Nick’s Bio

- Nick has been a Systems Engineer at Raytheon for 3 years, working in the Patriot BMC4I Requirements Team. Nick joined Raytheon after graduating from the University of Massachusetts Amherst with a Bachelor of Science in Electrical Engineering. He is currently pursuing a Master of Science in Industrial Engineering, with a certificate from the Gordon Institute of Engineering Leadership. As a part his capstone project, Nick has developed a series of MBSE work instructions and a proof of concept model of a notional Urban Traffic Control System.
Key MBSE Enablers with Examples

Nick Driscoll (Presenter)
Phil Levesque

Abstract: 19920
11/28/2017
Agenda

- Model Based Systems Engineering (MBSE) Description
- MBSE Environment and Enablers
- Example Model Using Enablers
Systems Engineering

- Traditional requirements-based designs have Undesirable Effects over the product lifecycle:
  - Incorrect
  - Incomplete
  - Uninformed
  - Ambiguous
  - Infeasible
  - Unverifiable
Model Based Systems Engineering

- Visual representations
  - System Composition
  - Interfaces
  - Behaviors

- Multiple levels of Decomposition
  - Operational – Concept of Operations, Operation and Maintenance
  - System – Requirements and Architecture, System Verification and Validation
  - Component – Detailed Design, Integration and Test

- MBSE can provide:
  - Integrated Environment
  - Design Validation
  - Document Generation
  - Generation of code
Model Based Systems Engineering

- **Visual representations**
  - System Composition
  - Interfaces
  - Behaviors

- **Multiple levels of Decomposition**
  - Operational – Concept of Operations, Operation and Maintenance
  - System – Requirements and Architecture, System Verification and Validation
  - Component – Detailed Design, Integration and Test

- **MBSE can provide:**
  - Integrated Environment
  - Design Validation
  - Document Generation
  - Generation of code
Model Based Systems Engineering

- Visual representations
  - System Composition
  - Interfaces
  - Behaviors

- Multiple levels of Decomposition
  - Operational – Concept of Operations, Operation and Maintenance
  - System – Requirements and Architecture, System Verification and Validation
  - Component – Detailed Design, Integration and Test

- MBSE can provide:
  - Integrated Environment
  - Design Validation
  - Document Generation
  - Generation of code
MBSE Environment

- **Project Management**
  - Task Management
  - Defect Management

- **Integrated Development Environment**
  - Automated Model Manipulation

- **Modeling Tool**
  - Requirements Analysis
  - Functional Analysis
  - Architecture Analysis
  - Design Synthesis
  - System Thread Analysis

- **Collaboration Tools**
  - Configuration Management
  - Model Review
  - Model Reporting
  - Model Impact Analysis

- **Publishing**
  - Model Reporting
  - Automated Document Generation

- **Requirements Management**
  - Requirements Definition
  - Requirements Analysis

- **Work Item Trace**
- **Model Stream**
- **Model Data**
- **Requirements Links**
- **Script Execution**
- **Review Comments**
MBSE Impact on Design Methodology

- **Design Efficiency**
  - Consistent approach to MBSE
  - Stricter Analysis

- **Enhanced Communication and Knowledge Transfer**
  - Ease complexity management and understanding
  - Graphics and flowcharts are less convoluted than requirements specifications

- **Improved Design Quality**
  - In-phase defect detection
  - Defect reduction
  - Configuration Management
Enablers Supported by MBSE Environment

- **Modeling Enabler/Methodology**
- **Integrated Design Reviews**
  - Improved Quality
  - In-phase Correction
  - Knowledge Dissemination
  - Save Costs
  - Reduce Schedule
- **Configuration Management**
  - Consistency
  - Collaboration
- **Team/Metric Tracking**
  - Defect Tracking
  - Project Progress Reports
Enablers Supported by MBSE Environment

- **Modeling Enabler/Methodology**
  - Improved Quality
  - In-phase Correction
  - Knowledge Dissemination
  - Save Costs
  - Reduce Schedule

- **Integrated Design Reviews**
  - Improved Quality
  - In-phase Correction
  - Knowledge Dissemination
  - Save Costs
  - Reduce Schedule

- **Configuration Management**
  - Consistency
  - Collaboration

- **Team/Metric Tracking**
  - Defect Tracking
  - Project Progress Reports
Enablers Supported by MBSE Environment

