

## MFI PART 5 – METAMODEL FOR PROCESS MODELLING

### THOUGHTS AFTER THE KUNMING MEETING

#### INDIVIDUAL EXPERT CONTRIBUTION FROM KEITH GORDON

##### 1. *The relationship between "Process" and "Process Model"*

My rationale for requiring two associations between "Process" and "Process Model" is as follows.

One of the fundamental concepts of process modelling is that processes can be decomposed. This is represented by Figure 1 below.

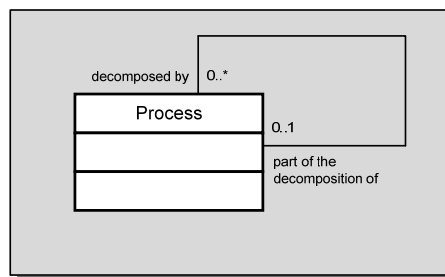


Figure 1: *The decomposition of processes*

Figure 1 shows that:

- Each **Process** may be decomposed by one or more other Processes
- Each **Process** may be part of the decomposition of one and only one other Process

There is then a requirement to describe that decomposition. The normal means for providing this description is a process model.

This requires the introduction of a Process Model metaclass as shown in Figure 2.

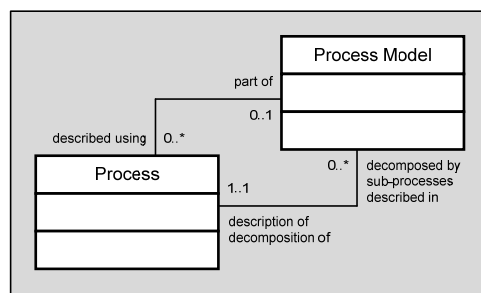


Figure 2: *The introduction of Process Model*

Figure 2 shows that:

- Each **Process** may be decomposed by sub-processes described in one or more Process Models
- Each **Process Model** may be described using one or more Processes.

- Each **Process** may be part of one and only one **Process Model**
- Each **Process Model** must be description of decomposition of one and only one **Process**

## 2. The relationship between "Composite Process" and "Dependency Construct"

Figure 3 shows a part of the metamodel in the current draft.

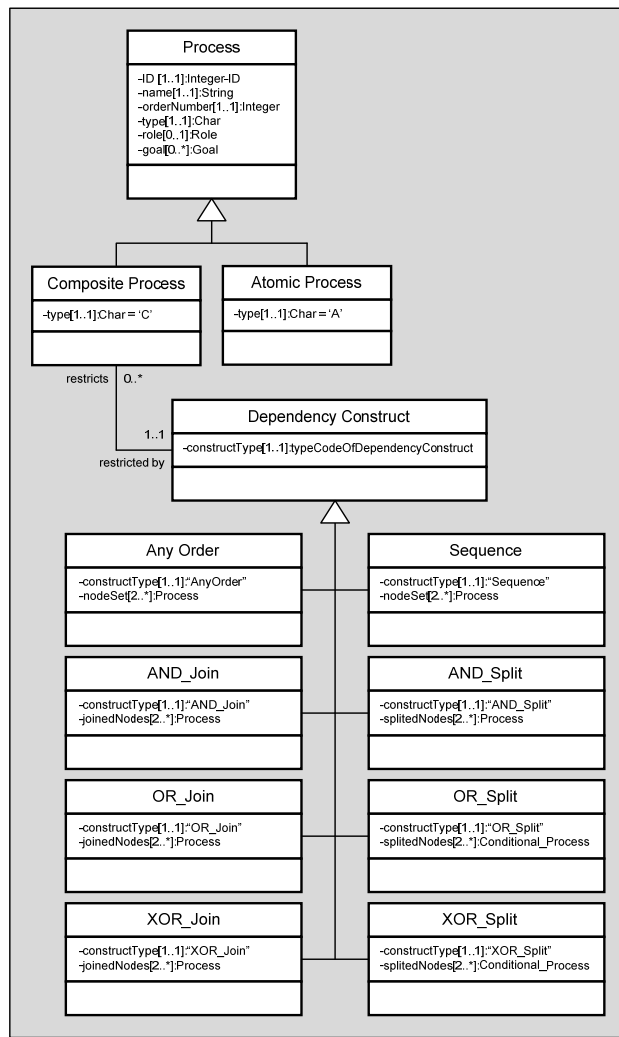


Figure 3: Current metamodel (part)

