MODELLING, ANALYSIS, AND OPTIMIZATION OF BPMN PROCESSES

GWEN SALAÜN

JOINT WORK WITH
FRANCISCO DURÁN, YLIÈS FALCONE, CAMILO ROCHA AND AHANG ZUO
“A BUSINESS PROCESS IS A COLLECTION OF RELATED, STRUCTURED ACTIVITIES OR TASKS BY PEOPLE OR EQUIPMENT IN WHICH A SPECIFIC SEQUENCE PRODUCES A SERVICE OR PRODUCT FOR A PARTICULAR CUSTOMER” (WIKIPEDIA)
MODELLING PROCESSES WITH BPMN

BPMN 2.0 (BUSINESS PROCESS MODELLING NOTATION) WAS PUBLISHED AS AN ISO STANDARD IN 2013
EXTENDED BPMN

BPMN EXTENDED WITH QUANTITATIVE INFORMATION: DURATIONS, RESOURCES, COSTS, POWER CONSUMPTION, CO2 EMISSION, ETC. (NOT MANDATORILY ALL AT THE SAME TIME)
MOTIVATING EXAMPLE

- How long does it take to one instance to complete in average?
- How much are each (type of) resource busy?
- What is the cost of this execution?
- What is the optimal number of resources for optimizing execution time and/or cost?
CONTRIBUTIONS

- An extension of BPMN for making explicit description of time & resources
- Automated analysis techniques for computing metrics on resource usage, execution time and cost
- Presentation of two possible approaches: design time vs. runtime analysis
- Alternative approach for optimizing processes: refactoring of BPMN processes
OUTLINE

1. BPMN WITH TIME AND RESOURCES
2. DESIGN TIME ANALYSIS
3. RUNTIME ANALYSIS
4. AUTOMATED REFACTORYING
5. TOOL SUPPORT
6. CONCLUDING REMARKS
SEVERAL MODELLING AND DEVELOPMENT FRAMEWORKS: ACTIVITI, BONITA, CAMUNDA, SIGNAVIO, …
CONTROL FLOWS AND GATEWAYS

- sequence flow
- start state
- end state
- exclusive gateway
- inclusive gateway
- parallel gateway
- event-based gateway

Split pattern:
- Task 1
- Task 2
- Task 3

Merge pattern:
- Task 1
- Task 2
- Task 3
SEMANTICS

Exclusive gateway (similar for Event-based gateway): split (left) and merge (right)

Parallel gateway: split (left) and merge (right)

Inclusive gateway: split (left) and merge (right)
IN THIS WORK, WE SUPPORT
- ACTIVITY DIAGRAMS
- COLLABORATION DIAGRAMS

EXTENDED BPMN

![Extended BPMN Diagram]
EXAMPLE: PRODUCT ORDER AND DELIVERY
EXAMPLE: VISA APPLICATION PROCESS

- Apply online
- Upload scanned passport
- Pay for fees
- Check file size
- Size too large
- Size OK
- Check quality
- Quality too low
- Quality OK
- Evaluate application
- Notify accept
- Notify reject
- Deliver visa
OUTLINE

1. BPMN WITH TIME AND RESOURCES
2. DESIGN TIME ANALYSIS
3. RUNTIME ANALYSIS
4. AUTOMATED REFACTORING
5. TOOL SUPPORT
6. CONCLUDING REMARKS
FORMAL MODEL

FORMAL SEMANTICS IS GIVEN TO THIS EXTENDED SUBSET OF BPMN BY ENCODING INTO REWRITING LOGIC

The Maude specification consists of three parts:

1. **PROCESS SYNTAX** is represented as an object with a set of flows and a set of nodes as attribute.
2. The **SIMULATION OBJECT** keeps information on the execution of the process.
3. **REWRITING RULES** represent how tokens evolve through the process and events are fired.
AUTOMATED ANALYSIS OF PROPERTIES

- Simulation Techniques take as input: A process description, a specification of resources and a workload (number of instances).

- Timing properties:
  - Average execution times (AET) of a process execution + shortest / longest execution.
  - Average synchronization time for merge gateways.

- Resource-based properties:
  - The global time usage ($GTU_R$) of all replicas of each resource $R$.
  - The GTU per replica of resource $R$ ($GTU^1_R$).
  - The average usage percentage for each resource $R$ over the global execution time ($UP^1_R$).

