

The liveness2BPMN transformation algorithm

The transformation algorithm is a recursive algorithm that takes the liveness formula expression elements (right hand side of the formula) from left to right and applies the templates shown in Figure 1, gradually building the BPMN process model. For applying templates, keep in mind that the control flows from left to right, thus, if a template follows another, then it is connected to its rightmost element.

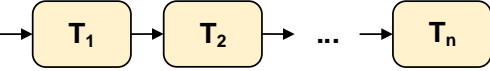
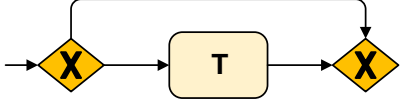
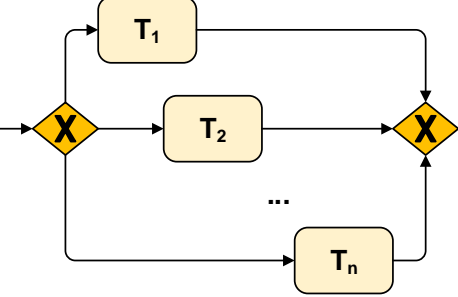
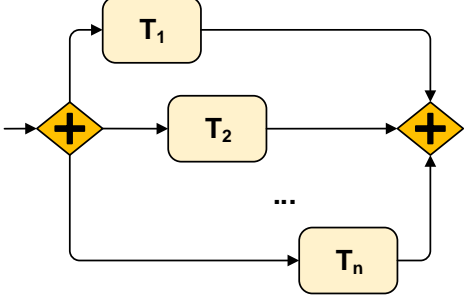
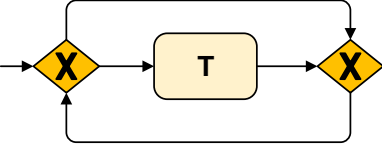
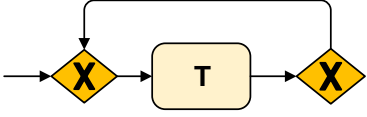




Op.	BPMN Transformation Template	Op.	BPMN Transformation Template
$T_1 \cdot T_2 \cdot \dots \cdot T_n$		$[T]$	
$T_1 T_2 \dots T_n$		$T_1 T_2 \dots T_n$	
T^*		$T \sim$	
$T = \text{"Send"}$		$T +$	
$T = \text{"Receive"}$		$T = \text{"Receive"}$	

Figure 1. The BPMN 2.0 transformation templates for the liveness formula operators and task names

