Identifying Decision Patterns Using Monterey Phoenix

presented to
Office of the Deputy Assistant Secretary of Defense, Systems Engineering
System of Systems Engineering
Collaborators Information Exchange (SoSECIE)
21 August 2018
authored by
John J. Quartuccio
PhD candidate, Systems Engineering Department
jjquartu@nps.edu
Previous Publications

• This presentation is based on content previously published, as follows:


System of System Architectures readily capture the intended interactions within the context boundary

- UML/SysML outlines a means to document system behaviors (ref: https://www.omg.org/)
  - Activity diagrams
  - Sequence diagrams
  - State-space diagrams
  - Use-case diagrams

What happens when things go wrong?

- Identify a way to capture both the desired behaviors and the undesirable behaviors of systems?
Outline

• Identification of patterns
  – Topology
  – Semantics

• Behaviors and a proposed analysis method

• Decision model example
  – Narrative
  – Interactions
  – Constraints
  – Analysis
    • Probability of a trace
    • N-squared diagram of all traces

• Wrap up and discussion
Why Conduct Behavior Analysis?

• Logical analysis at a high level of abstraction
  – Derived from the essence of a behavior – hierarchical and temporal aspects of an interaction
  – Considers the fundamental interactions of the system, both internal and external, but described separately
  – Conducted prior to high cost investment in detailed design
    • Prior to detailed modeling of discrete event, agent-based, physics-based, or hybrid models
    • Prior to physical design and manufacture
  – Enables analysis of both human and machine interactions

• Typical system behavior architectures do not anticipate all possible outcomes, without intentional analysis
  – This problem becomes intractable without tools to help (30 sequential choices of two alternatives results in over 1.07 billion possible outcomes)
  – Derivation of constraints forms a level of requirements to constrain the system behavior to what is expected and desired

• Not intended for detailed considerations such as data through-put, physical performance, geographical or spatial reference
Why consider Patterns?

• Design patterns
  – Re-use of successful patterns
  – Limit or eliminate unwanted patterns

• Model checking
  – Logical consistency
  • Positive-patterns: send then receive, write then read, request then authorize, have fuel then take action, …
  • Anti-patterns: receive before send, read before write, authorize before request, take action without fuel, …
  – Discovery of inherent nature of the architecture

• Design analysis
  – Derive the probability of successful outcomes
  – Derive relative frequency of interactions, e.g. N-squared diagram
    • Well-traveled pathways
    • Rare occurrences
    • Modularity of closely related interactions

• Design of experiments
  – Interactions enable an opportunity for verification in test
Monterey Phoenix (MP) Basics

- Based upon Small Scope Hypothesis (Jackson, 2012), such that most problems can be found with just a few iterations
- Behavior modeling platform that derives all possible combinations of behaviors, within the scope of execution
- Incorporates a concise language, employing principles of predicate logic
- Behaviors described as hierarchical \( (inclusion) \), temporal \( (precedence) \), or user-described
- Interactions \textit{within} a system defined separately from interactions \textit{among} systems
- Constraints limit the outcomes of unwanted behaviors and thereby establish a set of requirements for the system
- Attributes easily indicated in the model
  - favorable and unfavorable outcomes used in the example model
- Assertion checking provides a means to query the model, finding any occurrence of a pattern
- Available for anyone to use with the MP-Firebird Analyzer, at https://firebird.nps.edu
Methodology

1. Define the Behavior Narrative
   - Natural language description

2. Identify the Events
   - Root, Atomic, and Composite Events
   - Internal Inclusion and Precedence Relationships

3. Define Coordination
   - Interactive Inclusion and Precedence Relationships

4. Define Constraints
   - Logical consistency, Simplification, and Design Requirements

5. Identify Patterns
   - Favorable and unfavorable behaviors
   - Topology

6. Analyze the Model
   - Probability calculation
   - Design Structure Matrix (DSM)

Input from Subject Matter Experts (SMEs)

Architectural Execution, Verification, and Validation

Concept of Behaviors

Re-Use of Template from Repository

Formalism of System Behavior

Formalism of System to System and System to Environment Behavior

Re-Use of Template from Repository

Key
- SME Input
- Methodology Step
- MP Execution
- Progression
- Regression, rework