- Total cost.
**EXAMPLE: SIMULATION RESULTS**

<table>
<thead>
<tr>
<th>Numb. inst.</th>
<th>AET</th>
<th>Var</th>
<th>AST(_g)</th>
<th>AST(_e)</th>
<th>Total time</th>
<th>Resources</th>
<th>Analysis time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>GT(_U_e)</td>
<td>GT(_U_e)</td>
</tr>
<tr>
<td>100</td>
<td>106</td>
<td>72</td>
<td>58</td>
<td>58</td>
<td>326</td>
<td>271</td>
<td>135</td>
</tr>
<tr>
<td>200</td>
<td>185</td>
<td>134</td>
<td>71</td>
<td>139</td>
<td>670</td>
<td>514</td>
<td>257</td>
</tr>
<tr>
<td>400</td>
<td>284</td>
<td>173</td>
<td>98</td>
<td>237</td>
<td>1132</td>
<td>994</td>
<td>497</td>
</tr>
<tr>
<td>800</td>
<td>506</td>
<td>294</td>
<td>145</td>
<td>459</td>
<td>2217</td>
<td>1867</td>
<td>933.6</td>
</tr>
<tr>
<td>1600</td>
<td>891</td>
<td>473</td>
<td>240</td>
<td>844</td>
<td>4187</td>
<td>3714</td>
<td>1857</td>
</tr>
</tbody>
</table>

**Table 1.** Experimental results for the running example (2 employees, 3 drones)
EXAMPLE: AVERAGE EXECUTION TIME
EXAMPLE: AVERAGE USAGE (EMPLOYEE)
EXAMPLE: AVERAGE USAGE (DRONE)
OPTIMAL SOLUTIONS

COMPUTATION OF THE OPTIMAL ALLOCATION USING MULTI-OBJECTIVE OPTIMIZATION TECHNIQUES

\[
\min_{x \in X} \sum_{i \in \{c, t\}} w_i f_i(x)
\]

WHERE \( F_C \) AND \( F_T \) REPRESENT DRONE/EMPLOYEE COST AND AVERAGE EXECUTION TIME

DRONE COST = 20€  
EMPLOYEE COST = 50€
\( w_c = 0.4 \)
\( w_t = 0.6 \)

GOAL: MINIMIZE DELIVERY TIME

OPTIMAL SOLUTION:  
6 DRONES AND 2 EMPLOYEES
1. BPMN WITH TIME AND RESOURCES
2. DESIGN TIME ANALYSIS
3. RUNTIME ANALYSIS
4. AUTOMATED REFACTORIZING
5. TOOL SUPPORT
6. CONCLUDING REMARKS
NO RESTRICTION ON
THE BPMN SYNTAX
COMPUTATION OF PROPERTIES

Look-ahead Time

(i)th window

(i+1)th window

Task

\( R \)

Look-ahead Time

Time between Checks

Checkpoint

Checkpoint

Time

Compute the properties for each window (e.g., resource usage or AET)
DYNAMIC RESOURCE ALLOCATION

Each resource $R$ is defined with a minimum / maximum usage:

$U(R) \in [\minValue, \maxValue]$  

For each new window, we check:

- If $U_{\text{current}}(R) < \minValue$, then $N(R) = N(R) - n$.
- If $U_{\text{current}}(R) > \maxValue$, then $N(R) = N(R) + n$.
- Otherwise, no operation is performed.

**Example of the approach**

$U(R) = [70\%, 90\%]$  

$N(R_1) = 2, n = 1$  

$N(R_1) = 2 + 1 = 3$  

$U_{\text{current}}(R_1) = 93\%$
EXAMPLE: GOODS DELIVERY PROCESS
EXAMPLE: EXPERIMENTAL RESULTS

NUMBER OF REPLICA

RESOURCE USAGE

AVERAGE EXECUTION TIME
1. BPMN WITH TIME AND RESOURCES
2. DESIGN TIME ANALYSIS
3. RUNTIME ANALYSIS
4. AUTOMATED REFACTORING
5. TOOL SUPPORT
6. CONCLUDING REMARKS
MOTIVATIONS

- Adjusting the number of resources does not always induce process optimization.

- Solution: We propose refactoring techniques that automatically change the structure of the given input process with the final goal of reducing the total execution time of the process.

