PHM for Ground Support Systems Case Study: From requirements to integration

Chris Teubert, Matt Daigle, Kai Goebel

SGT, Inc.

NASA Ames Research Engineer

Overview

- Tell story: end-to-end process: From requirements to integration
- The surprises
- Differences between research-production
  - How to resolve differences
- Challenges in communicating with system engineers
  - Our needs/abilities/constraints/language
  - Their needs

Background- Who are we?

Diagnostic and Prognostic Group- Research group at NASA Ames Research Center- Intelligent Systems Division

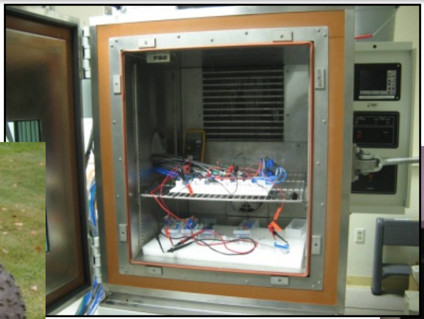
Algorithm Development

- Prototyping
- Modeling: Nominal and Fault
- Simulation

Experimental Validation

- Hardware-in-the-loop validation
- Benchmarking
- Motivation for new research

- Metric Development
- PHM in the Systems Engineering Process
- PHM as a decision support tool
- Autonomous decision making



Background- Ground Support Systems

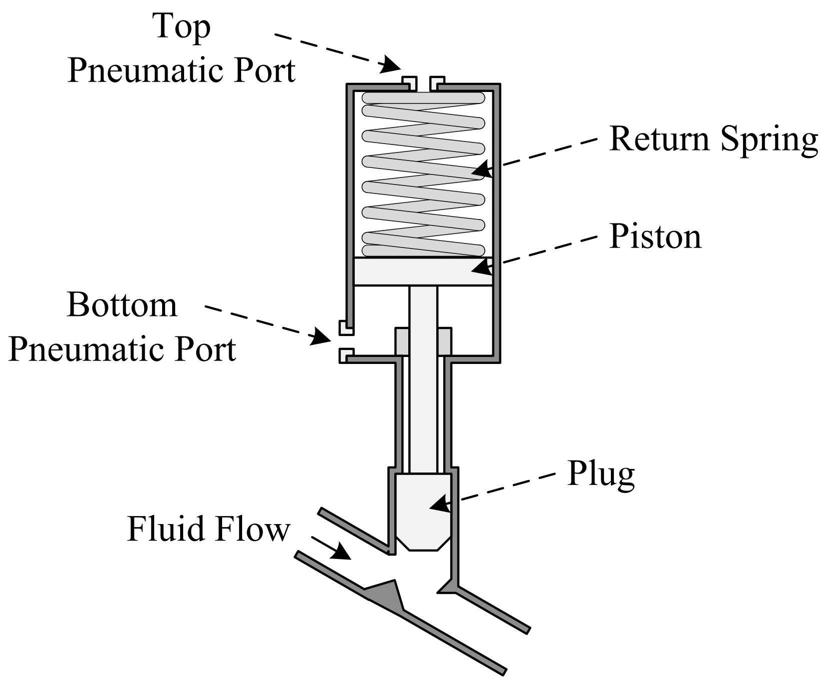
- Rocket ground support consists of many complex and critical systems. There is a real need for PHM
- Support development of a reliable low-cost launch capability for launch a variety of different rockets in a fraction of todays time
- Develop maintenance technologies for advanced ground systems at Kennedy Space Center
- PHM is an integral part of this technology portfolio
- Multiple targets- Iterative
- We first got involved in 2009

