Mapping OPM to BOM

Made by:

Noam Shmueli,

Hadar Bibi,

Michal Etzion, and

Sergey Bolshchikov

Technion - Israel Institute of Technology

2012

Contents

Contents 2
Introduction 3
Pattern of Interplay 3
  Mappings Table 4
  Rules 5
  Algorithm 5
State Machine 7
  Mappings Table 8
  Algorithm 8
Entity Type 9
  Mappings Table 9
  Algorithm 10
Event Type 11
  Mappings Table 11
  Algorithm 11

Introduction

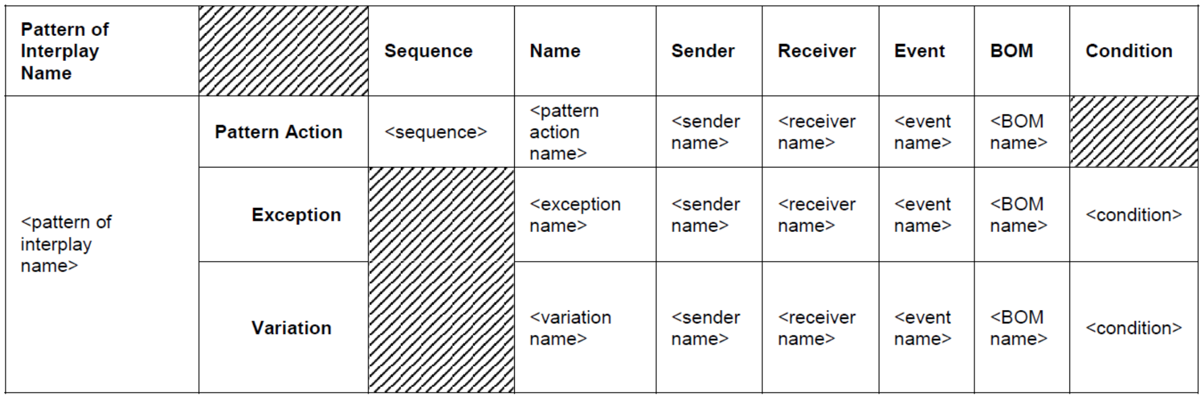
In this document we will map OPD (Object Process Diagram) to BOM (Basic Object Model).

We will focus in four tables related to Conceptual Model Definition:

- Pattern of Interplay
- State Machine
- Entity Type
- Event Type

Pattern of Interplay

Table 6-5 Pattern of Interplay Table Format



Pattern of Interplay Name	▨	**Sequence**	**Name**	**Sender**	**Receiver**	**Event**	**BOM**	**Condition**
<pattern of interplay name>	**Pattern Action**	<sequence>	<pattern action name>	<sender name>	<receiver name>	<event name>	<BOM name>	▨
	Exception	▨	<exception name>	<sender name>	<receiver name>	<event name>	<BOM name>	<condition>
	Variation	▨	<variation name>	<sender name>	<receiver name>	<event name>	<BOM name>	<condition>

