Exposing and Controlling Emergent Behaviors in a System of Systems (SoS) Model

An Overview for the INCOSE North Texas Chapter

Kristin Giammarco, Ph.D.
NPS Department of Systems Engineering
10 September 2019

This work was made possible in part by sponsorship from NAVAIR.
• Introduced this audience to Monterey Phoenix (MP), a Navy-developed lightweight formal methods framework for behavior modeling

• Presented use cases for MP, in particular detecting, classifying, predicting and controlling emergent behaviors

• Presented examples of both expected and unexpected emergent behaviors arising from three different MP models
Tonight’s Agenda

• Provide motivation & MP overview

• Show how to segment and extend a SysML activity model for emergent behavior analysis using MP

• Present, discuss and analyze examples of emergent behaviors found in the extended model

• Show how emergent behaviors may be classified as weak, strong, positive or negative.

• Conclude with some key takeaways and future work
Motivation

• SysML models are being developed in the Navy as, among other things, a basis for proposals from solution developers

• SysML models that are incomplete / incorrect could lead to requirements errors

• Complex system designs may permit “extra” unwanted system behaviors – how to we predict / expose these?

• This research developed methods and tools to help steer and shape behavioral design
  – to meet requirements (verification)
  – to meet expectations (validation)
What is Monterey Phoenix?

- Navy-developed lightweight formal methods framework for modeling human, technology, and environment behaviors.

- *Behavior* is defined as a set of events with two basic relations: precedence and inclusion.

- Generates sets of behavior scenarios that are exhaustive up to a user-defined scope (number of iterations).
MP-Firebird Layout

- Run button
- https://firebird.nps.edu
- Scope of execution
- Trace window
- Number of traces
- Console window
- Code window
- All traces

Scope of execution:

```plaintext
/*
  Example1_simple_message_flow.m
  Event grammar rules for each root define derivations for event traces,
  in this case a simple sequence of zero or more events for each root.
  The COORDINATE composition takes two root traces and produces
  a modified event trace, merging behaviors of Sender and Receiver
  and adding the PRECEDES relation for the selected send/receive pairs.
  The coordination operation behaves as a "cross-cutting" derivation rule.
  Run for scopes 1 and up. The "Sequence" or "Swim Lanes" layouts are
  the most appropriate for browsing traces here.
*/

SCHEMA simple_message_flow

ROOT Sender: (* send *);
ROOT Receiver: (* receive *);
COORDINATE $x: send FROM Sender,
               $y: receive FROM Receiver
       DO ADD $x PRECEDES $y; OD;
```

Trace window:

- Completed Sender: 4 traces (0 MARKed) 10 events average 2.5 ev/trace min 1 max 4
- Completed Receiver: 4 traces (0 MARKed) 10 events average 2.5 ev/trace min 1 max 4
- Completed Sender/Receiver: 4 traces (0 MARKed) 24 events average 6 ev/trace min 3 max 9
- Elapsed time 0 sec, Speed: inf events/sec

Finished Compiling! Graphing 4 event traces...
Non-Combat Operations Scenario 1

- Fourteen (14) Actors / Swim Lanes
- Four (4) Phases
- Fifty-four (54) activities
- Zero (0) alternative behaviors
  - shows baseline desired scenario
• **Convert** SysML model into MP model
• **Segment** the model into phases
• **Elaborate** each phase model with alternatives
• **Generate** exhaustive set of traces for each phase
• **Inspect** for incorrect or unintended behaviors
Convert SysML activity model into logically equivalent MP model

Non-Combat Operations Scenario 1
Segment the model into phases

Non-Combat Operations Scenario 1
Elaborate each phase model with alternatives & generate scenarios

