Composition/Decomposition in Event-B

Renato Silva (University of Southampton)
Carine Pascal (Systerel)
T. S. Hoang (ETH Zurich)
Michael Butler (University of Southampton)

www.deploy-project.eu
www.event-b.org
Outline

• Event-B
• Motivation
• Shared Event Approach: Composition/Decomposition
• Shared Variable Approach: Composition/Decomposition
• Contributions
• Tool
• Conclusions/Future work
Event-B

Event-B: formal methodology allowing the specification of systems (inspired by classical B, Z and actions systems)

- Mathematical techniques based on set theory
- First order logic

Structured in two parts:

- **Context**: Static Part
- **Machine**: Dynamic part
  - Event \texttt{evt}: ANY parameter
    - WHEN guard
    - THEN action
Top down development:

- **Refinement of machines**: proof obligations (POs), gluing invariant
- **Extension of context**
Motivation

• “Top-down” development style: new events and data-refinement of variables during refinements

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Variables</th>
<th>Events</th>
</tr>
</thead>
<tbody>
<tr>
<td>Door_M0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Door_M1</td>
<td>9</td>
<td>12</td>
</tr>
<tr>
<td>Door_M2</td>
<td>10</td>
<td>18</td>
</tr>
<tr>
<td>Door_M3</td>
<td>10</td>
<td>18</td>
</tr>
</tbody>
</table>
Motivation (cont.)

• “Top-down” development style: new events and data-refinement of variables during refinements

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Variables</th>
<th>Events</th>
<th>POs: Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>MetroSystem_M0</td>
<td>7</td>
<td>10</td>
<td>43</td>
</tr>
<tr>
<td>MetroSystem_M1</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>MetroSystem_M2</td>
<td>12</td>
<td>16</td>
<td>73</td>
</tr>
<tr>
<td>MetroSystem_M3</td>
<td>12</td>
<td>16</td>
<td>21</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Variables</th>
<th>Events</th>
<th>POs: Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train_M0</td>
<td>7</td>
<td>13</td>
<td>15</td>
</tr>
<tr>
<td>Train_M1</td>
<td>9</td>
<td>15</td>
<td>64</td>
</tr>
<tr>
<td>Train_M2</td>
<td>13</td>
<td>20</td>
<td>148</td>
</tr>
<tr>
<td>Train_M3</td>
<td>12</td>
<td>20</td>
<td>48</td>
</tr>
<tr>
<td>Train_M4</td>
<td>13</td>
<td>22</td>
<td>102</td>
</tr>
</tbody>
</table>

• **Problem**: increasing complexity of the refinement process when having to deal with many events and many state variables (and likely many POs)
Motivation (cont.)

- Possible solution: **Decomposition**
  - Splitting a large model into smaller sub-components
  - Design/architectural decision
  - Team development
  - Alleviate the complexity of discharging POs
Shared Event Approach

• Theoretical work developed by Butler
• Events are synchronised through shared parameters they can communicate
• Restriction: no shared variables
• Events synchronisation:
  • Parameters are merged
  • Guards are conjoined
  • Actions are executed in parallel
Shared Event Composition

Machine M1

\[
\begin{align*}
\text{e1} & \quad \text{e2} \\
\downarrow & \quad \downarrow \\
v1 & \quad v1
\end{align*}
\]

\[
\begin{align*}
e2 & \triangleq \text{ANY } p \\
\text{WHERE } p & \in \mathbb{N} \land v1 = 0 \\
\text{THEN } v1 & := p
\end{align*}
\]

Machine M2

\[
\begin{align*}
\text{e3} & \quad \text{e4} & \quad \text{e5} \\
\downarrow & \quad \downarrow & \quad \downarrow \\
v2 & \quad v2 & \quad v3
\end{align*}
\]

\[
\begin{align*}
e3 & \triangleq \text{ANY } p \\
\text{WHERE } v2 & = p \\
\text{THEN } v2 & := 0
\end{align*}
\]

Machine M

\[
\begin{align*}
\text{e1} & \quad \text{e2//e3} & \quad \text{e4} & \quad \text{e5} \\
\downarrow & \quad \downarrow & \quad \downarrow & \quad \downarrow \\
v1 & \quad v2 & \quad v2 & \quad v3
\end{align*}
\]

\[
\begin{align*}
e2 \parallel e3 & \triangleq \text{ANY } p \\
\text{WHERE } p & \in \mathbb{N} \land v1 = 0 \land v2 = p \\
\text{THEN } v2 & := 0 \parallel v1 := p
\end{align*}
\]
Shared Event Decomposition

- Variables of M are first partitioned into M1, ..., Mn.
- Event partition is a consequence of the variable splitting.
- If an event shares variables belonging to different sub-models, that event (parameters, guards, actions) is split over the sub-models (preparation step might be required to separate variables).