As an aside, I think there is an error because from our discussions in Kunming I believe the second attribute of "Sequence" should be "nodeList[2..\*]:Process"

The association between Composite Process and Dependency Construct in Figure 3 is:

- Each **Composite Process** must be restricted by one and only one **Dependency Construct**
- Each **Dependency Construct** may be restriction of one or more **Composite Processes**

The "may be" in the second sentence implies that it is possible for a Dependency Construct to exist in its own right without having to 'restrict' a Composite Process. But the only purpose of a Dependency

Construct is to restrict a Composite Process and it makes no sense for it to be allowed to exist without restricting a Composite Process.

Furthermore, the "one or more" in the second sentence implies that it is possible for a Dependency Construct to 'restrict' more than one Composite Process. But a Dependency Construct describes the Processes (Atomic and/or Composite) that are part of a Composite Process, so a Dependency Construct can only 'restrict' a single Composite Process.

This leads me to believe that the association between Composite Process and Dependency Construct should be:

- Each **Composite Process** must be restricted by one and only one **Dependency Construct**
- Each **Dependency Construct** must be restriction of one and only one **Composite Process**

ie, a fully mandatory one-to-one association, as shown in Figure 4 below.

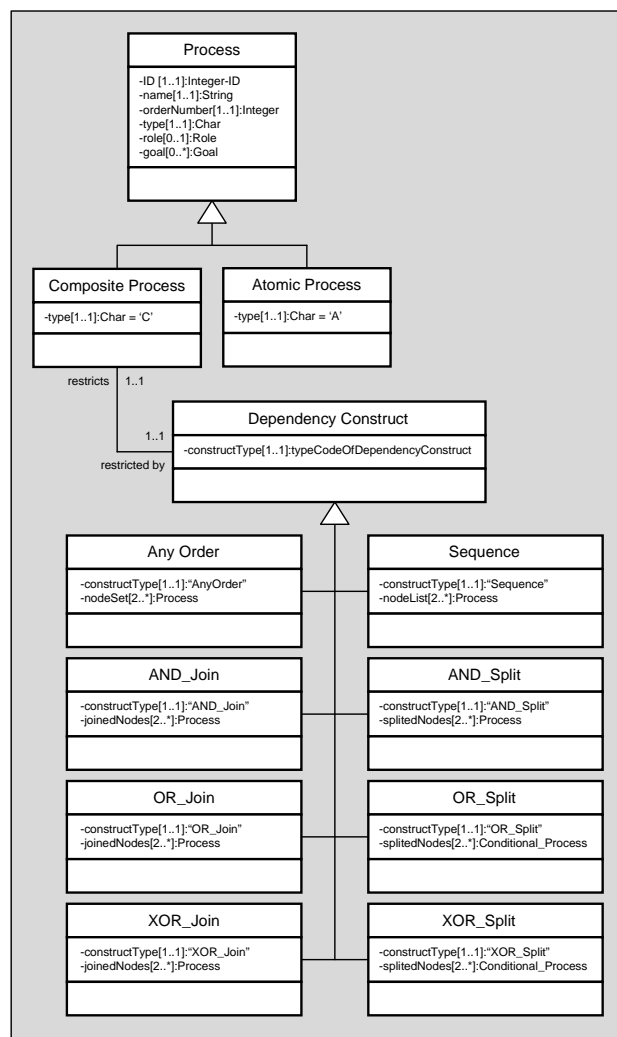


Figure 4: Part metamodel revised to show 1:1 association

Fully mandatory one-to-one associations normally imply that the two associated classes represent different information about the same 'thing'. This leads me to believe that Dependency Construct is a redundant metaclass and can be removed, with the current subclasses of Dependency Construct becoming subclasses of Composite Process.

This is shown in Figure 5 below.