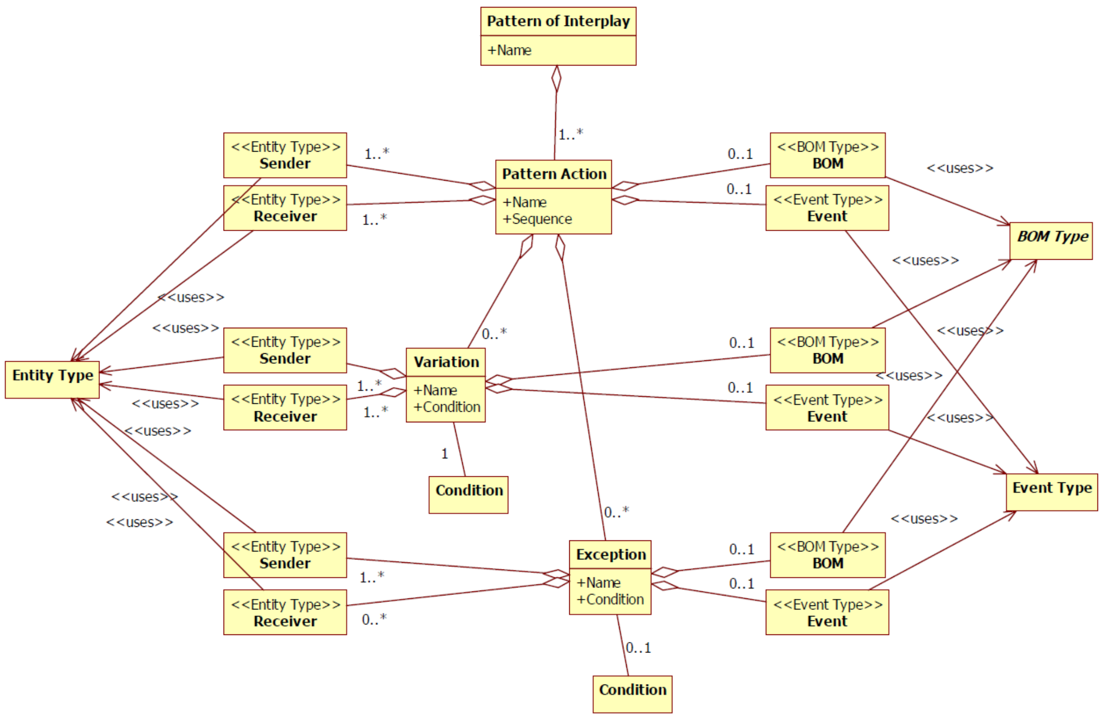


Figure 6-4 BOM Pattern Action Relationship

Mappings Table

#	OPM	BOM	Comment
1	Process Name of leaf	Pattern of Interplay Name	Diagram in the tree
2	Sub Process Name of leaf	Pattern Action Name	
3	Agent Link	Sender	
4	Instrument Link	Sender	
	Consumption Link	Sender/ Receiver	Need to check if the sender and the receiver are the same - If already exists sender, then the object that is connected with consumption link is the receiver. If already exists object with consumption link take it as sender.
5	Result Link	Receiver	
6	Effect Link	Sender/ Receiver	Need to check if the sender and the receiver are the same - If already exists sender then, the object that is connected with effect link is the receiver. If already exists object with result link take it as sender.
7	State associated with condition /event link	Event	

8	Object associated with condition /event Link	Condition	
9	Invocation Link	Receiver and Event	We create a new object called "Output of the top process' name". This object is the receiver of the top process and the event of the above process.

Notes:

1. We will not produce variants since OPM has no way to define a meta-process that can occur in several different variants.

Exceptions:

1. If the sub process is connected to an attribute, then this is not a sender, we need to navigate to the object that exhibit the attribute, and take is as a sender.

Rules

1. To avoid the exceptions above: we will replace the effect link of the attribute with effect link of the object that exhibits the attribute.
2. In case of invocation link – we will replace the connection between the two processes with:
    a. We will create a new object named 'Output of the top process' name.
    b. We will connect the top process to the new object.
    c. We will connect the new object with the second process.

Algorithm

1. First of all transform the current model representation to other easier to manipulate model representation:
    a. Remove irrelevant links and objects.
    b. Remove objects that are not directly linked to processes (for example in generalization – the parent will be a characteristic of the inherit class).
    c. Remove Undirectional/Bidirectional links.
2. Apply the above rules on the xml file (the rest of the algorithm will work on the transformed xml file).
3. For the SD's of the lowest level do as follows:
    - 3.1. Set the 'Pattern of Interplay Name' as the Process Name of leaf.
    - 3.2. Iterate over the sub-processes
        - 3.2.1. Initiate a new object that will represent a row to be added to the list (the list will represents the required table)
        - 3.2.2. In the row set the 'Pattern Action Name' as the Sub Process name.
        - 3.2.3. For each agent/instrument link add the name of the connected object to the list of 'Sender' of the row.

3.2.4. For each instrument/consumption link add the name of the connected object to the both lists of ‘Sender’ and ‘Receiver’ of the row.

3.2.5. For each result link add the name of the connected object to the list of ‘Receiver’ of the row.

3.2.6. For each effect link

3.2.6.1. If already exists sender then add the object that is connected with effect link to the list of ‘Receiver’ of the row.

3.2.6.2. Otherwise if already exists object with result link add the object name that is connected with effect link to the list of ‘Sender’ of the row.

3.2.7. For each condition/event link that connects state of object – add the following string: ‘<object_name> is in <state_name>’ to the condition/event of the row.