All possible outcomes within scope of execution

Newly derived Templates entered into Repository for Re-use

Constrained behaviors, restricting undesirable interactions

Formal Definition of Requirements

Derived deficiencies and common errors
Decision Pattern Example

The importance of checklists

- Surgical checklists are now standard in all hospitals
- Inspired by other high pressure industries like aviation
- Checklists have helped cut death and complication from surgery by more than an order of magnitude
- A checklist helps to minimise the many layers of hierarchy in the theatre
- It helps all team members know what to follow basic procedures

Source: Dr Atul Gawande, Lead advisor to the World Health Organisation on patient safety

References:
Flight 173: http://lessonslearned.faa.gov/ll_main.cfm?TabID=1&LLID=42
The Behavior Narrative

1. Formal Hierarchy
2. Routine Procedure
3. Unexpected and unplanned occurrence
4. Intense focus and impaired perception
5. Change of external conditions
6. Subordinate recognizes true problem
7. Attempt to communicate solution
8. Leadership failure to recognize the problem
9. Failure
10. Response

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Medical example</th>
<th>Aviation example</th>
</tr>
</thead>
</table>

References:
Flight 173: http://lessonslearned.faa.gov/ll_main.cfm?TabID=1&LID=42
The Behavior Events for Each Scenario

<table>
<thead>
<tr>
<th>Surgery Scenario</th>
<th>Model Abstraction</th>
<th>Aircraft Mishap Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condition of the patient</td>
<td>Environment</td>
<td>Condition of the aircraft</td>
</tr>
<tr>
<td>Surgeon</td>
<td>Leader</td>
<td>Pilot</td>
</tr>
<tr>
<td>Nurse</td>
<td>Subordinate</td>
<td>Flight Engineer</td>
</tr>
<tr>
<td>Surgery</td>
<td>Routine procedure</td>
<td>Landing</td>
</tr>
<tr>
<td>Patient airway</td>
<td>Problem state</td>
<td>Fuel state</td>
</tr>
<tr>
<td>Surgeon Perception</td>
<td>Perception</td>
<td>Pilot Perception</td>
</tr>
<tr>
<td>Nurse Perception</td>
<td>Perception</td>
<td>Flight Engineer Perception</td>
</tr>
<tr>
<td>Nurse Speaking</td>
<td>Communication</td>
<td>Flight Engineer Speaking</td>
</tr>
<tr>
<td>Surgeon Processes Nurse’s Voice</td>
<td>Reception</td>
<td>Pilot Processes Flight Engineer’s Voice</td>
</tr>
<tr>
<td>Next Action</td>
<td>Decision</td>
<td>Next Action</td>
</tr>
<tr>
<td>Death or Survival</td>
<td>Outcome state</td>
<td>Mishap or Landing</td>
</tr>
</tbody>
</table>

Key:
- Root event - establishes a hierarchy
- Composite event - contains sub-events
- Atomic event - contains no sub-events
The Behavior Model in Monterey Phoenix

Leader and subordinate Hierarchy

Procedure shared by participants within the environment

Individual perception of the environment

Communication by the subordinate

Leader’s context toward subordinate

Decision

Leader | Subordinate | Environment
---|---|---

Routine procedure

Environmental Conditions

Perception

Problem state

Communication

Reception

Decision

Outcome state

Key
- Root event (establishes hierarchy)
- Composite event (contains sub-events)
- Atomic event (contains no sub-events)
- Inclusion relationship
- Precedence relationship
- Note (not part of MP)

Precedence and Inclusion Relationships are shown as solid and dotted arrows, respectively.

The composite events consist of alternatives between two events, one favorable and one not favorable (e.g. either a problem exists or does not exist)

Execution of the model results in 128 possible event traces or use cases.
The topology is constant for all traces

Additional semantics are needed to distinguish each of the use cases.

