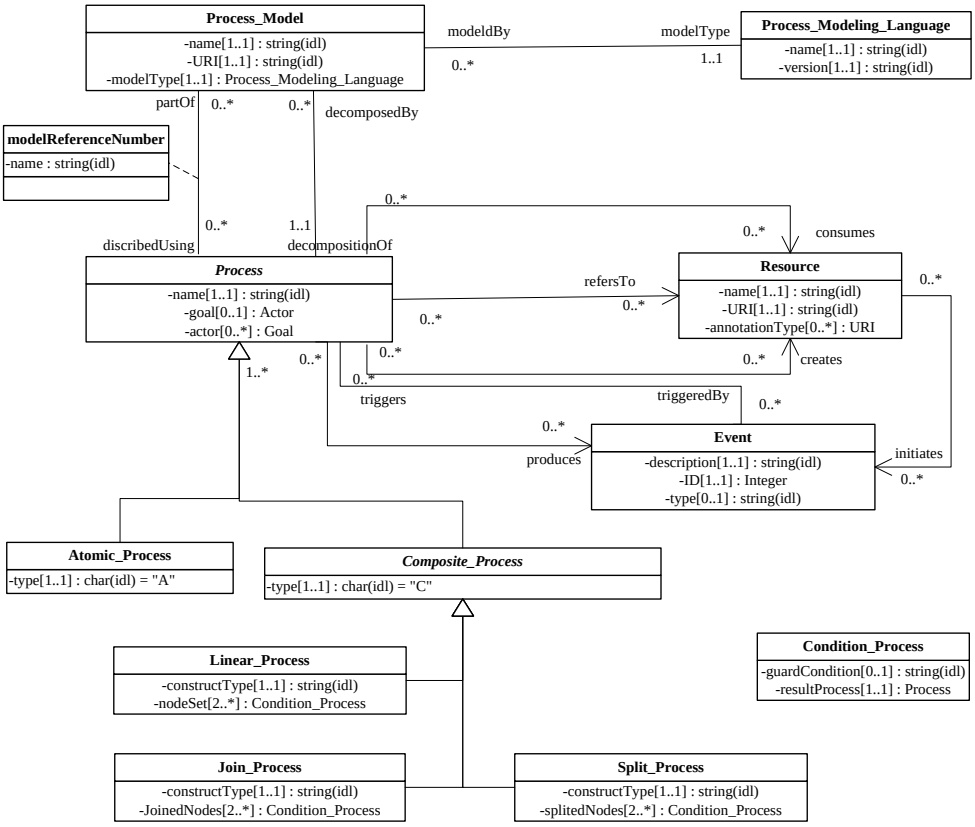
The new metamodel:



1.1 The relationship between "Process" and "Process Model".

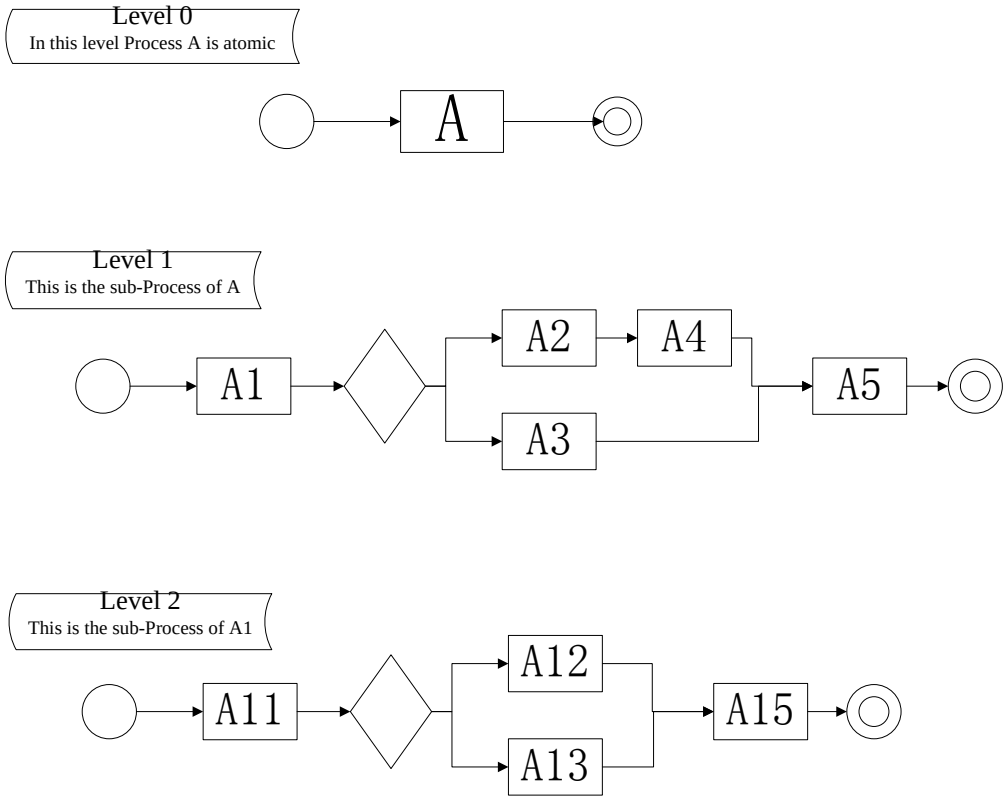
Because the process can be decomposed, the relationship of "Process" and "Process Model" may have to change.