1. Phase 1 – Prepare/Configure
2. Phase 2 – Take Off
3. Phase 3 – Transit/Navigate
4. Phase 4 – Post Mission Task

Non-Combat Operations Scenario 1

1 scenario
### Inspect for incorrect or unintended behaviors

**Phase 3**

**Far left:**
Baseline scenario; vessel located and payload on target.

**Middle left:**
Vessel located but payload missed target.

**Middle right:**
AV needs to return before vessel is located.

**Far right:**
Vessel not found but AV drops payload.

<table>
<thead>
<tr>
<th>2</th>
<th>3</th>
<th>4</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Air Vehicle</strong></td>
<td><strong>Air Vehicle</strong></td>
<td><strong>Air Vehicle</strong></td>
<td><strong>Air Vehicle</strong></td>
</tr>
<tr>
<td>Start GPS Navigation</td>
<td>Transit to Mission Location</td>
<td>Transit to Mission Location</td>
<td>Start GPS Navigation</td>
</tr>
<tr>
<td>Transit to Mission Location</td>
<td>Detect Beacon</td>
<td>Detect Beacon</td>
<td>Transit to Mission Location</td>
</tr>
<tr>
<td>Detect Beacon</td>
<td>Locate Vessel</td>
<td>Locate Vessel</td>
<td>Detect Beacon</td>
</tr>
<tr>
<td>Locate Vessel</td>
<td>Holds over Vessel</td>
<td>Vessel Not Found</td>
<td>Transmit Video</td>
</tr>
<tr>
<td>Holds over Vessel</td>
<td>Transmit Video</td>
<td>Transmit Video</td>
<td>Receive Command</td>
</tr>
<tr>
<td>Transmit Video</td>
<td>Receive Command</td>
<td>Return to Hold Point</td>
<td>Shut Down</td>
</tr>
<tr>
<td>Receive Command</td>
<td>Drop Payload</td>
<td>Payload Misses Target</td>
<td>Payload Misses Target</td>
</tr>
<tr>
<td>Drop Payload</td>
<td>Payload on Target</td>
<td>Return to Hold Point</td>
<td>Return to Hold Point</td>
</tr>
<tr>
<td>Payload on Target</td>
<td>Return to Hold Point</td>
<td>Shut Down</td>
<td>Shut Down</td>
</tr>
</tbody>
</table>
What should happen if the payload just misses the target?

Could the payload still be retrieved by target vessel?
Could this scenario really happen?

Under what circumstances might this be negative behavior or positive behavior?

Though unintended, does trace 6 contain an idea for handling out of range vessels or AVs experiencing a return to base condition?
**Detection**: Initial discovery of emergent behavior.

**Classification**:  
- **Simple**: derived from element properties and relationships in non-complex or ‘ordered’ systems [5].  
- **Weak**: desired (or at least allowed) emergence produced by a complex system [5].  
- **Strong**: unexpected emergence not observed until simulation, testing, or operations [6].

**Prediction**: Postulation of potential future states of emergence based on detected behaviors.

**Control**: Management of positive or negative emergent behaviors through M&S or other analysis.

Definition set paraphrased from [4]
<table>
<thead>
<tr>
<th>Trace</th>
<th>Detected Behavior</th>
<th>Predicted Behavior</th>
<th>Classification</th>
<th>Control Strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Vessel located and payload on target.</td>
<td>Mission success - The payload meets the target and the patient is able to use the medication.</td>
<td>Weak Positive Emergence</td>
<td>Valid possible outcome (baseline scenario). Clarify the assumed outcome that the patient is able to use the medication.</td>
</tr>
<tr>
<td>3</td>
<td>Vessel located but payload missed target.</td>
<td>Mission failure - The payload misses the target and the patient falls into a diabetic coma.</td>
<td>Weak Negative Emergence</td>
<td>Valid possible outcome. Clarify the assumed outcome that the patient falls into a diabetic coma.</td>
</tr>
<tr>
<td>4</td>
<td>AV needs to return before vessel is located.</td>
<td>Mission failure - The AV detects the emergency beacon, but has to return before it can locate the vessel.</td>
<td>Weak Negative Emergence</td>
<td>Valid possible outcome. No further control recommended.</td>
</tr>
<tr>
<td>6</td>
<td>Vessel not found but AV drops payload anyway.</td>
<td>Mission failure - The payload is dropped into the ocean without knowing the location of the vessel. Either the system experienced a malfunction, or the command to drop the payload was sent too soon.</td>
<td>Strong Negative Emergence</td>
<td>Add new event System_malfunction as alternative to Receive_command in Air Vehicle root event. Downgrade to Weak Negative Emergence.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Mission success - The payload is intentionally dropped without video on the vessel and it is ultimately received by the vessel. The AV Operator may know from another source (such as the beacon) that the vessel is close by, or the payload may be equipped to close the remaining distance so that the AV has the range necessary for its return trip.</td>
<td>Strong Positive Emergence</td>
<td>Add new events to the model to clarify the specifics, assumed outcome, and associated new requirements. Downgrade to Weak Positive Emergence.</td>
</tr>
</tbody>
</table>
• Operational “what ifs” were exposed through MP modeling of the provided baseline scenario.

• The MP model exposed some unexpected and unwanted behaviors, leading to discovery of requirements.

• MP modeling of SysML behavior diagrams can help to expose requirements that may otherwise not be considered until later in the lifecycle.
Future Work

• Automate model transformation between SysML and MP
  – MP version 4 can now generate many SysML -style diagrams
  – MP version 5 will synthesize MP models from a representative set of use cases

• Train model developers how to verify and validate contents of SysML models using MP
RT-176 Reports and Models:
https://sercuarc.org/project/?id=35&project=Verification+and+Validation+%28V%26V%29+of+System+Behavior+Specifications

Monterey Phoenix and Related Work:
https://wiki.nps.edu/display/mp
https://4.firebird.nps.edu

Kristin Giammarco: kmgiamma (at) nps.edu
References