Credit- NASA

Precursor work: FDIR

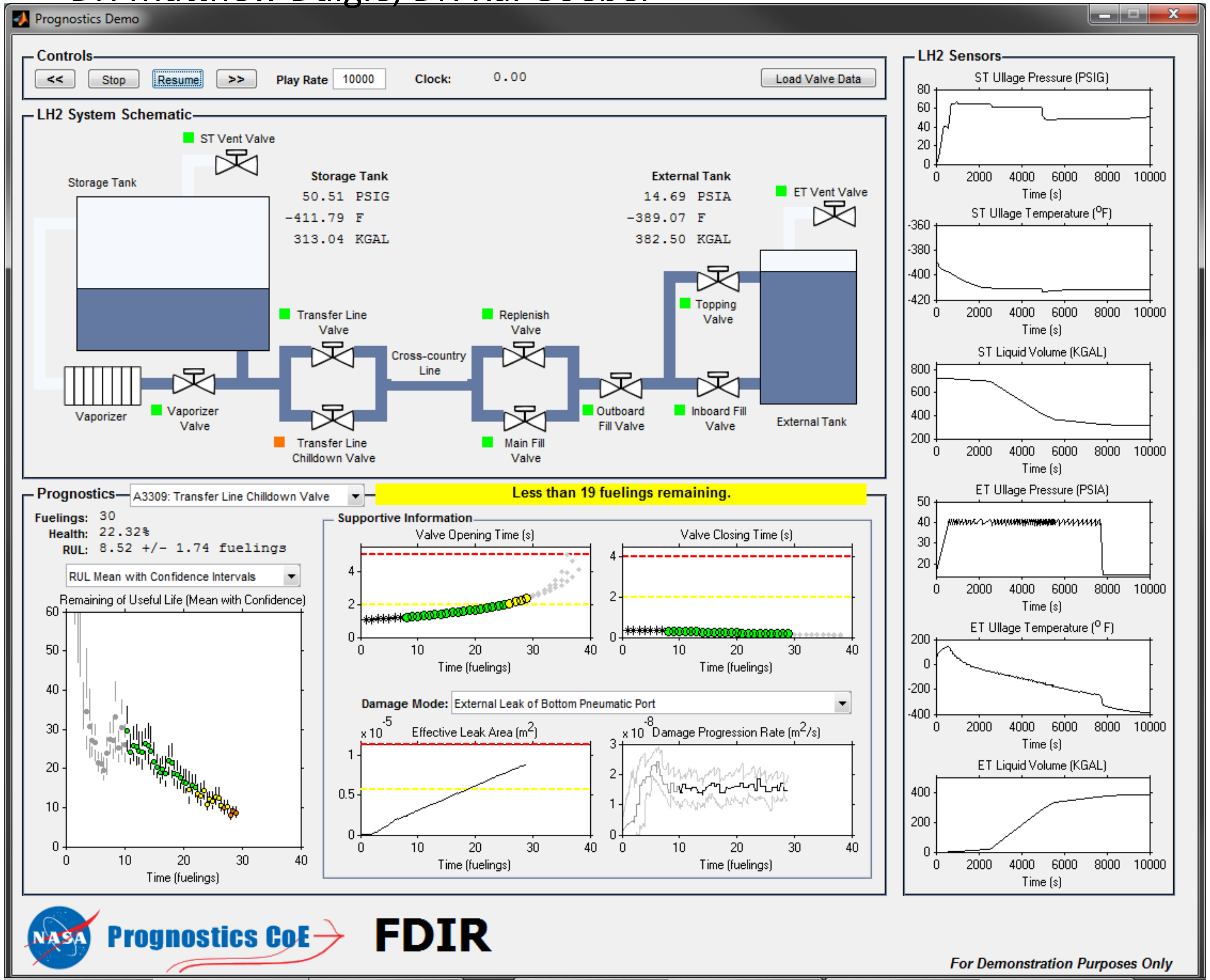
Dr. Matthew Daigle, Dr. Kai Goebel

- Goal was to provide proof-of-concept demonstration of prognostics for ground support systems (cryogenic propellant loading)
- Analyzed PRACA database identifying component faults, repairs, and other issues to identify which components are most suitable for prognostics
  - Investigated pneumatic valves, centrifugal pumps, solenoid valves
- Developed physics models of components with damage propagation, and used particle filter based prognostics approach
  - Included leak faults (most common), friction faults, and spring faults
- Partially validated with Shuttle valve data



