RT-176: Verification and Validation (V&V) of System Behavior Specifications

Sponsor: DASD(SE) and NAVAIR

By
Dr. Kristin Giammarco

9th Annual SERC Sponsor Research Review
November 8, 2017
FHI 360 CONFERENCE CENTER
1825 Connecticut Avenue NW
8th Floor
Washington, DC 20009

www.sercuarc.org
Project Title: V&V of System Behavior Specifications  

Principal Investigator: Kristin Giammarco, PhD (NPS)  
- Co-Investigator: Ron Carlson, CAPT USN (ret) (NPS)  
- Co-Investigator: Mark Blackburn, PhD (Stevens Institute)  

Duration: 22 Months (23 Nov 2016 – 30 Sep 2018)  
NPS Resources Applied: 1.62 FTE*  
- Year 1: 1.06 FTE  
- Year 2: 0.56 FTE  

Sponsors: DASD(SE) and NAVAIR  
- DASD(SE) - Mr. Scott Lucero  
- NAVAIR - Mr. Jaime Guerrero  

Objectives:  
- To conduct a model-based V&V demonstration  
- To develop and deliver training content on model-based methods  
- To provide coaching and mentoring to MBSE pilot projects  

Problem Statement:  
This project was established in response to a NAVAIR need for:  
- MBSE methods, processes, and tools for assessing system behavior specifications and other requirements earlier in the lifecycle of a system  
- NAVAIR workforce education in the use of automated tools for conducting early and continuous V&V across the entire lifecycle  

Deliverables:  
- Interim report and models 30 Sep 2017  
- Final report and models 30 Sep 2018  

Key Finding:  
Conducting model-based V&V on system and SoS behavior specifications can improve documented requirements and expose tacit expectations as additional requirements, prior to contract.  
(Twelve specific findings available in the interim report.)  

Key Recommendations:  
- Develop a body of evidence that demonstrates the efficacy of the new model-based V&V methods.  
- Deploy Monterey Phoenix (MP) to automate event trace generation and inspection for MBSE pilot projects.  
(Ten recommendations available in the interim report.)
Organization of the Interim Report

• Main Body
  — Motivation and Objectives
  — Related Work
  — Technical Accomplishments
  — Conclusions
  — FY18 Plans

• Appendix A: List of Publications and Invited Talks

• Appendix B: References Cited

• Appendix C: Collaborator Courses that Integrate or Contribute Research Results

• Appendix D: Monterey Phoenix Overview

• Appendix E: Preliminary Catalog of Reusable Architecture Patterns

• Appendix F: Instructions for Downloading UAV Models

• Appendix G: Model Based V&V (MCSE MPT) Demonstration
Model-based V&V Demo (Task Lead: Kristin Giammarco)

- **Formalize UAV behavior specifications into MBSE architecture tool(s)**
  - using Monterey Phoenix for comprehensive use case scenario generation

- **Demonstrate use of the UAV behavior model for early V&V analysis of requirements**
  - using MP to expose positive and negative system behaviors permitted by the design

- **Formalize patterns of common design flaws or other model properties**
  - using MP event grammar to store system behavior templates
Objectives 2 and 3

Training Content Development and Delivery (Task Lead: Ron Carlson)

- **Work on this task is planned in FY18.**

Coaching & Mentoring (Task Lead: Kristin Giammarco)

- **Create a catalog of typical architecture model views for behavior containing good practices (patterns\(^1\)), poor practices (anti-patterns\(^2\)), and pattern/anti-pattern examples.**

- **Provide ongoing mentoring and coaching support as needed on pilot projects.**

---

\(^1\) Tool-agnostic patterns with style guidance for certain critical and useful model views will be incorporated.

\(^2\) Discovered practices that should be avoided are often called anti-patterns since they illustrate examples contrary to good practice.
Research Approach

• Prepare source data / conduct pre-work
• Develop Design Reference Mission (DRM)
• Conduct MP behavior model development
• Conduct functional requirements behavior model-based V&V
  — Appendices D, G
• Reusable architecture patterns and anti-patterns
  — Appendix E
• Conduct nonfunctional / system-wide requirements behavior model-based V&V
• Transition the MP Analyzer tool to the workforce
Functional requirements behavior model-based V&V: Approach

1. Frame the problem

2a. Extract narrative of behavior

2b. Identify actors and actions

Humanitarian Assistance / Disaster Relief (HADR) Design Reference Mission (DRM)

Mission Narrative

Ingress Phase

Pre-Conditions: Ingress Phase begins when the GCS Operator receives a launch clearance from the JTF C2