Figure above shows that:

1) Each Process may be decomposed by sub-processes described by zero to many Process Models
2) Each Process Model must be description of decomposition of one and only one Process
3) Each Process may be part of zero to many Process Model
4) Each Process Model may be described using zero to many Processes.

This makes the process decomposition more clearly. All the atomic processes are real processes while all the composite processes are template processes in a certain level. Composite processes record associations of reference of atomic processes. Each instance of Process_Model is a description of its higher level Process.

For example,



For level0, it doesn't have a process_Model. In level1, there is an instance of Process_Model that describs the decomposition of process A in level 0. And, it is described by Process A1, A2, A3, A4 and A5.

1.2 The attributes of "Process"

The attribute of Process is changed according to the latest version of MFI-9:

1) name: String[1..1]
2) goal: Goal[1..1]
3) role: Role[1..*]

Firstly, the association between "process" and "process model" is to be many-to-many then this attribute would need to be moved to an associative class for that association. So we remove the attribute "orderName" and make it as a relation class "modelReferenceNumber". This can be used to record the level and the relationship between "Process" and "Process_Model".

Secondly, we remove the attributes of type and ID because we do not need them. Type is just needed for implementation.

1.3 The relationship between "Composite Process" and "Dependency Construct"

We delete the "Dependency Construct", and then "Composite Process" adds a new attribute. The attributes of "Composite Process" are changed to:

1) type: Char[1..1]

A Dependency Construct describes the Processes (Atomic and/or Composite) that is a part of a Composite_Process, so a Dependency Construct can only 'restrict' a single Composite Process. Then the relationship between composite Process and Dependency Construct becomes one to one. Fully mandatory one-to-one associations normally imply that the two associated classes represent different information about the same 'thing'. The Dependency Construct can be removed.

1.4 The subclasses of "Composite Process"

We divided 8 kinds of composite process into three categories. Both "Sequence" and "AnyOrder" can be treated as linear processes. When the value of attribute "constructType" is sequence means that the processes in "nodeSet" are orderable. "XOR_Join", "AND_Join" and "OR_Join" belong to "Join_Process"; while "XOR_Split", "AND_Split" and "OR_Split" are three kinds of "Split_Process". Then, we change the type of attribute "joinedNodes" and "splitNodes" to "Condition_Process" in order to satisfy the need of guardCondition.

1.5 The attributes of "Condition_Process"

The attribute of Condition_Process in may be changed to:

1) guardCondition: String[1..1]
2) resultProcess: Process[1..1]

The type of "guardCondition" is String rather than Event since we usually decide which way to go according to the result of condition, such as "yes" or "no". They are Strings.

1.6 The attributes of "Event"

The attributes of Event in the current draft are:

1) description: String[1..1]
2) ID: Integer[1..1]
3) type: String[0..1]

The description provided for the attribute “type” is the type of an event. Events that trigger processes are normally classified as "external", "internal time-based" or "internal conditional" instead of "internal", "external" or "conditional" in the previous version of MFI-5..