Precursor work: FDIR

Dr. Matthew Daigle, Dr. Kai Goebel



Play Rate 10000 Clock: 0.00

Storage Tank	External Tank
50.51 PSIG	14.69 PSIA
-411.79 F	-389.07 F
313.04 KGAL	382.50 KGAL

Prognostics— A3309: Transfer Line Chilldown Valve

Less than 19 fuelings remaining.

Fuelings: 30
Health: 22.32%
RUL: 8.52 +/- 1.74 fuelings

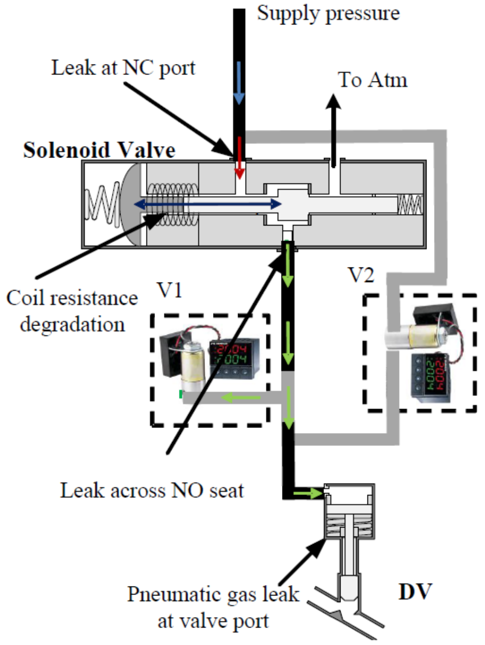
RUL Mean with Confidence Intervals

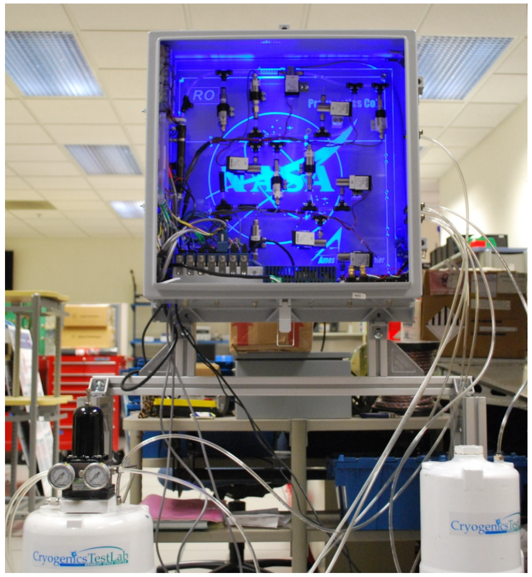
Damage Mode: External Leak of Bottom Pneumatic Port

Pathfinder: Cryogenics Testbed

Dr. Matthew Daigle, Dr. Kai Goebel

- Validation in FDIR was limited – difficult to find run-to-failure data because repairs are made before that happens
- Obtained two cryogenic valves from KSC and developed lab testbed and swappable fault injection rig to inject leakage faults into valves in a controlled manner to validate prognostics for real components with real data
  - Fault injection rig could be disconnected from lab setup and connected to real KSC system to inject faults





Problem Statement

Create reusable software and a “Prognostic Library” to accurately conduct health state estimation and prediction on select ground support components (spacecraft refueling, etc.) and provide useful health state information to operators.

Problem Statement- Notes

Create reusable software and a "Prognostic Library" to accurately conduct health state estimation and prediction on select ground support components (spacecraft refueling, etc.) and provide useful health state information to operators.

