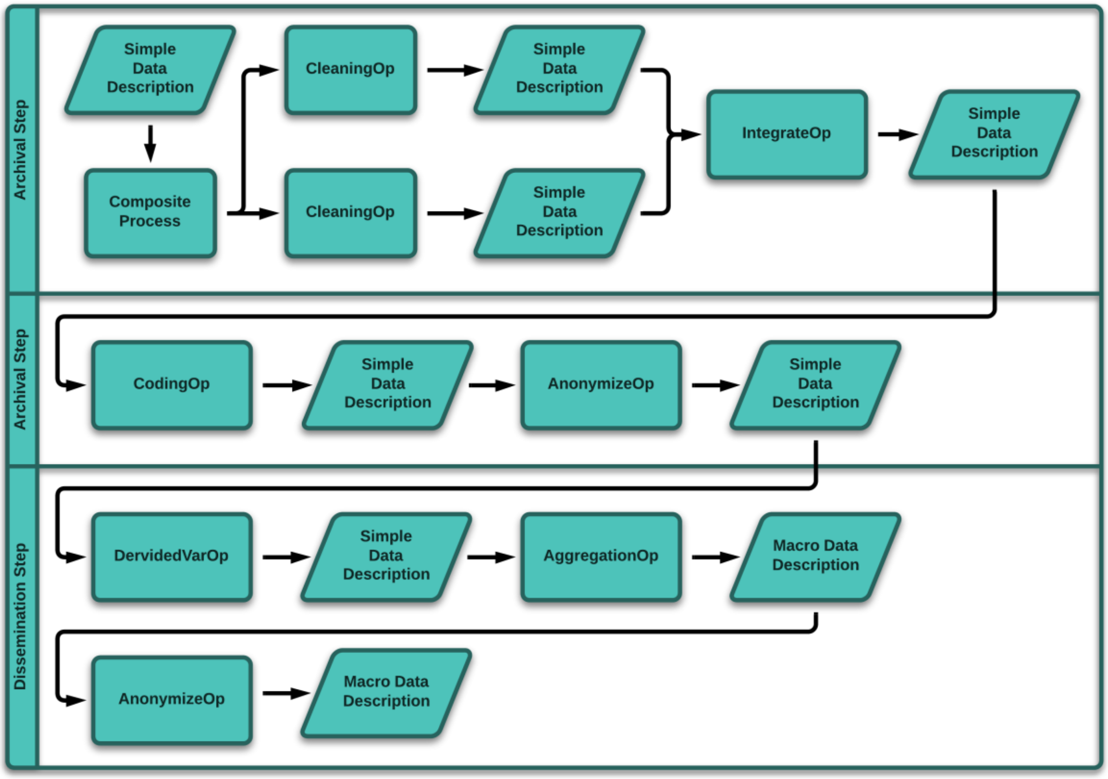


Figure 15: Provenance Chain

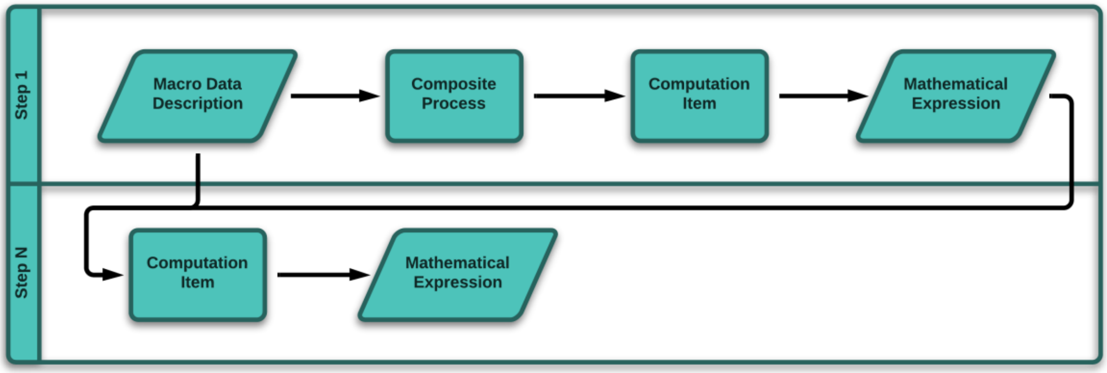


Figure 16: Statistics Production Pipeline

Implementation/Deployment: Possible Application and Technology Architectures

To be completed by Denis later this month. This section goes to these remarks Denis made:

- *In BPM this type of layering is sometimes considered: design (possibly with a layering between business and IT design), implementation/deployment, execution and monitoring. We could perhaps add on how process design and* *implementation are or are not supported by the model. Monitoring is already well covered…*

- *More generally do you think it could be relevant to distinguish representing business processes and data processing?*

In this section we may or may not want to discuss the process model in the context of Model-Driven Development[13] (MDD). We may also want to talk about related MDD efforts.[14]

Next Steps

Recall that in the Paris sprint the idea of a document type that supported views was endorsed. Views are an alternative way to handle complexity alongside the construction of so-called simple and advanced models. However, instead of proliferating objects with copies, a view includes and excludes objects by reference. In this approach a package is defined which contains the whole. Then, views slice and dice the package, tailoring it for targeted user profiles. Perhaps the next step is to create several views corresponding to different user profiles.

[13] See http://msdn.microsoft.com/en-us/library/aa964145.aspx

[14] See http://www.cdisc.org/healthcare-link. I can talk to this if Denis and Mary think it would be useful. I might point put that the GSIM/SDMX and DDI groups are pursuing in observational research a path that is comparable to the one CDISC is following with its Healthcare Link and the precursors of Healthcare Link – first Retrieve Protocol for Execution (RPE) and then Retrieve Form for Data Capture (RFD).