Interaction between sub-components: synchronized events communicate via shared parameters.

\[ e_2 \equiv \text{ANY } p \]
\[ \text{WHERE } p \in \mathbb{N} \land v_1 = 0 \land v_2 = p \]
\[ \text{THEN } v_2 := 0 \parallel v_1 := p \]

\[ e_{2.1} \equiv \text{ANY } p \]
\[ \text{WHERE } p \in \mathbb{N} \land v_1 = 0 \]
\[ \text{THEN } v_1 := p \]

\[ e_{2.2} \equiv \text{ANY } p \]
\[ \text{WHERE } v_2 = p \]
\[ \text{THEN } v_2 := 0 \]
Shared Variable Approach

• Theoretical work developed by Abrial
• Variable sharing allowing the communication between sub-components

• External event: simulates the way the shared variables are handled in the non-decomposed machine. Correspond to the rely condition in a rely/guarantee view
Shared Variable decomposition

- Events of M are first partitioned into M1, M2, ..., Mn.

- Variable partition is a consequence of the event splitting.

Shared variable: variable accessed by events of distinct sub-machines (in opposition to private variable).

External event: built from an event of the non-decomposed machine and. Correspond to the rely condition in a rely/guarantee view

\[ e_2 \triangleq \text{ANY } p \]
\[ \text{WHERE } p \in \mathbb{N} \land v_1 = 0 \land v_2 = p \land v_2 \neq 0 \]
\[ \text{THEN } v_2 := 0 \parallel v_1 := p \]

\[ e_{2\text{-ext}} \triangleq \text{ANY } p, p \cdot v_1 \]
\[ \text{WHERE } p \in \mathbb{N} \land p \cdot v_1 = 0 \land v_2 = p \land v_2 \neq 0 \]
\[ \text{THEN } v_2 := 0 \]
Shared Event/Shared Variable Decomposition

- Decomposition is semantic (in opposition to syntactic)
- The composition of the sub-components do refine the abstract component
  - Individual sub-components do not necessarily refine the abstract component
  - Properties of the abstract component may not hold in the individual sub-components but they hold in the composed sub-components (refinement of abstract)
- A sub-component can be refined independently from the other sub-components
Contributions

• Based on the two styles described, develop the method and tool support for decomposition in Event-B.

• Which decomposition to choose?
  • It depends on the system being developed and on the user’s preference.
  • Shared Variable:
    • Variables being shared suggest use in parallel applications
  • Shared Event:
    • Suggests the use in message passing distributed systems or middleware systems.
Tool

- A single plug-in for both styles in the Rodin platform.
- Usage of extension mechanisms: wizard, menu, editor, static checker.
- Decomposition wizard:
Conclusion

• **Decomposition**: used to decrease the complexity and increase the modularity of large systems

• **Main benefits**:
  - Further refinement of independent sub-models in parallel (monotonicity).
  - Allow team development for each sub-model (attractive option for the industry)
  - Distribution of POs
• Decomposition in Event-B:
  • **Shared Variable** approach: seems more suitable for modelling parallel systems*
  • **Shared Event** approach: seems more suitable for modelling messaging-passing distributed systems**
  • **Decomposition tool** is available since the release 1.2 of the Rodin platform.
  • Already used in several case studies with positive feedback 😊

Future work

- **Visual perspective of decomposition**
  - It seems easier to partition a system by visualising how to allocate the elements in the sub-components
  - Option: using Graphical Modelling Framework (GMF)
  - Integration of the decomposition plug-in with other plug-ins (compatibility problems): Records, Modularisation plug-in, etc
Future work (cont.)

- Application of more complex case studies and analyse the results (Ongoing work: decomposition of a metro system)
Questions?

Thank you