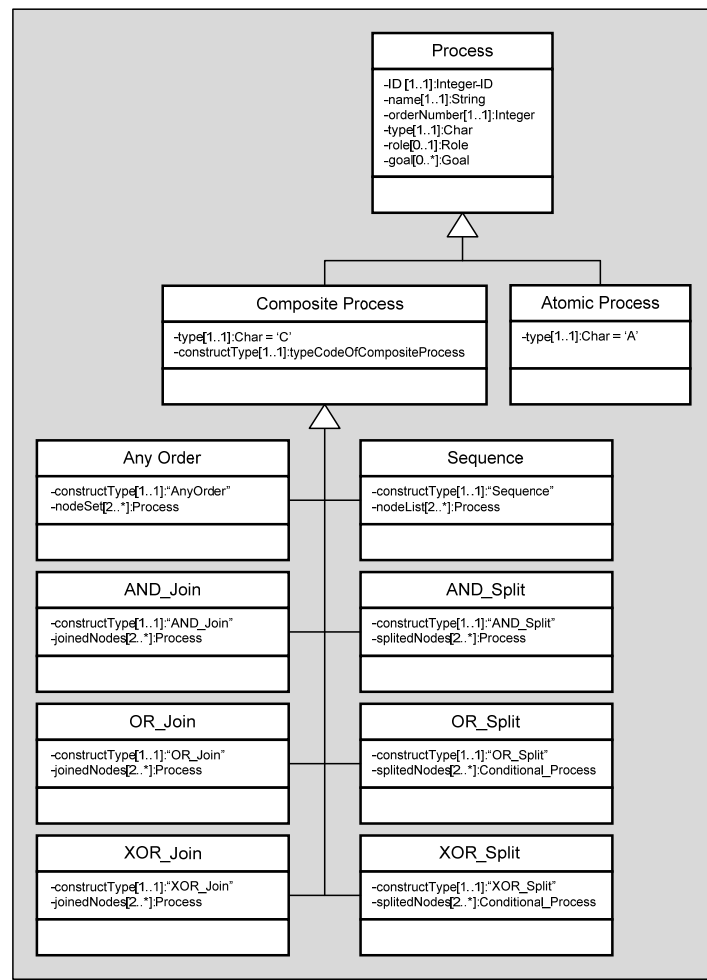


Figure 5: Part metamodel revised without Dependency Construct

### 3. "Conditional Process"

The attributes of Conditional Process in the current draft are:

- ❖ guardCondition: Event[1..1]
- ❖ resultProcess: Process[1..1]

Since the description of Event is "a metaclass designating a notable occurrence that triggers a process" it seems inappropriate to use Event as the datatype for guardCondition. A guard condition is not a trigger for a process but a statement determining which path is to be followed through the workflow specified by the model given the particular circumstances. Examples of sets of guard conditions are: ["yes", "no"], ["funds available", "funds not available"], ["acceptable", "minor revisions required", "major revisions required", "unacceptable"]. It would appear that the most appropriate datatype for guardCondition is probably String.

### 4. The description of "Process"

In the current draft Process is described as "an abstract metaclass representing the process described by a process model, which is the superclass of Atomic\_Process and Composite\_Process".

There are two problems with this description.

Firstly, it does not describe what a process is but how the process metaclass fits into the metamodel. A description should describe the significance of the object to the 'business' and in this case the business is process modelling. A description which merely describes how the class fits into the model does not meet this requirement.

Secondly, as shown above, a process may be described by a process model (or, more precisely, the decomposition of a process may be described by a process model) but a process may also be a component of a process model. The description does not allow for the second role.

There was some discussion of possible descriptions for Process during the Kunming meeting. A very detailed description (maybe too-detailed for MFI-5) from James Martin is:

"A repetitive, well-defined set of logical tasks that support one function, are repeatedly executed in a business segment, can be defined in terms of inputs and outputs, and have a definable beginning and end. Processes can be decomposed into processes and are triggered by an event and carried out by a business segment to achieve a stated purpose. A low-level process may be replicated across the business segment."