- Main idea: Add parallelism as often as possible.
BPMN SYNTAX

Initial event  End event  Strong flow  Weak flow  Timed flow

Untimed task

Timed task

Task with resource

Exclusive and parallel gateways (split and merge)

Resource R
SIMULATION AND ANALYSIS

- **Simulation Techniques**: Execute the process a certain number of times (parameter) and log data about this multiple execution.

- **Analysis**: Looks for specific moments during the simulation at which a task is still waiting to execute, and all resources required for executing this task are available.

- This means that this specific task could execute earlier in the process, and this information is used to change the structure of the process.
EXPLORATION OF RESULTS

- The whole approach applies by successive iterations, and stops when the queue of processes to be explored is empty.

- This approach may take time, because there may be many processes to be explored, therefore, some strategy is required to guide the exploration of new processes.

  - The first strategy applies refactoring for the task closer to the initial event in the process by trying to move first the task closer to its final position.

  - The second strategy is an exhaustive exploration of all processes to be explored up to a certain bound.
REFACTORING PATTERNS

- Given as input a process and a task that has to be moved earlier in the process, refactoring returns a new process as output.

- The refactoring pattern to be applied depends on what type of node precedes the task to be moved backwards, which can be:
  - Another task
  - A merge gateway (parallel / exclusive)
  - A split gateway (parallel / exclusive)

- This refactoring step focuses on the process structure and on the usage of resources by tasks, but does not take into account strong flows.
REFACTORING PATTERNS: TASK

IF THE TASK IS PRECEDED BY ANOTHER TASK, AND IF THEY **DO NOT SHARE ANY RESOURCES**, WE TRANSFORM THE PROCESS TO EXECUTE THESE TWO TASKS WITHIN A COMMON PARALLEL GATEWAY.
IF THE TASK ‘T’ IS PRECEDED BY A MERGE PARALLEL GATEWAY, IF THAT MERGE IS PRECEDED BY A SET OF TASKS, AND NONE OF THESE TASKS SHARE RESOURCES WITH ‘T’, THEN ALL TASKS ARE GATHERED IN PARALLEL BEFORE THE MERGE GATEWAY
REF. PATTERNS: MERGE PARALLEL GATEWAY (2/2)

- MERGE PARALLEL GATEWAY WITH PRECEDING TASKS (SHARED RESOURCES WITH ONE TASK)

- MERGE PARALLEL GATEWAY WITH PRECEDING TASKS (SHARED RESOURCES WITH SEVERAL TASKS)
REF. PATTERNS: MERGE EXCLUSIVE GATEWAY

- MERGE EXCLUSIVE GATEWAY WITH PRECEDING TASKS (T1 SHARES RESOURCES WITH T, T2 DOES NOT SHARE RESOURCES WITH T)

- MORE PATTERNS: CASCADING MERGES, SPLIT PARALLEL GATEWAY, SPLIT EXCLUSIVE GATEWAY, ...

![Diagram of a process with merging and exclusive gates]
EXAMPLE: INITIAL PROCESS