- Recognize_environment: favorable;
- Not_recognize_environment: unfavorable;
- Receive_input: favorable;
- Not_receive_input: unfavorable;
- Correct_decision: favorable;
- Not_correct_decision: unfavorable;
- Communicate_observation: favorable;
- Not_communicate_observation: unfavorable;
- Problem: unfavorable;
- No_problem: favorable;
- Successful_outcome: favorable;
- Failed_outcome: unfavorable;
An Instance of Behavior

- Execution of the model produced all possible traces or use cases
- The scenario outlined at the beginning of the presentation is identified as Template 9: Leader fails to consider the subordinate input

Template 9 (T9): Leader fails to consider subordinate input

Black textboxes are unfavorable
Gray textboxes are favorable
All Possible Behaviors of the Model

T1: Both leader and subordinate correctly perceive no real problem

T2: Both leader and subordinate perceive a real problem

T3: Subordinate perceives a problem, though none exists

T4: Subordinate fails to perceive real problem

T5: Leader perceives a problem, though none exists

T6: Leader trusts subordinate perception

T7: Incorrect perception, but no problem exists

T8: Incorrect perception, but correct decision and action

T9: Leader fails to consider subordinate input

T10: Subordinate fails to communicate problem

T10: Leader and Subordinate(s) are wrong with communication

T12: Everything unfavorable
<table>
<thead>
<tr>
<th>Event type</th>
<th>Event name</th>
<th>Leader</th>
<th>Subordinates</th>
<th>Environment</th>
<th>Perception</th>
<th>Recognize_environment</th>
<th>Reception</th>
<th>Receive_input</th>
<th>Decision</th>
<th>Correct_decision</th>
<th>Subordinate</th>
<th>Communication</th>
<th>Communication_observation</th>
<th>Problem_state</th>
<th>No_problem</th>
<th>Outcome_state</th>
<th>Successful_outcome</th>
<th>Problem</th>
<th>Not_recognize_environment</th>
<th>Not_receive_input</th>
<th>Incorrect_decision</th>
<th>Failed_outcome</th>
<th>Not_communicate_observation</th>
<th>Routine_procedure</th>
<th>favorable</th>
<th>unfavorable</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROOT</td>
<td>Leader</td>
<td>1</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROOT</td>
<td>Subordinates</td>
<td>2</td>
<td>12</td>
<td></td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ROOT</td>
<td>Environment</td>
<td>3</td>
<td>12</td>
<td></td>
<td>12</td>
<td>12</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Perception</td>
<td>4</td>
<td>10</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Recognize_environment</td>
<td>5</td>
<td>10</td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Reception</td>
<td>6</td>
<td>8</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Receive_input</td>
<td>7</td>
<td>8</td>
<td></td>
<td>8</td>
<td>12</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Decision</td>
<td>8</td>
<td>8</td>
<td></td>
<td>12</td>
<td>12</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Correct_decision</td>
<td>9</td>
<td>12</td>
<td></td>
<td>12</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Subordinate</td>
<td>10</td>
<td>12</td>
<td></td>
<td>12</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Communication</td>
<td>11</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Communicate_observation</td>
<td>12</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Problem_state</td>
<td>13</td>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>No_problem</td>
<td>14</td>
<td>24</td>
<td></td>
<td>10</td>
<td>12</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Outcome_state</td>
<td>15</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Successful_outcome</td>
<td>16</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Problem</td>
<td>17</td>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Not_recognize_environment</td>
<td>18</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Not_receive_input</td>
<td>19</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Incorrect_decision</td>
<td>20</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Failed_outcome</td>
<td>21</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COMPOSITE</td>
<td>Not_communicate_observation</td>
<td>22</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATOM</td>
<td>Routine_procedure</td>
<td>23</td>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATOM</td>
<td>favorable</td>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ATOM</td>
<td>unfavorable</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Key:**
- **2** Fewest Interactions
- **24** Most Interactions
• A model developer has interest in controlling the behaviors of the system of interest
  – Desired behaviors need to be prominent
  – Undesired behaviors need to be identified, then constrained or eliminated

• Constraints form conditional probabilities and can be described within a Bayesian belief network

• Determining the probability of a particular sequence of events (use case) of a Behavior model can help the developer to gauge the effectiveness of the system.
Applying the Approach to the Cross Domain Problem

- The Monterey Phoenix model topology creates the structure for the Bayesian belief network.
- Additional relationship is shown for one of the constraints of the model.