- GCS Operator receives launch command from JTF C2
- GCS Operator checks launch parameters for safety
- GCS Operator receives launch clearance from host platform
- GCS Operator commands Ground Crew to launch UAV
- Ground Crew commands UAV to launch
- UAV launches
- UAV maneuvers to clear obstacles then maneuvers to reach ingress altitude
- UAV levels off at ingress altitude
- UAV transmits status (payload and systems) and position messages (on-going throughout each phase of mission)
- If status and position are acceptable, GCS Operator commands UAV to ingress flight path
- If status and position are acceptable, GCS Operator commands UAV to proceed to egress phase
- UAV follows ingress flight path to reach on-station area and altitude
- GCS Operator monitors UAV status and position during flight path to on-station area and altitude
- The UAV reaches initial on-station waypoint and reports position to GCS Operator

Post-Conditions: The Ingress Phase ends when the UAV reaches the initial on-station waypoint
Functional requirements behavior model-based V&V: Approach

Write as machine-readable pseudo-code:

ROOT UAV: Launch;

2b. Identify actors and actions

Ingress Phase
Pre-Conditions: Ingress Phase begins when the GCS Operator receives a launch clearance from the JTF C2

• GCS Operator receives launch command from JTF C2
• GCS Operator checks launch parameters for safety
• GCS Operator receives launch clearance from host platform
• GCS Operator commands Ground Crew to launch UAV
• Ground Crew commands UAV to launch
• UAV launches

• UAV maneuvers to clear obstacles then maneuvers to reach ingress altitude
• UAV levels off at ingress altitude
• UAV transmits status (payload and systems) and position messages to GCS Operator (on-going throughout each phase of mission)
• If status and position are acceptable, GCS Operator commands UAV to proceed on ingress flight path, otherwise the GCS Operator commands UAV to return to base. Proceed to egress phase.
• UAV follows ingress flight path to reach on-station area and altitude
• GCS Operator monitors UAV status and position during flight path to on-station area and altitude
• The UAV reaches initial on-station waypoint and reports position to GCS Operator

Post-Conditions: The Ingress Phase ends when the UAV reaches the initial on-station waypoint
3. Identify event coordination

ROOT JTF_C2: Provide_launch_command;
ROOT GCS_Operator: Receive_launch_command;

COORDINATE $a$: Provide_launch_command FROM JTF_C2,
$b$: Receive_launch_command FROM GCS_Operator
DO ADD $a$ PRECEDES $b$; OD;

Ingress Phase
Pre-Conditions: Ingress Phase begins when the GCS Operator receives a launch clearance from the JTF C2

- GCS Operator receives launch command from JTF C2
- GCS Operator checks launch parameters for safety
- GCS Operator receives launch clearance from host platform
- GCS Operator commands Ground Crew to launch UAV
- Ground Crew commands UAV to launch
- UAV launches
- UAV maneuvers to clear obstacles then maneuvers to reach ingress altitude
- UAV levels off at ingress altitude

Post-Conditions: The Ingress Phase ends when the UAV reaches the initial on-station waypoint

Proceed to egress phase.
- UAV follows ingress flight path to reach on-station area and altitude
- GCS Operator monitors UAV status and position during flight path to on-station area and altitude
- The UAV reaches initial on-station waypoint and reports position to GCS Operator
What is Monterey Phoenix?

• MP is a **Navy-developed** framework for modeling human, technology, and environment behaviors all in one framework

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

• An **event grammar** is used to express behavior compositions using some combination of:

  A: B C;  Ordered sequence of events (A includes B followed by C)
  A: (B | C); Alternative events (A includes B or C)
  A: [B]; Optional event (A includes B or no event at all)
  A: (* B *); Ordered sequence of zero or more occurrences of event B in A
  A: (+ B +); Ordered sequence of one or more occurrences of event B in A
  A: {B, C}; Unordered set of events B and C in A (B and C may happen concurrently)
  A: {* B *}; Unordered set of zero or more occurrences of event B in A
  A: {+ B +}; Unordered set of one or more occurrences of event B in A
The full UAV behavior model for Ingress:

```c
/*-----------------------------------------------
   UAV_MODEL
-----------------------------------------------*/

ROOT UAV:

UAV_Receive_navigation_reference
UAV_Receive_launch_command
Launch
Execute_climb
Maneuver_to_clear_obstacles
Maneuver_to_ingress_altitude
Level_off_at_ingress_altitude
Transmit_status_and_position

  ( Receive_command_to_proceed
    Follow_flight_path_to_reach_onstation_area_and_altitude
    Reach_onstation_waypoint |

    Receive_command_to_abort
    Abort_mission )

;```
Functional requirements behavior model-based V&V: Approach

MP Models separate the behaviors in different systems...

JTF C2

GCS Operator

Ground Crew

Environment

UAV

... from interactions among the systems.
The **Small Scope Hypothesis**: most flaws in models can be demonstrated on small counterexamples.

1. Type model here.
2. Run the model.
3. Inspect resulting event traces.

- Model system behaviors separately
- System interactions treated as constraints
- Exhaustive generation of SoS behaviors up to a specified scope
# Behavior Model V&V Activities

<table>
<thead>
<tr>
<th>Look for and address:</th>
<th>e.g.,</th>
</tr>
</thead>
<tbody>
<tr>
<td>Syntax errors</td>
<td>missing semicolons, misplaced parentheses</td>
</tr>
<tr>
<td>Typographical and transcription errors</td>
<td>misspelled event names, forgotten_underscores,</td>
</tr>
<tr>
<td></td>
<td>wrong event names used</td>
</tr>
<tr>
<td>Notational errors</td>
<td>deviations from required notation and style guides adopted by the organization</td>
</tr>
<tr>
<td>Scope errors</td>
<td>running only at scope 1 when errors need scope 2 or 3 to manifest</td>
</tr>
<tr>
<td><strong>Required behaviors that are missing</strong></td>
<td>none of the event traces show a certain behavior that was required</td>
</tr>
<tr>
<td><strong>Prohibited behaviors that are present</strong></td>
<td>some event traces show a certain behavior that was prohibited</td>
</tr>
<tr>
<td>Unspecified valid behaviors</td>
<td>some event traces show wanted behaviors that were not explicitly required</td>
</tr>
<tr>
<td><strong>Unspecified invalid behaviors</strong></td>
<td>some event traces show unwanted behaviors that were not explicitly prohibited</td>
</tr>
</tbody>
</table>
Required Behaviors that are Missing

Identification of missing required behaviors involves *cycling on requirement specification completeness*

6.1.3.3.14. Ingress Phase:

Pre-Conditions: Ingress Phase begins when the GCS Operator receives a launch clearance from the JTF C2

- GCS Operator receives launch command from JTF C2
- If shipboard, the GCS Operator requests host ship maneuver to achieve launch parameters
- GCS Operator checks launch parameters for safety
- GCS Operator receives launch clearance from host platform
- GCS Operator commands Ground Crew to launch UAV
- Ground Crew commands UAV to launch
- UAV launches
- UAV maneuvers to clear obstacles then maneuvers to reach ingress altitude
- UAV levels off at ingress altitude
- UAV transmits status (payload and systems) and position messages to GCS Operator (on-going throughout each phase of mission)
- If status and position are acceptable, GCS Operator commands UAV to proceed on ingress flight path, otherwise the GCS Operator commands UAV to return to base. Proceed to egress phase.
- UAV follows ingress flight path to reach on-station area and altitude
- GCS Operator monitors UAV status and position during flight path to on-station area and altitude
- The UAV reaches initial on-station waypoint and reports position to GCS Operator

Post-Conditions: The Ingress Phase ends when the UAV reaches the initial on-station waypoint

**Does GCS Operator confirm receipt?**

**Should there be a similar command to the host ship to prepare/maneuver to receive the UAV?**

**Can anyone else give the launch command?**

**Are there situations in which the UAV wouldn’t climb out?**
Prohibited Behaviors that are Present

- The larger a behavior model, the more automated tools are needed for verifying absence of known unwanted behaviors

Are there any scenarios in which the UAV is launched without permission from the JTF C2?

**Assertion Check:**

```
CHECK  #Launch
       #JTF_Provide_launch_command FROM UAV <=
       FROM JTF_C2
ONFAIL SAY("Unauthorized Launch");
```

**Formal Requirement:**