Stakeholders

1. Parent project
2. Missions at KSC- future users
3. Mission Control (the operators)
4. Advanced Ground Systems Maintenance Engineers
5. Software maintainers
6. System Designers
7. PHM Community

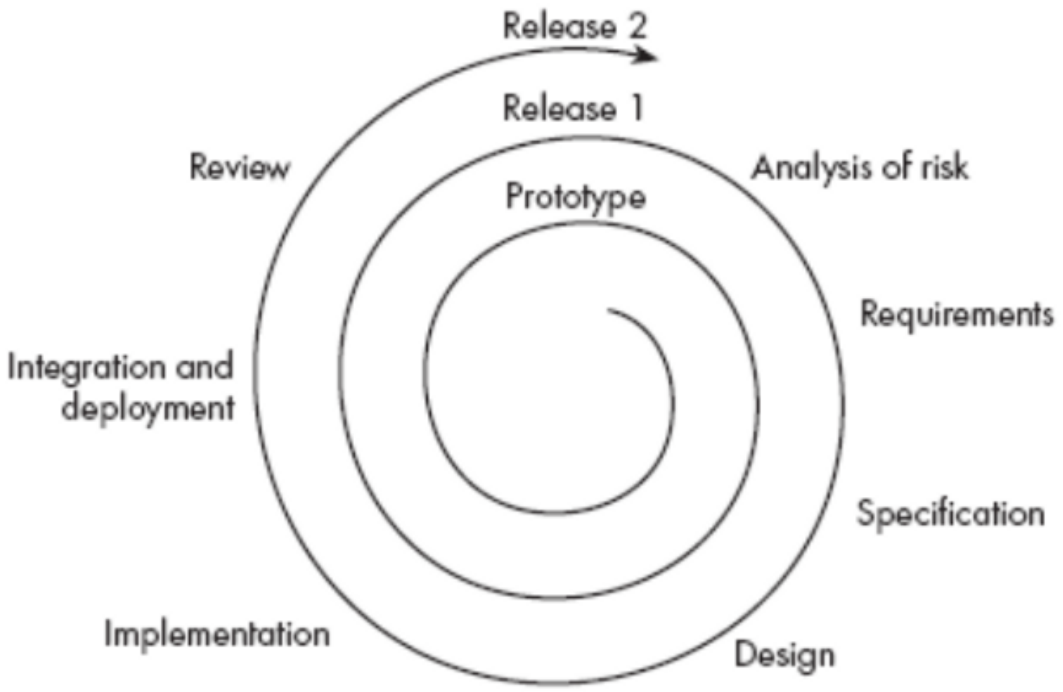
Software Expectations

1. Reusable: Usable under multiple situations- configurable, modular
2. Conduct health state estimation and prediction 1) Accurately and 2) On multiple systems
3. Interface with existing advanced ground support systems
4. Provide health state information to operators
5. The health state information must be useful

Getting Started – SE Process

How it first looked to us

How it looked to them: Organized Systems Engineering process



Credit: Peter Kemp / Paul Smith

Getting Started: Requirement Analysis and Definition

Create reusable software and a "Prognostic Library" to accurately conduct health state estimation and prediction on select ground support components (spacecraft refueling, etc.) and provide useful health state information to operators.

- Interface
  - With Software Infrastructure
  - With Operator- GUI
  - With User- Configuration
- Modularity
- Configurability
- Performance
  - Algorithm Accuracy
  - Speed
- Usability
- Maintainability
- Control Requirements
- Reliability

Requirements "flowed down" to us

Ended in review

Getting Studied: More planning

- Requirements
- Development Plan:
  - Persons Involved
  - Schedule

Software Requirements and Design Specification (SRDS)