*The constraints establish explicit cases for conditional probability.*
**Constraints**

**Constraint 1:** If no problem exists, \((k=1)\), then have a successful outcome \((q=1)\).
\[
P(q = 1 \mid k = 1) = 1; \quad P(q = 2 \mid k = 1) = 0
\]

**Constraint 2:** If the subordinate makes no communication \((m = 2)\), then leader does not receive communication \((n=2)\).
\[
P(n = 2 \mid m = 2) = 1; \quad P(n = 1 \mid m = 2) = 0
\]

**Constraint 3:** If leader correctly perceives the environment, \((i=1)\), and receives no input from the subordinate, \((n=2)\), then the leader makes a correct decision \((p=1)\).
\[
P(p = 1 \mid i = 1, n = 2) = 1; \quad P(p = 2 \mid i = 1, n = 2) = 0
\]

**Constraint 4:** If the leader receives communication \((n=1)\), the leader makes a correct decision \((p=1)\), and its corollary.
\[
P(p = 1 \mid n = 1) = 1; \quad P(p = 2 \mid n = 1) = 0; \quad P(p = 1 \mid n = 2) = 0; \quad P(p = 2 \mid n = 2) = 1
\]

**Constraint 5:** A correct decision, \((p=1)\), leads to a successful outcome \((q=1)\), and its corollary.
\[
P(q = 1 \mid p = 1) = 1; \quad P(q = 2 \mid p = 1) = 0; \quad P(q = 1 \mid p = 2) = 0; \quad P(q = 2 \mid p = 2) = 1
\]

**Conditional probability listed for each constraint**
All Possible Behaviors of the Model

T1: Both leader and subordinate correctly perceive no real problem

T2: Both leader and subordinate perceive a real problem

T3: Subordinate perceives a problem, though none exists

T4: Subordinate fails to perceive real problem

T5: Leader perceives a problem, though none exists

T6: Leader trusts subordinate perception

T7: Incorrect perception, but no problem exists

T8: Incorrect perception, but correct decision and action

T9: Leader fails to consider subordinate input

T10: Subordinate fails to communicate problem

T11: Leader and Subordinate(s) are wrong with communication

T12: Everything unfavorable

\[ P_{success} = \sum_{i=1}^{8} P_{trace_i} = 0.8125 \]

\[ P_{failure} = \sum_{i=9}^{12} P_{trace_i} = 0.1825 \]
Demonstration of Model Execution
Findings

- Behavior modeling of a cross-domain problem provides insight to decision events
- Patterns of behavior identified as templates
- Assertion checking finds all matches to the template, and marks the trace or use case for identification
- Stochastic properties applied to the MP model
- Monterey Phoenix is available for anyone to use at [https://firebird.nps.edu](https://firebird.nps.edu)
Conclusions

• Behavior analysis helps the developer to derive alternative paths of execution
  – Exposes the logic behind inherent within the model
  – Enables insight to the fundamental nature of the system

• Once the logical level is established, more detailed levels of performance can be investigated
Questions and Discussion

• Questions?
• Discussion?
• Contact information:

John Quartuccio
jjquartu@nps.edu


• Back-up notes on MP syntax
Behaviors of the Leader

**Leader**

- **Routine procedure**
  - Perception
    - Recognize_environment
      - Favorable
    - Not_recognize_environment
      - Unfavorable
  - Reception
    - Receive_input
      - Favorable
    - Not_receive_input
      - Unfavorable
  - Decision
    - Correct_decision
      - Favorable
    - Not_correct_decision
      - Unfavorable

**Key**
- Root event (establishes hierarchy)
- Composite event (contains sub-events)
- Atomic event (contains no sub-events)
- Inclusion relationship
- Precedence relationship
- Note (not part of MP)
Event Behaviors of the Leader

- **Root event**
  - established hierarchy

- **Atomic event**
  - no subordinate events

- **Composite events**
  - have subordinate events

**Order**
- events listed in sequence order

**Alternatives**
- separated by the “pipe” character, meaning “or”

```plaintext
ROOT Leader:

Routine_procedure
  Perception
  Reception
  Decision

Perception:
  ( Recognize_environment
    | Not_recognize_environment )

Reception:
  ( Receive_input
    | Not_receive_input )

Decision:
  ( Correct_decision
    | Incorrect_decision )
```
Event Behaviors of the Subordinate

Subordinate

- Routine procedure
- Perception
- Communication

Key
- Root event (establishes hierarchy)
- Composite event (contains sub-events)
- Atomic event (contains no sub-events)
- Inclusion relationship
- Precedence relationship
- Note (not part of MP)