- **Modeling Enabler/Methodology**
- **Integrated Design Reviews**
  - Improved Quality
  - In-phase Correction
  - Knowledge Dissemination
  - Save Costs
  - Reduce Schedule

- **Configuration Management**
  - Consistency
  - Collaboration

- **Team/Metric Tracking**
  - Defect Tracking
  - Project Progress Reports

- **Element-wise lockout:**

- **Collaborative Lockout Notifications:**

- **Out-of-Sync Notifications:**

Images Extracted from Rhapsody using Rational Design Manager
Enablers Supported by MBSE Environment

- **Modeling Enabler/Methodology**
- **Integrated Design Reviews**
  - Improved Quality
  - In-phase Correction
  - Knowledge Dissemination
  - Save Costs
  - Reduce Schedule
- **Configuration Management**
  - Consistency
  - Collaboration
- **Team/Metric Tracking**
  - Defect Tracking
  - Project Progress Reports
Example Model Using Enablers

- **Rationale for Urban Traffic Control (UTC) System as an Example:**
  - Notional example of a highly-variable complex system
  - Multiple levels of decomposition
  - Sharable across-company and externally without divulging customer or company information

- **UTC System Customer Needs:**
  - Maintain Traffic Flow
  - Public Transportation Priority
  - Timely Response to Incidents
  - Maintain Pedestrian Well-Being
  - Control Center Design Constraints
  - System Maintenance and Fault Detection
  - Interface Requirements
UTC System Operational Block Diagram

- Operational Block Definition Diagram: high level graphical overview of the operational concept
- Identifies the other organizations and systems in the system under design’s operational environment
- Describes the relationships between the system under design and the identified organizations and systems
UTC System Block Definition Diagram

- Block Definition Diagram: A representation of the structure elements and their relationships.
UTC System Use Case Diagram

- Use Case Diagram: Define the main functions that the system must perform. Used to develop the operational threads.
UTC System Activity Diagram

- Activity Diagram: Represents a specific system behavior or set of system behaviors. Similar to a flow chart, can depict the interactions between various external actors, or elements within the system.
- Describes flow-based behavior.
UTC System Sequence Diagram

- Sequence Diagram: Represents message exchanges between systems, subsystems, or components.
- Describes message-based behavior
UTC System Internal Block Diagram

- Internal Block Diagram: Represents the interconnection and interfaces between the internal parts of a block (enterprise, system, or subsystem)
UTC System State Machine Diagram

- State Machine Diagram: Defines the states and modes of the system, and depicts the transitions from one state to another.
- Describes event-based behavior
## UTC System Metrics

<table>
<thead>
<tr>
<th>Metric Type</th>
<th>Count</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diagrams/Views (Total)</td>
<td>85</td>
<td>Total number of diagrams in model</td>
</tr>
<tr>
<td>Activity Diagrams</td>
<td>27</td>
<td>Total number of activity diagrams</td>
</tr>
<tr>
<td>Sequence Diagrams</td>
<td>24</td>
<td>Total number of sequence diagrams</td>
</tr>
<tr>
<td>Block Definition Diagrams</td>
<td>16</td>
<td>Total number of block definition diagrams</td>
</tr>
<tr>
<td>Internal Block Diagrams</td>
<td>13</td>
<td>Total number of internal block diagrams</td>
</tr>
<tr>
<td>State Charts</td>
<td>1</td>
<td>Total number of state charts</td>
</tr>
<tr>
<td>Use Case Diagrams</td>
<td>3</td>
<td>Total number of use case diagrams</td>
</tr>
<tr>
<td>Requirements Diagrams</td>
<td>1</td>
<td>Total number of requirements diagrams</td>
</tr>
<tr>
<td>Structural Elements</td>
<td>51</td>
<td>Includes blocks for Enterprise, Systems, Subsystems, Nodes, Organizations</td>
</tr>
<tr>
<td>Interface Items</td>
<td>142</td>
<td>Includes send event actions, exchanged messages, interfaces, interface blocks</td>
</tr>
<tr>
<td>Functional Elements</td>
<td>46</td>
<td>Includes use cases (threads), activities and call behaviors</td>
</tr>
<tr>
<td>People Elements</td>
<td>20</td>
<td>Enterprise Actors</td>
</tr>
<tr>
<td>Time-Related Events</td>
<td>485</td>
<td>Includes transitions, events, flows, interaction occurrences, sequences, and states</td>
</tr>
<tr>
<td>Satisfied Requirements</td>
<td>29</td>
<td>Number of requirements traced to an element</td>
</tr>
<tr>
<td>Unsatisfied Requirements</td>
<td>27</td>
<td>Number of requirements not traced to an element</td>
</tr>
<tr>
<td>Percent of Requirements Linked</td>
<td>52%</td>
<td>Percentage of total requirements traced to a model element</td>
</tr>
<tr>
<td>Percent Under Configuration Control</td>
<td>100%</td>
<td>Model is configure controlled in RDM with the candidate as the only approver</td>
</tr>
</tbody>
</table>
UTC System Requirements Compliance