3.2.8. Add the row to the list.

3.3. Return the list (this list represents the required table).

State Machine

Table 6-8 State Machine Table Format

State Machine Name	Conceptual Entities	State		
		State Name	Exit Condition	
			Exit Action	Next State
<state machine name>	<entity type name>, <entity type name>	<state name>	<exit action name>	<next state name>
			<exit action name>	<next state name>
		<state name>	<exit action name>	<next state name>
<state machine name>	<entity type name>, <entity type name>	<state name>	<exit action name>	<next state name>
			<exit action name>	<next state name>
<state machine name>	<entity type name>	<state name>	<exit action name>	<next state name>
			<exit action name>	<next state name>

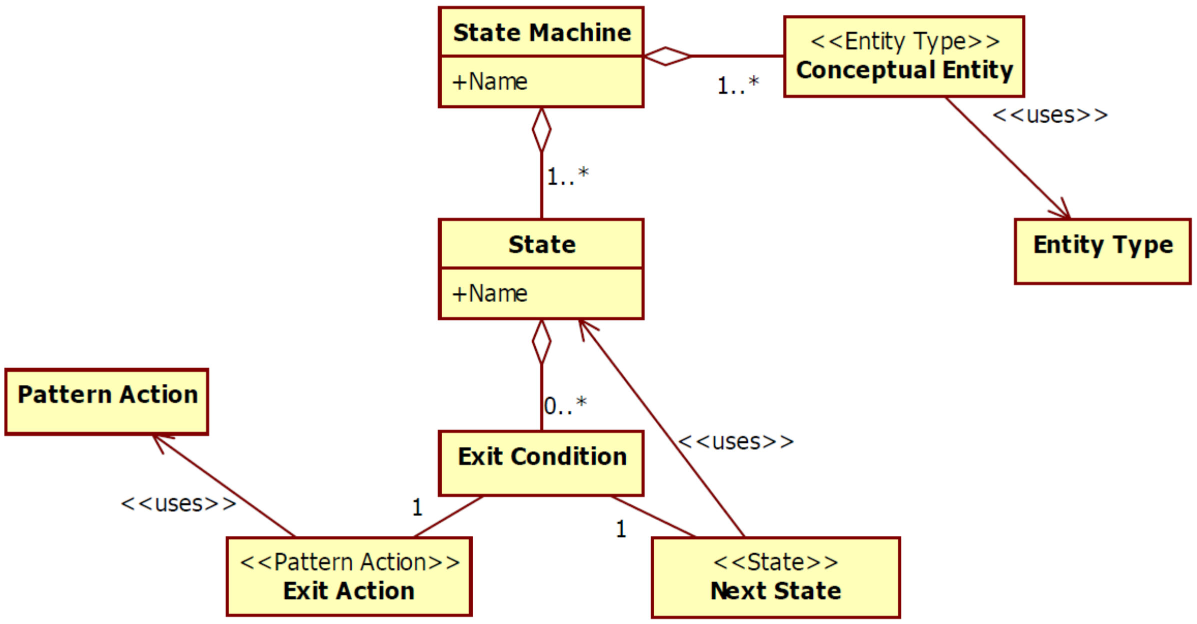


Figure 6-5 BOM State Machine Relationship

Mappings Table

#	OPM	BOM
1	Object Name	State Machine Name, Conceptual Entity
2	Object State	State
3	Process with Result Link	Exit Action
4	Object new state	Next State

Note:

1. Exit Condition consists of Exit Action and Next State.
2. Assume that changing of object state is done by only one process (i.e. changing from state 1 to state 2 is done by process A, changing from state 2 to state 3 is done by process B and so on)s

Algorithm

1. As mentioned below in the first table, perform the preliminary step.
2. Initiate a map – key = object name, value = state machine
3. Scan all objects in all levels in the SD hierarchy.
4. For each found object –
    - 3.1. If Object contains states:
        - 3.1.1. If Object not in map - add it to the map
        - 3.1.2. Otherwise get a reference to the object
        - 3.1.3. Update a state machine for the object – according to the change of the states + the processes that affects the states.
5. In the end we have list of state machines.

Entity Type

Table 6-11 Entity Type Table Format

Entity Type Name	Entity Type Characteristic Name
<entity type name>	<characteristic name>, <characteristic name>
<entity type name>	<characteristic name>, <characteristic name>

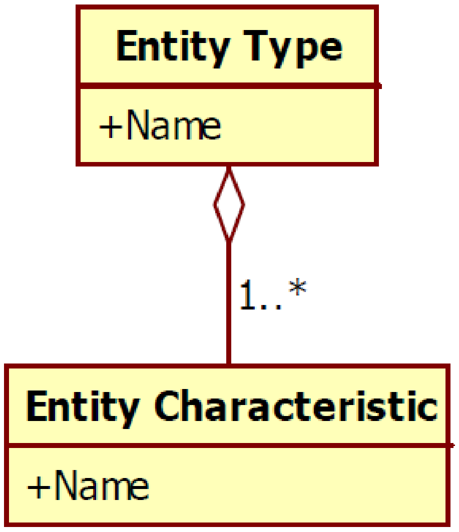


Figure 6-6 BOM Entity Type Relationship

Mappings Table

#	OPM	BOM
1	Object	Entity Type Name
2	Object states	Entity Type Characteristics Name
3	Object attributes (Exhibition / Aggregation)	

Algorithm

1. In that algorithm all objects are relevant to the transformation (even those that are not linked directly to a process i.e. abstract object)
2. Initiate a map – key = object name, value = Entity
3. Scan all objects in all levels in the SD hierarchy.
4. For each found object –
    - 3.2. If Object not in map - add it to the map
    - 3.3. Otherwise get a reference to the object
    - 3.4. Add the object attributes to the entity of the map.
5. In the end we have a map of entities, where each entity consists of list of characterizes.