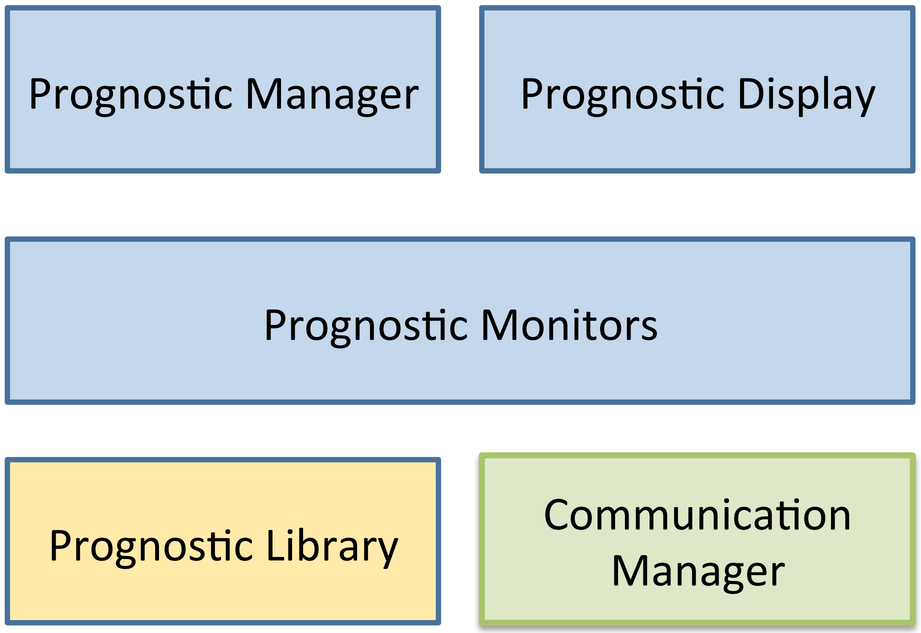
- Control Flow: A roadmap of how individual steps will occur
- Operational Scenarios: A description of how the product will be used
- Architecture: Top-level design of the Prognostic Tool
- Context: How it fits into the greater product

- Test Plan: Unit, Verification, and Validation Tests

Verification Test Procedure

Software Architecting

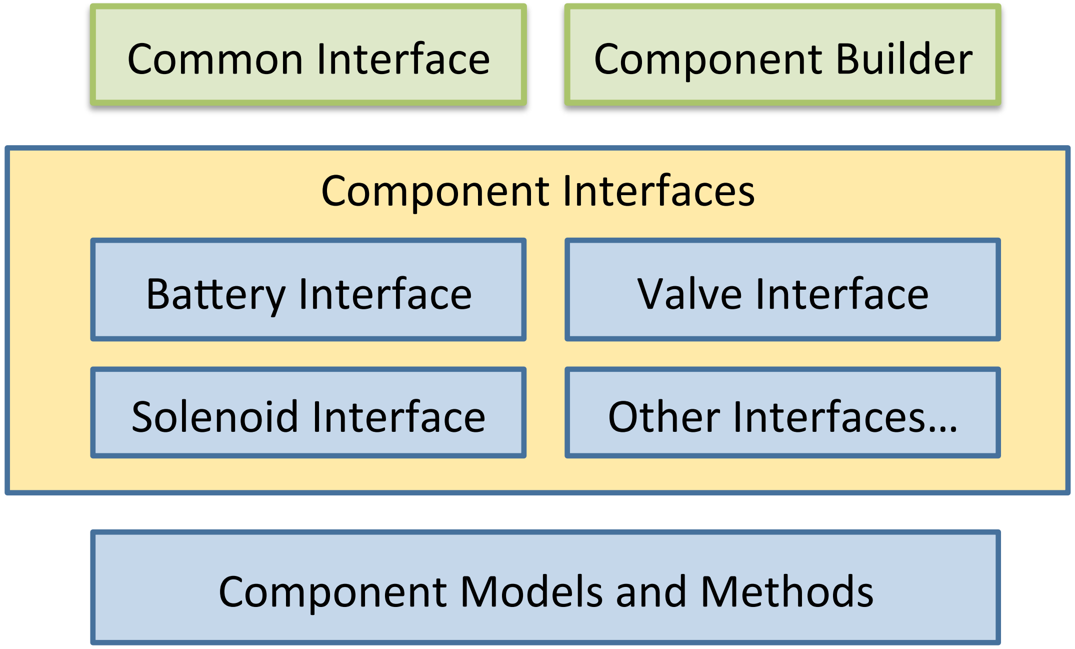
- Interface with higher-level software
- Multithreaded
- C++