[Martin, J. (1990) *Information Engineering, Book II: Planning and Analysis* Prentice-Hall]

## 5. The attributes of "Process"

The attributes of Process in the current draft are:

- ❖ ID: Integer[1..1]
- ❖ name: String[1..1]
- ❖ orderNumber: Integer[0..1]
- ❖ type: char[1..1]
- ❖ goal: Goal[0..1]
- ❖ role: Role[0..\*]

The description provided for the ID attribute is "The unique identifier of a process." Is it appropriate to have this attribute in a conceptual model? Conceptually, processes do not have a 'unique identifier', leading to the conclusion that this attribute is included for implementation. But the Process metaclass will be a subclass of a metaclass in the Core model which will, in turn, be a subclass of a metaclass defined in Part 3 of 11179. As a subclass, Process should not have its own unique identifier.

The inclusion of the orderNumber attribute in Process means that the process being described by an instance of this class is not a process in the real world (which may be described in many models) but a process described in a particular process model. This reinforces that the association between Process and Process Model is many-to-one, as shown in Figure 2. If the association is to be many-to-many (as has sometimes been suggested) then this attribute would need to be moved to an associative class for that association.

The description provided for the orderNumber attribute is "A number allocating to a process. It's an optional attribute to identify a process." The second part of this description is redundant since the multiplicity of the attribute is clearly shown as "[0..1]" and should be deleted. The first part is, however, incomplete. Conceptually, processes do not have numbers allocated. Some modelling notations use 'numbers' to reference processes within a particular model, but there is no implied sequencing or ordering in this numbering. For example, a Level 1 model may have processes numbered 1, 2, 3, etc. A Level 2 model that decomposes process 2 on the Level 1 model may have processes numbered 2.1, 2.2, 2.3, etc and a Level 3 model that decomposes process 2.2 on the Level 2 model may have processes numbered 2.2.1, 2.2.2, 2.2.3, etc. This leads me to conclude that a better name for this attribute would be modelReferenceNumber, its datatype should be String, and

the description should be "A number allocated to a process to identify or reference that process within a particular process model."

The type attribute is something that is only needed for implementation (and then only for particular types of database management systems) and so should not be in this model, which is supposed to be a conceptual model. (The same comment applies to all 'type' attributes in the model.)

I feel a little uncomfortable with the goal attribute. In WG2 N1391 I said that we may need to be able to record the goal of a process (probably only for high-level processes) and recommended a multiplicity of [0..1]. I also said that it that there may be many outcomes from a process. In addition to the single optional goal, the current metamodel shows that a process can create many resources and can produce many events and the current working draft of MFI-8 shows that goals can be decomposed. Are goals different to outcomes? Are all outcomes either the creation of resources or the production of events?

In WG2 N1391 I also said that it may be necessary to know the actors performing this process and recommended a multiplicity of [0..\*] since some process modelling notations allow shared allocation of processes, particularly when two or more actors need to collaborate to complete a process. Is it the intention that this is achieved with the role attribute? If so, I am concerned. There are other 'roles' associated with processes, such as 'owner', 'beneficiary' and it is unclear to me how one role with respect to a particular process is distinguished from another role associated with the same process if the role attribute is taken at face value and encompasses all roles. I have looked at the current working draft of MFI-8 and see that Role has a name attribute, but it is unclear to me whether this is used to give the name of the role with respect to the organisation or the name of the role with respect to, in this case, the process.

## 6. "Resource" and "Event" associations with "Process"

The associations between the Resource and Event metaclasses and the Process metaclass in the current draft are shown in Figure 6 below.

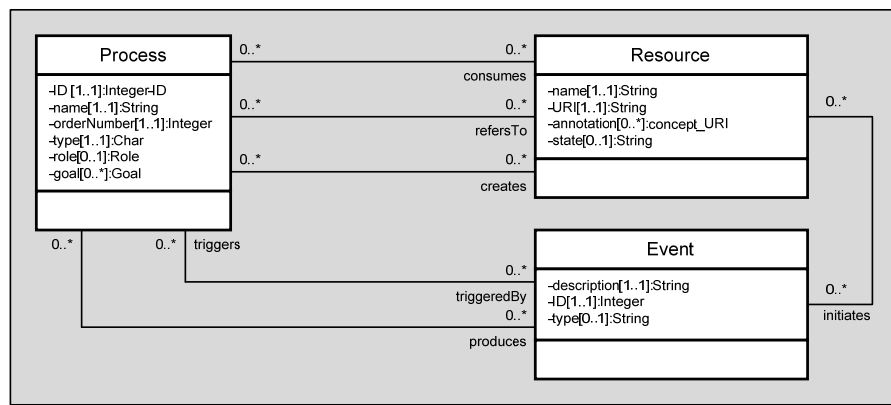


Figure 6: "Resource" and "Event" associations

Figure 6 shows that:

- Each **Process** may be consumer of one or more Resources
- Each **Resource** may be consumed by one or more Processes.
- Each **Process** may be creator of one or more Resources
- Each **Resource** may be created by one or more Processes.

- Each **Process** may be reference to one or more Events
- Each **Resource** may be referred to by one or more Processes.
- Each **Process** may be producer of one or more Events
- Each **Event** may be produced by one or more Processes.
- Each **Process** may be triggered by one or more Events
- Each **Event** may be trigger of one or more Processes.
- Each **Resource** may be initiator of one or more Events
- Each **Event** may be initiated by one or more Resources.

The fact that all of these associations are many-to-many associations means that Resource and Event are 'independent' metaclasses in that their instances are not tied to any one Process or Process Model and may be reused in many models. This means that they cannot be subclasses of the proposed Model or Model Component metaclasses of MFI Core. In the definitions in the current draft they are shown as having "Registered Item (from MDR)", but as their immediate superclass but there was discussion about having a third metaclass in MFI Core to cope with the 'independent' metaclasses in the subordinate parts of MFI.

Three of these associations cause particular concern.

Firstly, in process models each process has a single triggering event, so the "triggers" association between Process and Event should be many-to-one such that:

- Each **Process** may be triggered by one and only one Event
- Each **Event** may be trigger of one or more Processes.

Secondly, the implication of the only "initiates" association between Resource and Event is that if events are initiated at all they are initiated by resources. But in business process modelling the initiators of events, particularly external events, will often be people or organisations playing roles, all of which are represented by metaclasses in the current draft of MFI-8.

Thirdly, the "refersTo" association between Process and Resource seems very vague. In the Resource class definition in the current draft there is a "stateChangedBy" reference with a description that says "The process that changes the state of a kind of resource to another." Is the "refersTo" a typographical error? But if the intention is to record which processes change the state of which resources, the single state attribute in Resource will be inadequate (see below).

## **7. The attributes of "Resource"**

The attributes of Resource in the current draft are:

- ❖ name: String[1..1]
- ❖ URI: String[1..1]
- ❖ annotation: concept\_URI[0..\*]
- ❖ state: String[0..1]

The description of the mandatory URI attribute says "URI where a resource exists", but not every resource referred to in a process model will have a URI. Should this be [0..1]?

The description of the optional annotation attribute says "URI of the registered Ontology\_Atomic\_Construct based on MFI Ontology Registration. The concepts in ontology can be used to annotate resources participating in a process." and the datatype is shown as "concept\_URI".

"concept\_URI" is not defined in this part or in the current FDIS of MFI-3. Should the datatype be String?

As stated above, the single state attribute can only record a current state and this is borne out by the description "The current state that a resource remains to describe a state transition." But there appears to be the intention to record which processes cause resources to change state and this cannot be handled by the many-to-many association between Process and Resource and this state attribute in Resource. If the intention is to record which processes cause resources to change state then the state attribute should be moved to an associative class for the existing many-to-many association.

#### **8. The attributes of "Event"**

The attributes of Event in the current draft are:

- ❖ description: String[1..1]
- ❖ ID: Integer[1..1]
- ❖ type: String[0..1]

The description provided for the ID attribute is "The unique identifier of an event." As with the ID attribute in Process, is it appropriate to have this attribute in a conceptual model?

The description provided for the type attribute is "Type of an event. Its value could be "internal", "external" or "conditional"." Events that trigger processes are normally classified as "external", "internal time-based" or "internal conditional".

*Keith Gordon (10 August 2010)*