Event Type

Table 6-14 Event Type Definition Table Format

Event Type Name	Source Characteristic Name	Target Characteristic Name	Content Characteristic Name	Trigger Condition
<event type name>	<source char>	<target char>, <target char>,	<content char>, <content char>	<trigger expression>, <trigger expression>,
<event type name>	<source char>	<target char>	<content char>, <content char>	<trigger expression>, <trigger expression>,
<event type name>	<source char>	<target char>, <target char>	<content char>, <content char>	<trigger expression>

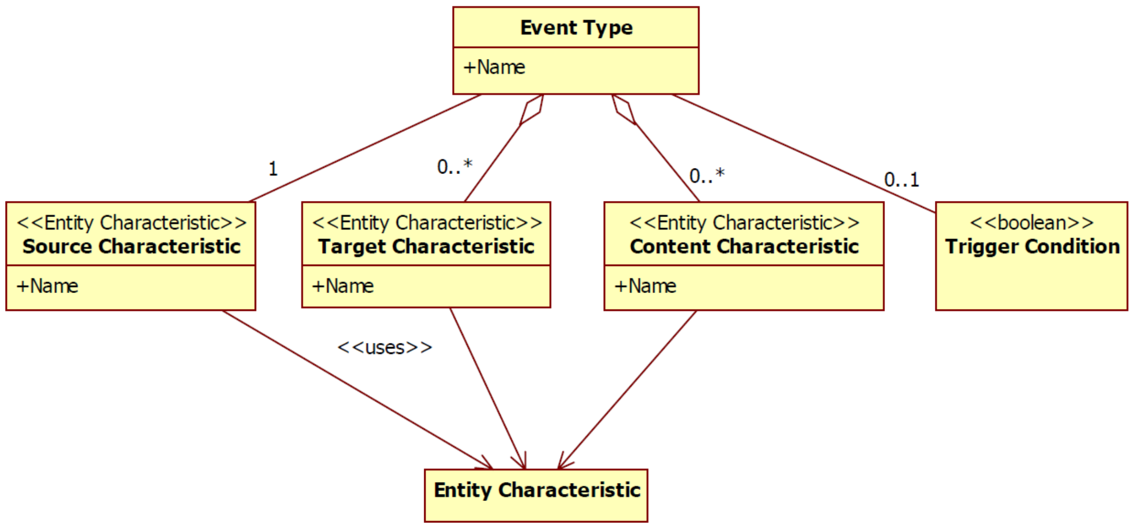

Event Type
+Name
1
0..*
0..*
0..1
<<Entity Characteristic>> Source Characteristic
+Name
<<Entity Characteristic>> Target Characteristic
+Name
<<Entity Characteristic>> Content Characteristic
+Name
<<boolean>> Trigger Condition
<<uses>>
Entity Characteristic

Figure 6-8 BOM Event Type Relationship

Mappings Table

#	OPM	BOM
1	Agent Link	Source Characteristic
2	Instrument Link	
3	Result Link	Target Characteristic
4	Process Name of leaf	Event type name
5	Effect Link	Content Characteristic
6	Instrument Event Link	Trigger Condition
7	Consumption Event Link	

Algorithm

1. For the SD's of the lowest level do as follows:
   1.1. Iterate over the sub-processes

    1.1.1. Initiate a new object that will represent a row to be added to the list (the list will represents the required table)
    1.1.2. In the row set the ‘Event type name’ as the Sub Process name.
    1.1.3. For each agent/instrument link add the name of the connected object to the list of ‘Source Characteristics’ of the row.
    1.1.4. If a result link emerge from the sub process, add the object name that is the target of the link to the list of ‘Target Characteristic’.
    1.1.5. If an effect link emerge from the sub process, add the object name that is the target of the link to the list of ‘Content Characteristic’.
    1.1.6. If an instrument / consumption event link is an input to the sub process, add the object name that is the source of the link to the list of ‘Trigger Condition’.
    1.1.7. Add the row to the list.

1.2. Return the list (this list represents the required table).