<u>Software Expectations</u>

1. **Reusable: Usable under multiple situations- configurable, modular**
2. Conduct health state estimation and prediction
   1) Accurately and
   2) On multiple systems
3. **Interface with existing advanced ground support systems**
4. **Provide health state information to operators**
5. The health state information must be useful

Prognostic Library



1. Some trends are often long term
   - Record prognostic history
2. Need for quality assurance
   - Input data validity checks
   - Results validity checks
3. Software must be maintainable
4. Models and Methods are in Matlab
   - Use Matlab codegen to port into C

<u>Software Expectations</u>

1. **Reusable: Usable under multiple situations- configurable, modular**
2. **Conduct health state estimation and prediction 1) Accurately and 2) On multiple systems**
3. Interface with existing advanced ground support systems
4. Provide health state information to operators
5. **The health state information must be useful**

Configuration

Module Configuration

- Models/Methods to be used
- Health threshold for warning
- Verbosity

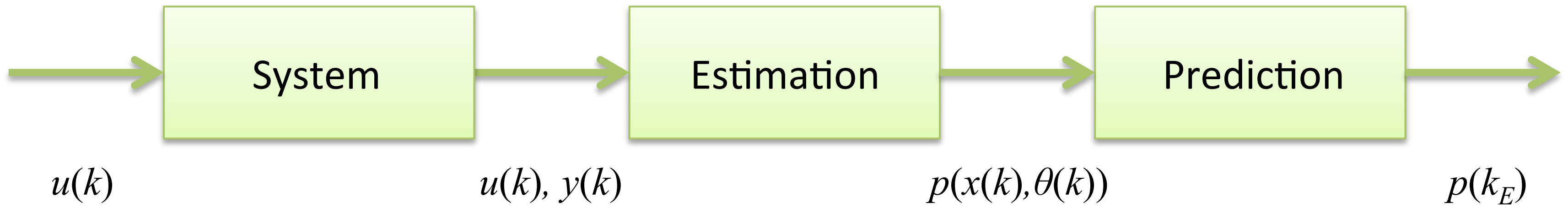
- Communication Configuration
- Reset history for component
- Loop time
- Save Interval
- Prediction Interval
- Name of Component
- Id of Component
- Model Configuration Parameters
- Method Configuration Parameters

Component Configuration

Software Expectations

1. Reusable: Usable under multiple situations- configurable, modular
2. Conduct health state estimation and prediction
   1) Accurately and
   2) On multiple systems
3. Interface with existing advanced ground support systems
4. Provide health state information to operators
5. The health state information must be useful

Prognostic Method: Model Based



- System gets input and produces output
- Estimation module estimates the states and parameters, given system inputs and outputs
  - Must handle sensor noise
  - Must handle process noise
- Requires a model that
  - Describes nominal behavior
  - Describes fault/damage modes
  - Describes progression of faults/damage
- Prediction module predicts time of critical event (eg, EOL), k_E:
  - Must handle state-parameter uncertainty at k_P (time of prediction)
  - Must handle future process noise trajectories
  - Must handle future input trajectories
  - A diagnosis module can inform the prognostics what model to use
- Tools: UKF, Physics-based modeling

Provide health state information to operators

Want standard messages so we can use one GUI template

Question: What information would be useful to Operators?