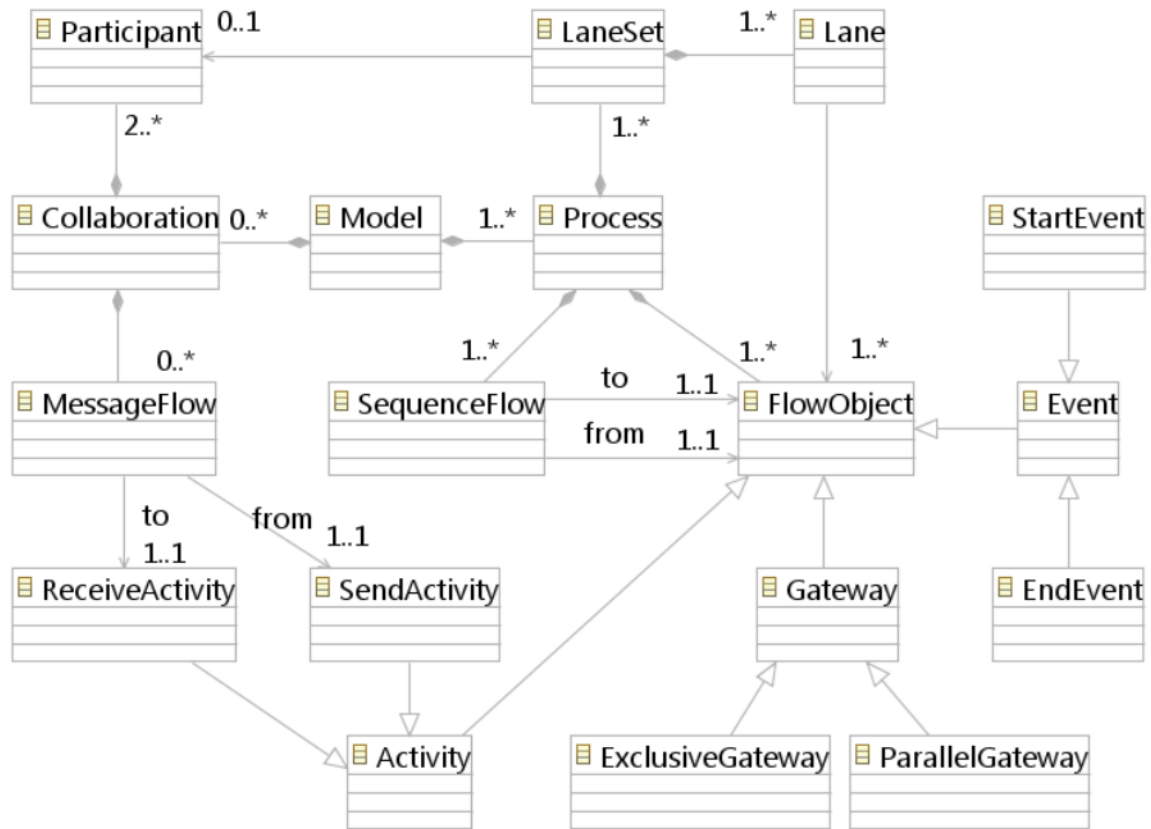


Figure 2. Abstract BPMN 2.0 metamodel

The transformation algorithm uses the elements depicted in Figure 2 in order to provide a complete BPMN model out of a simple liveness formula. It is important to note that all BPMN elements inherit from FlowObject therefore they can be connected with SequenceFlows. The algorithm is provided with a liveness formula and an empty BPMN model in order to recursively produce the desired result. The complete algorithm is provided for the reader.

```

1  Program transform(String liveness, Model model)
2      Process process = new Process
3      model.processes.add(process)
4      StartEvent startEvent = new StartEvent
5      startEvent.type = start
6      process.add(startEvent)
7
8      FlowObject lastActivity = createProcess(liveness.formula1.expression,
9  workflowProcess, startEvent)
10     EndEvent endEvent = new EndEvent
11     endEvent.type = end
12     process.add(endEvent)
13     SequenceFlow transition = new SequenceFlow
14     transition.from = lastActivity
15     transition.to = endEvent
16     process.add(transition)
17 End Program
18

```

```

19 Function FlowObject createProcess(String expression, Process process,
20 FlowObject activity)
21     List terms = new List
22     For Each termi In expression
23         terms.add(termi)
24     End For
25     If terms.size() > 1 Then
26         If expression Is sequentialExpr Then
27             For Each termi In expression
28                 FlowObject newActivity = createProcess(termi,
29 process, activity)
30                 activity = newActivity
31             End for
32         Else If expression Is orExpr
33             Gateway xorEntryGateway = new Gateway
34             xorEntryGateway.gatewayType = XOR
35             process.add(xorEntryGateway)
36             SequenceFlow transition = new SequenceFlow
37             transition.from = activity
38             transition.to = xorEntryGateway
39             process.add(transition)
40             Gateway xorExitGateway = new Gateway
41             xorExitGateway.gatewayType = XOR
42             process.add(xorExitGateway)
43             For Each termi In expression
44                 FlowObject newActivity = createProcess(termi,
45 workflowprocess, xorEntryGateway)
46                 transition = new Transition
47                 transition.from = newActivity
48                 transition.to = xorExitGateway
49                 process.add(transition)
50             End for
51             activity = xorExitGateway
52         Else If expression Is parallelExpr
53             Gateway parallelEntryGateWay = new Gateway
54             parallelGateWay.gatewayType = Parallel
55             process.add(parallelEntryGateway)
56             SequenceFlow transition = new SequenceFlow
57             transition.from = activity
58             transition.to = parallelEntryGateway
59             process.add(transition)
60             Gateway parallelExitGateway = new Gateway
61             parallelExitGateway.gatewayType = Parallel
62             process.add(parallelExitGateway)
63             For Each termi In expression
64                 FlowObject newActivity = createProcess(termi,
65 process, parallelEntryGateway)
66                 transition = new Transition
67                 transition.from = newActivity
68                 transition.to = xorExitGateway
69                 process.add(transition)
70             End For
71             activity = parallelGateway
72         End If
73     For Each termi In expression
74         If termi Is basicTerm
75             boolean foundLeftHandSideEqualsBasicTerm = false