- Model Elements are linked to requirements within Rhapsody, and satisfaction tables can be output to help determine model completeness:

<table>
<thead>
<tr>
<th>Requirement ID</th>
<th>Specification</th>
<th>Satisfying Element</th>
</tr>
</thead>
<tbody>
<tr>
<td>UTC_46</td>
<td>The UTC System shall have an Operational State.</td>
<td>Operational</td>
</tr>
<tr>
<td>UTC_51</td>
<td>The UTC System shall avoid large fluctuations in traffic control behavior due to temporary traffic pattern changes.</td>
<td>changeSignal, detectCongestion, evDetectCongestion, commandSignalChange, detectCongestion, executeSignalChange</td>
</tr>
<tr>
<td>UTC_53</td>
<td>The UTC System shall provide a limited sub-set of capabilities when faced with a disaster scenario.</td>
<td>Limited, Emergency Operations</td>
</tr>
<tr>
<td>UTC_54</td>
<td>The UTC System shall be able to transition to Emergency Operations within 1 hour of a State of Emergency Declaration.</td>
<td>evEmergencyOps, Emergency Operations</td>
</tr>
<tr>
<td>UTC_56</td>
<td>The UTC system shall provide priority to public transportation without increasing traffic congestion.</td>
<td>commandSignalChange, executeSignalChange, changeSignal, detectBus, evDetectBus</td>
</tr>
</tbody>
</table>
| UTC_58         | The UTC system shall detect all traffic incidents within 1 minute of occurrence to include:  
  - Multiple Vehicle Collisions  
  - Single Vehicle Collisions with stationary objects (light posts, buildings, etc.)  
  - Single Vehicle Collisions with pedestrians, bicyclists and/or animals  
  - Debris in the roadway. | assessSensorData, senseEnvironment, detectIncident, determineIncidentType, evDetectIncident |
Summary

- Facilitating transition to Model Based Systems Engineering
- Enhanced communication and knowledge transfer
- Reduced lifecycle cost through improved design quality
- MBSE and SysML to model complex systems
- Potential re-use
Questions?
Contact Information

Nicholas Driscoll
IDS HQ, Tewksbury, MA
Nicholas.J.Driscoll@Raytheon.com
Nick has been a Systems Engineer at Raytheon for 3 years, working in the Patriot BMC4I Requirements Team. Nick joined Raytheon after graduating from the University of Massachusetts Amherst with a Bachelor of Science in Electrical Engineering. He is currently pursuing a Master of Science in Industrial Engineering, with a certificate from the Gordon Institute of Engineering Leadership. As a part his capstone project, Nick has developed a series of MBSE work instructions and a proof of concept model of a notional Urban Traffic Control System.

Philip Levesque
IDS HQ, Tewksbury, MA
Philip_R_Levesque@Raytheon.com
Phil Levesque is a Senior Principal Systems Engineer with Raytheon. Phil is a Raytheon Certified Architect and has worked in Systems Engineering for the past 14 years. Phil holds a MS in Computer Engineering degree from the University of Massachusetts at Lowell and BS in Electrical Engineering degree with a double major in Electrical Engineering & Computer Science from the University of Massachusetts at Lowell.
Backup
MBSE Environment Tooling

**Modeling Tool**
- Rhapsody
- MagicDraw
- Enterprise Architect
- Etc.

**Project Management**
- Rational Team Concert
- NoMagic Teamwork
- MS Project
- Etc.

**Integrated Development Environment**
- Eclipse
- NetBeans
- Sun Java Studio
- Etc.

**Collaboration Tools**
- Rational Design Manager
- NoMagic Teamwork
- WebEA
- Rational ClearCase
- GitHub
- Etc.

**Requirements Management**
- DOORS
- PTC Integrity
- Excel/CSV
- Requirements Modeling

**Publishing**
- Rational Publishing Engine
- Native to Modeling Tool