State of Health

Uncertainty

Prediction

Remaining Useful Life

Uncertainty

Prediction

Meta Data

State Variables

Health:
0: Unavailible
1: Warning
2: Advisary
3: Nominal

Data Quality

Results Quality

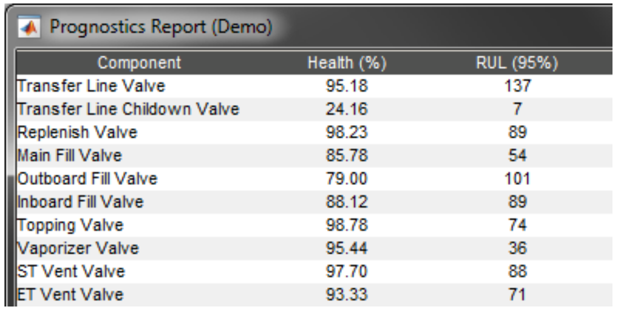
<u>Software Expectations</u>

1. Reusable: Usable under multiple situations- configurable, modular
2. Conduct health state estimation and prediction
   1) Accurately and
   2) On multiple systems
3. Interface with existing advanced ground support systems
4. Provide health state information to operators
5. The health state information must be useful

GUI- *Provide health state information to operators*

Considerations:

- Display important information quickly
- Allow more information to be seen as needed
- Standard format for all components



Prognostics Report (Demo)

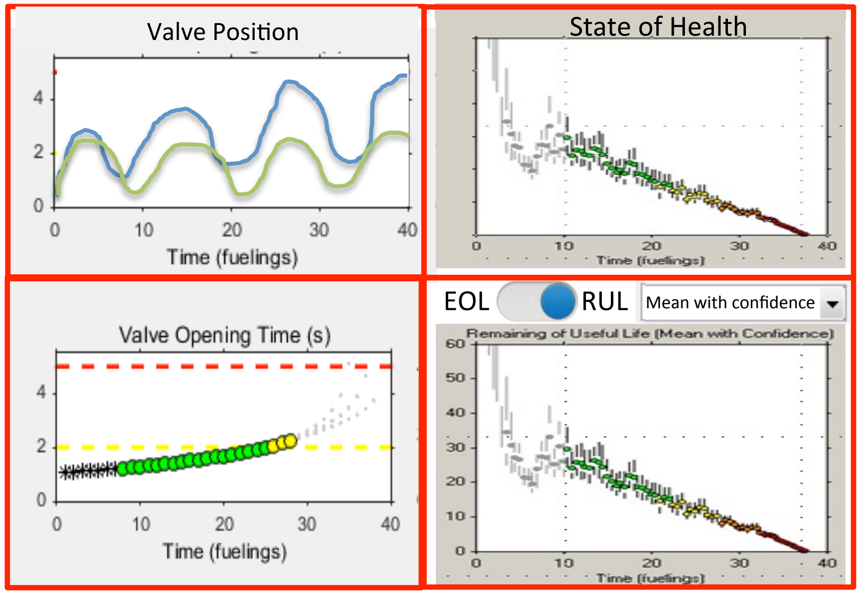
Component	Health (%)	RUL (95%)
Transfer Line Valve	95.18	137
Transfer Line Childown Valve	24.16	7
Replenish Valve	98.23	89
Main Fill Valve	85.78	54
Outboard Fill Valve	79.00	101
Inboard Fill Valve	88.12	89
Topping Valve	98.78	74
Vaporizer Valve	95.44	36
ST Vent Valve	97.70	88
ET Vent Valve	93.33	71

Top Level Summary:

- See status of all components at a single glance
- Click on the component for more information
- Color line for advisory/ warning