```

```

76      For Each formulai In liveness
77          If formulai.leftHandside = termi Then
78              FlowObject newActivity =
79                  createProcess(formulai.expression,
80                      process, activity)
81                  activity = newActivity
82                  foundLeftHandSideEqualsBasicTerm = true
83          End If
84          If foundLeftHandSideEqualsBasicTerm = false
85              Activity newActivity = new Activity
86              process.add(newActivity)
87              SequenceFlow transition=new SequenceFlow
88              transition.from = activity
89              transition.to = newActivity
90              process.add(transition)
91              activity = newActivity
92          End If
93      Else If (termi is of type '(' term ')') Then
94          FlowObject newActivity = createProcess(term, process,
95              activity)
96          activity = newActivity
97      Else If (termi is of type '[' term ']') Then
98          Gateway xorEntryGateway = new Gateway
99          xorEntryGateway.gatewayType = XOR
100         process.add(xorEntryGateway)
101         Gateway xorExitGateway = new Gateway
102         xorEntryGateway.gatewayType = XOR
103         process.add(xorEntryGateway)
104         SequenceFlow transition = new SequenceFlow
105         transition.from = activity
106         transition.to = xorEntryGateway
107         process.add(transition)
108         FlowObject newActivity = createProcess(term, process,
109             xorEntryGateway)
110         SequenceFlow transition = new SequenceFlow
111         transition.from = newActivity
112         transition.to = xorExitGateway
113         process.add(transition)
114         SequenceFlow transition = new SequenceFlow
115         transition.from = xorEntryGateway
116         transition.to = xorExitGateway
117         activity = xorExitGateway
118      Else If (termi is of type '*') Then
119          Gateway xorEntryGateway = new Gateway
120          xorEntryGateway.gatewayType = XOR
121          process.add(xorEntryGateway)
122          Activity xorExitGateway = new Activity
123          xorEntryGateway.gatewayType = XOR
124          process.add(xorEntryGateway)
125          SequenceFlow transition = new SequenceFlow
126          transition.from = activity
127          transition.to = xorEntryGateway
128          process.add(transition)
129          FlowObject newActivity = createProcess(term, process,
130              xorEntryGateway)
131          SequenceFlow transition = new SequenceFlow
132          transition.from = newActivity

```

```

133         transition.to = xorExitGateway
134         process.add(transition)
135         SequenceFlow transition = new SequenceFlow
136         transition.from = xorEntryGateway
137         transition.to = xorExitGateway
138         process.add(transition)
139         SequenceFlow transition = new SequenceFlow
140         transition.from = xorExitGateway
141         transition.to = startof(term)
142         process.add(transition)
143         activity = xorExitGateway
144     Else If (termi is of type '~') Then
145         FlowObject newActivity = createProcess(term, process,
146         activity)
147         SequenceFlow transition = new SequenceFlow
148         transition.from = newActivity
149         transition.to = startof(term)
150         process.add(transition)
151         activity = newActivity
152     Else If (termi is of type '+') Then
153         Gateway xorExitGateway = new Gateway
154         xorExitGateway.gatewayType = XOR
155         process.add(xorExitGateway)
156         FlowObject newActivity = createProcess(termi, process,
157         activity)
158         SequenceFlow transition = new SequenceFlow
159         transition.from = newActivity
160         transition.to = xorExitGateway
161         process.add(transition)
162         SequenceFlow transition = new SequenceFlow
163         transition.from = xorExitGateway
164         transition.to = startof(term)
165         process.add(transition)
166         activity = xorExitGateway
167     End If
168 End If
169 End For
170 return activity

```
