How to Create a Functional View

See also: Functional Views for Business Modelers

Requirements:

1. Clear definition of the purpose (Use Cases and Target Audience) for each Functional View
2. Clear definition of which classes are included and their specific use within the Functional View
3. Clear head class that allows identification of all included classes within the Functional View and all included objects in an instance of that view. This is the head class of the Functional View but there may be 1..n instances.

Structure:

1. A Functional View will be a subset of classes from the Library. They will be included with no restrictions or extensions.
2. All classes will retain the DDI Library namespace.
3. Documentation will be used to clarify what the properties and relationships of a class are relevant or “core” to the Use Case of the Functional View. The target class of all required relationships should be included in the “core”.
4. Any reference from an included class to a class NOT contained in a Functional View will be defined as an external reference.

   Note that in determining that a class is “out of scope” for the view the modeler is stating that for the intended use of the Functional View, the related classes are not required and may not be supported by tools/systems using the Functional View. Essentially restrictions should be used carefully, primarily to trim a level of detail not required or recommended by the Functional View. No required relationships can be removed. Documentation on a restricted class should make it clear that providing a reference to and external object will not make that content usable to a system that restricts it’s support to the contents of the Functional View, it should be considered to be locally usable content.

Creating a View:

1. Only classes listed in the view will be available in-line within an instance of the Functional View
2. Classes can be constrained through documentation and by NOT including a class that has a relation to an included class (For example if I included a Process but did not want to allow a description of the ProcessSteps and routing I would NOT include the class Sequence)
3. A Functional View must have a required class that will serve as a starting point for entry into an object model hierarchy identifying all the component classes that make up the Functional View

Include in a Functional View:

1. DocumentInformation – this provides a standard set of discovery and/or self-descriptive documentation of an instance of the Functional View and is optional for Functional Views that are transitory in nature
    a. Include related classes as appropriate for the Functional View:
        i. Annotation
            1. Agent types
            2. Access
            3. FundingInformation
            4. ExternalMaterial
            5. Coverage
                a. TemporalCoverage
                b. TopicalCoverage
                c. SpatialCoverage
                    i. BoundingBox
2. Enter each primary class
    a. Do NOT enter Extension Base or Abstract classes as this is handled in the transformation process and binding
3. Enter each class with a relation to that class which you wish to include ***in-line***
    a. ***All required relationships (1..1 or 1..n) must be included in the Functional View***
4. In the Functional View documentation include the following:
    a. Purpose of the Functional View
    b. Use Case(s)
    c. Target Audience (who are the intended users of the Functional View)
    d. List of constrained classes (those where one or more relations were not entered)
        i. Document use
    e. Documentation of any specialized use of classes within this Functional View
    f. General documentation on the use of the Functional View

    NOTE: As the requirements for entering Functional Views become stable, changes will be made to Drupal to support object level documentation and the identification of constrained classes

5. Additional notes for documentation:
    a. [include discussion of how Abstract classes are incorporated]

Applied Use of Functional Views:

1. Common core information (as defined by the Functional View based on DDI guidelines) should be completed if the binding is to be shared, in all or part, with other systems. This includes access to the instance or its contents in external search or retrieval systems.
2. Extensions of the Functional View beyond the documented usage of the classes with the Functional View should FIRST use the existing properties and relationships within the class.
3. When there is a need to restrict a Functional View it should be done on the instance level (i.e. like a DDI-L profile).

Appendix A: Decision of Modeling Team regarding Functional Views

1. A Functional View will be a sub-set of classes from the Library. They will be included with no restrictions or extensions.
2. Documentation will be used to clarify what the properties and relationships of a class are relevant or “core” to the Use Case of the Functional View.
3. Any reference (within an included class) to a class NOT contained in a view will be defined as an external reference. This is to limit the inclusion of referenced-but-unimportant cascade of objects being included in a view (unintentionally).
4. This requires documentation of the use of a class in a Functional View. Drupal will be edited to include the following in the current 2 column edit table for adding classes to a Functional View:
   a. A flag indicating that the full content of the class is being used by the Functional View
   b. A documentation field to provide usage information specific to the use of the class in the Functional View. Guidelines as to content and structure will be provided. Documentation is required when the full content flag is not checked.

General documentation on all Functional Views should include:

- Extensions beyond the documented usage of the classes within the Functional View should FIRST use the existing properties and relationships contained in the class.
- Information regarding buffering and the need to avoid the loss of content during exchanges.
- When there is a need to restrict a functional view it should be done on the instance level (i.e. like a DDI-L profile).