```
ENSURE #Launch
       #JTF_Provide_launch_command FROM UAV <=
       FROM JTF_C2;
```
Abort command could result from
- an overriding third party
- a problem observed through other means
- operator error
## Unspecified Invalid Behaviors
### Summary of Emergent Behaviors

<table>
<thead>
<tr>
<th>Example</th>
<th>Figure</th>
<th>Detection</th>
<th>Classification</th>
<th>Prediction</th>
<th>Control</th>
</tr>
</thead>
<tbody>
<tr>
<td>UAV Ingress</td>
<td>G.17 left</td>
<td>Automatic and scope-complete with MP</td>
<td>Strong positive emergence</td>
<td>The UAV gives the appearance of an acceptable status, but the operator commands the UAV to abort the mission.</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Strong negative emergence</td>
<td>The UAV’s status is acceptable but the operator erroneously commands the UAV to abort the mission.</td>
<td>Provide adequate rest and training for operators to reduce the likelihood of occurrence of this scenario</td>
</tr>
</tbody>
</table>
Research Approach

• Prepare source data / conduct pre-work
• Develop Design Reference Mission (DRM)
• Conduct MP behavior model development
• Conduct functional requirements behavior model-based V&V
  — Appendices D, G
• Reusable architecture patterns and anti-patterns
  — Appendix E
• Conduct nonfunctional / system-wide requirements behavior model-based V&V
• Transition the MP Analyzer tool to the workforce
# MP, SysML and LML Crosswalk Example

<table>
<thead>
<tr>
<th>Activity Diagram</th>
<th>Corresponding MP Code and Graph(s)</th>
</tr>
</thead>
</table>
| **SysML:**
  B or C. The “Or Action” can have custom named branches. | /* A includes: B or C */
| ![SysML Diagram](image) | `A: ( B | C );` |
| **LML:**
  B or C. The “Or Action” can be a named block phrased as a question (with Yes/No branches as in the example) or a decision point / event with custom named branches. | /* An alternate expression can be used to name the specific decision point branches as events in their own right. A includes: Or Action, followed by B if Yes or C if No, then Merge Action. */
| ![LML Diagram](image) | `A: Or Action ( Yes B | No C ) Merge Action;` |

/* Whether or not to include the action and branch names as separate events in MP can be based on user preference. */
General Anti-Patterns (Examples)

• Hierarchy
  — An activity having no children and no parent

• Functional/physical allocation
  — An activity that is not performed by any performer

• Functional interaction
  — An activity that produces some resource, but does not consume any resources
General Anti-Patterns (Examples)

• Physical interaction anti-patterns
  — A performer that does not connect to at least one connector

• Traceability
  — An activity that is not subject to any requirement

• Standardization
  — Two performers that exchange some resource through performed activities, but are not constrained by a common standard
Findings

• Five key concepts support the development of system and SoS behavior models
• Model developers can expose both wanted and unwanted behaviors without specialized skills
• Model developers can generate all use case scenarios automatically up to a scope limit
• All behaviors not explicitly suppressed will eventually emerge
• Model developers can detect, classify, predict and control emergent behaviors with MP
• Probabilities for behavior outcomes can be calculated
• Templates for behavior patterns can be derived inductively
• All possible interactions for a model can be summarized in matrix form
• System behaviors could be formally specified consistently by both government (in requirements) and contractors (in implementation)
• Many architecture modeling lessons learned are formalizable into anti-patterns
• Design Reference Missions (DRMs) are useful source data for model development
• MP may be mapped to SysML and other graphical notations
Recommendations

- Develop a body of evidence for the efficacy of new modeling methods
- Further test the Monterey Phoenix approach on MBSE pilot projects
- Train model developers how to verify and validate SysML models using MP
- Generate sequence diagrams automatically from other behavior views
- Scrub the requirements with MP before writing contracts
- Use the model’s ontology as a language to describe model anti-patterns
- Formalize the types and definitions of emergent behavior for use in risk analysis
- Provide modelers with good source materials for the development effort
- Develop a graphical gateway to MP
- Inform the workforce of the results of this research
RT-176 Interim Report and Models:  

Monterey Phoenix and Related Work:  
https://wiki.nps.edu/display/mpfirebird.nps.edu

Kristin Giammarco: kmgiamma (at) nps.edu
Acknowledgements

The research team would like to thank our research sponsor, NAVAIR Systems Engineering Transformation (SET), for being an early investor in and adopter of the Monterey Phoenix (MP) behavior modeling approach for V&V of system behavior specifications.

We would also like to acknowledge the contributions of the many NPS Master’s students who contributed content and insight to the RT-176 project, especially LT Stephan Mathos, Mr. Tom Moulds, Mr. Stephen (Alex) Rambikur, Mr. Ernie Lemmert, Mr. Cody Reese, LCDR Chris Krukowski, and Mr. Anthony Constable, each of whom made direct and specific contributions to the project deliverables.
BACKUP
Identification of missing required behaviors involves **checking that all required behaviors in the specification are in fact represented in the model**.

Since this event is coordinated with `Receive_command_to_proceed` in lines 97-99,

and `Receive_command_to_proceed` is inside a composition of alternative events,

these alternative events will always be rejected.
Required Behaviors that are Missing

To correct the issue, corresponding branches of coordinated behavior are established employing alternative events in the COORDINATE statement.

New alternative paths for Status_acceptable and Status_unacceptable are added…

…and coordinated with the preceding event Transmit_status_and_position …

…to provide two possible branches of coordinated behavior in both the GCS_Operator and the UAV.