TRIP ORGANIZATION PROCESS

EXECUTION TIME: 43 DAYS
EXAMPLE: REFACTORED PROCESS

TRIP ORGANIZATION PROCESS

EXECUTION TIME: 28 DAYS
# Experiments

<table>
<thead>
<tr>
<th>BPMN Proc.</th>
<th>Tasks</th>
<th>Flows</th>
<th>Gateways</th>
<th>SF</th>
<th>AET&lt;sub&gt;c&lt;/sub&gt;</th>
<th>AET&lt;sub&gt;b&lt;/sub&gt;</th>
<th>Bounded Explo. AET</th>
<th>Time</th>
<th>Heuristic AET</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>13</td>
<td>4+2</td>
<td>0</td>
<td>70</td>
<td>50</td>
<td>50</td>
<td>1s</td>
<td>50</td>
<td>1s</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>15</td>
<td>6</td>
<td>0</td>
<td>40</td>
<td>25</td>
<td>25</td>
<td>2s</td>
<td>25</td>
<td>2s</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>13</td>
<td>2+2</td>
<td>0</td>
<td>19</td>
<td>6</td>
<td>6</td>
<td>2s</td>
<td>9</td>
<td>1s</td>
</tr>
<tr>
<td>4</td>
<td>8</td>
<td>12</td>
<td>2</td>
<td>4</td>
<td>43</td>
<td>28</td>
<td>28</td>
<td>21s</td>
<td>35</td>
<td>1s</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>12</td>
<td>2</td>
<td>2</td>
<td>21</td>
<td>14</td>
<td>14</td>
<td>23s</td>
<td>16</td>
<td>1s</td>
</tr>
<tr>
<td>6</td>
<td>9</td>
<td>10</td>
<td>0</td>
<td>6</td>
<td>90</td>
<td>50</td>
<td>50</td>
<td>31s</td>
<td>90</td>
<td>1s</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>20</td>
<td>6</td>
<td>0</td>
<td>24</td>
<td>18</td>
<td>19</td>
<td>38s</td>
<td>20</td>
<td>1s</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>20</td>
<td>2+4</td>
<td>3</td>
<td>23</td>
<td>13</td>
<td>13</td>
<td>67s</td>
<td>17</td>
<td>1s</td>
</tr>
<tr>
<td>9</td>
<td>12</td>
<td>29</td>
<td>8</td>
<td>0</td>
<td>200</td>
<td>120</td>
<td>140</td>
<td>50s</td>
<td>120</td>
<td>1s</td>
</tr>
<tr>
<td>10</td>
<td>15</td>
<td>36</td>
<td>10</td>
<td>0</td>
<td>260</td>
<td>180</td>
<td>220</td>
<td>103s</td>
<td>260</td>
<td>3s</td>
</tr>
</tbody>
</table>
OUTLINE

1. BPMN WITH TIME AND RESOURCES
2. DESIGN TIME ANALYSIS
3. RUNTIME ANALYSIS
4. AUTOMATED REFACTORYING
5. TOOL SUPPORT
6. CONCLUDING REMARKS
TOOL SUPPORT

- **DESIGN TIME ANALYSIS**
  - ORIGINAL VERSION BASED ON THE MAUDE FRAMEWORK (~4,000 LOC)
  - RECENT VERSION WRITTEN IN JAVA, SPRING BOOT, POSTGRESQL, JAVASCRIPT, REACTJS, NODE.JS (~10,000 LOC)

- **RUNTIME ANALYSIS**: EXTENSION OF THE ACTIVITI FRAMEWORK + JAVA CODE (~4,000 LOC)

- **REFACTORING TOOL** IMPLEMENTED IN PYTHON (~5,000 LOC) + USE OF VBPMN FOR TRANSFORMATION FROM PYTHON TO BPMN
1. BPMN WITH TIME AND RESOURCES
2. DESIGN TIME ANALYSIS
3. RUNTIME ANALYSIS
4. AUTOMATED REFACTURING
5. TOOL SUPPORT
6. CONCLUDING REMARKS
CONCLUDING REMARKS

- Several solutions have been presented for the optimization of BPMN processes with time and resources.

- The first one focuses on the problem of allocation of resources and explored two complementary options (design time vs runtime):
  - Design time approach is useful before deployment but static allocation of resources is not optimal for unstable workloads.
  - Runtime approach dynamically updates the number of resources but this is not always possible (employees).

- Refactoring techniques change a process into a new one, whose execution time is lower than the original one.
PERSPECTIVES

- DESIGN TIME / RUNTIME APPROACHES: THE MODEL OF RESOURCES NEEDS TO BE MORE ACCURATE (E.G., AN EMPLOYEE NEEDS BREAKS AND HOLIDAYS, A RESOURCE CAN BE EMPTY, ETC.)

- RUNTIME APPROACH: DEVELOP NEW DYNAMIC RESOURCE ALLOCATION STRATEGIES BASED ON AI PREDICTION ANALYTICS

- REFACTORING TECHNIQUES: INVESTIGATE OTHER STRATEGIES TO COMPUTE FASTER THE OPTIMAL PROCESS

- SYNTHESIS TECHNIQUES TO GENERATE AN OPTIMAL VERSION OF A PROCESS
SELECTED PUBLICATIONS

FRANCISCO DURÁN, YLIÈS FALCONE, CAMILO ROCHA, GWEN SALAÜN, AHANG ZUO: FROM STATIC TO DYNAMIC ANALYSIS AND ALLOCATION OF RESOURCES FOR BPMN PROCESSES. WRLA 2022.


FRANCISCO DURÁN, CAMILO ROCHA, GWEN SALAÜN: A REWRITING LOGIC APPROACH TO RESOURCE ALLOCATION ANALYSIS IN BUSINESS PROCESS MODELS. SCI. COMPUT. PROGRAM. 183, 2019.