Recognize_environment
  - favorable

Not_recognize_environment
  - unfavorable

Communicate_observation
  - favorable

Not_communicate_observation
  - unfavorable
Event Behaviors of the Subordinate

One or many Subordinate events
• indicated by the “plus” character
• determined by scope of execution

```
ROOT Subordinates: {+ Subordinate +}

Subordinate:
  Routine_procedure
  Perception
  Communication

Communication:
  ( Communicate_observation
    | Not_communicate_observation )
```
### Event Behaviors of the Environment

**Environment**

- **Routine procedure**
  - **Problem_state**
    - **No_problem** (favorable)
    - **Problem** (unfavorable)
  - **Outcome_state**
    - **Successful_outcome** (favorable)
    - **Failed_outcome** (unfavorable)

#### Key
- **Root event (establishes hierarchy)**
- **Composite event (contains sub-events)**
- **Atomic event (contains no sub-events)**
- **Inclusion relationship**
- **Precedence relationship**
- **Note (not part of MP)**

---

The diagram illustrates the hierarchy of events and their outcomes within the environment, highlighting the relationships and conditions under which favorable or unfavorable outcomes can be achieved.
Event Behaviors of the Environment

```plaintext
ROOT Environment:
    Routine_procedure
    Problem_state
    Outcome_state

Problem_state:
    ( No_problem
    | Problem )

Outcome_state:
    ( Successful_outcome
    | Failed_outcome )
```
Interactions across events are defined separately from the event behaviors, listed on the previous slides. Separating these descriptions affords great flexibility to the model developer.
Coordination – Interaction Across Events

Interaction 1: Shared procedure

Interaction 2: Leader receipt of input depends on communication by the Subordinate, *as in speaking precedes hearing*

Interaction 3: A Decision Leads to an Outcome

Interaction 4: The Problem State precedes the Perception of both the Leader and Subordinate

Interactions across events are defined separately from the event behaviors, listed on the previous slides.

Separating these descriptions affords great flexibility to the model developer.
Types of Constraints

• Logical consistency
  – A correct model needs to restrict illogical behavior
  – As an example, *If* a message is not sent, *then* it cannot be received, or
    *If* a car does not exist, *then* I cannot drive it.

• Simplification
  – Simplification may be applied to improve clarity and encourage the developer’s focus on key events
  – As an example, *If* a leader receives input, *then* always have a correct decision
  – This results in fewer use cases to analyze

• Design
  – Design requirements may be built to eliminate unwanted behaviors
  – As an example, *If* an aircraft is out of fuel, *then* make the nearest safe landing, ignoring less critical tasks.
  – This example may use automation to achieve the desired result.

• Definition
  – Definition of a particular series of events
  – As an example, *If* a leader makes a correct decision, *then* always have a successful outcome
Constraints

Constraint 1: If there is no problem in the Environment, then have a successful outcome.

Constraint 2: If all Subordinates do not communicate, then the Leader has no input.

Constraint 3: If the Leader recognizes the Environment and does not receive input, then the Leader makes a correct decision.

Constraint 4: If the Leader receives input, then the Leader makes a correct decision, and if the Leader does not receive input, then the Leader makes an incorrect decision.

Constraint 5: If the Leader makes a correct decision, then have a successful outcome and if the Leader makes an incorrect decision, then have a failed outcome.

These constraints reduce the number of possible traces or use cases from 128 to 12.
These constraints reduce the number of possible traces or use cases from 128 to 12.
Assertion Checking:
An automated search for each of twelve templates is conducted during execution.

Template 1 is shown, where all alternatives are favorable.

Mark/Say command provides a text statement.
• Prior patterns were demonstrated for the entire system function

• Segments of the system function also show patterns:
  – Observe, Orient, Decide, Act (OODA) Loop

  ![OODA Loop Diagram]

  Perception ➔ Compilation ➔ Decision ➔ Action

  Observe ➔ Orient ➔ Decide ➔ Act

  – Cooperative OODA Loop

  ![Cooperative OODA Loop Diagram]

  Perception ➔ Transmission ➔ Reception ➔ Decision ➔ Action

  Observe ➔ Orient ➔ Decide ➔ Act

  Tx ➔ Rx