Drill Down

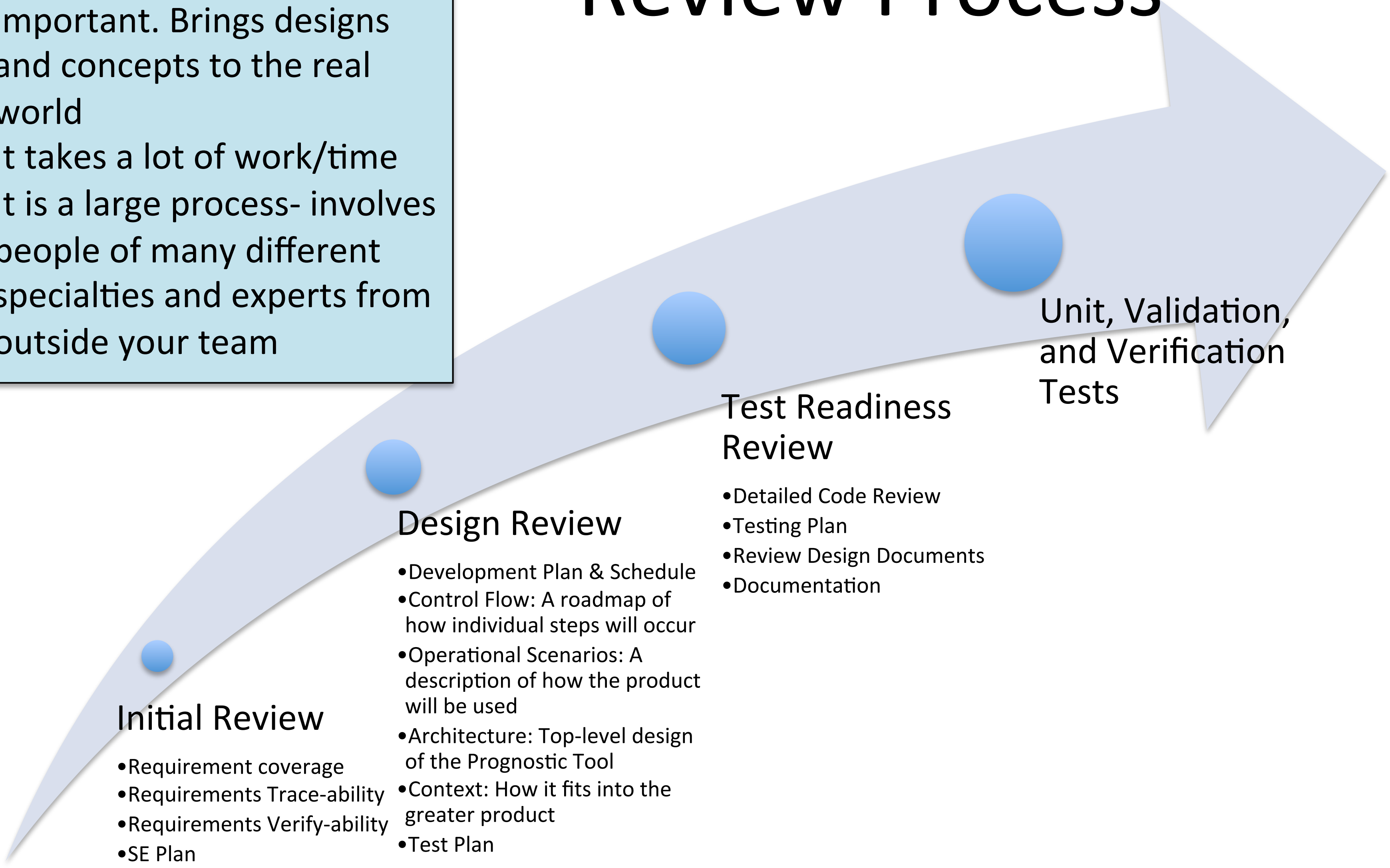
- See more information for a specific component
- See confidence levels
- Display information from state vector



Review Process

Takeaway

1. The review process is very important. Brings designs and concepts to the real world
2. It takes a lot of work/time
3. It is a large process- involves people of many different specialties and experts from outside your team



Initial Review

- Requirement coverage
- Requirements Trace-ability
- Requirements Verify-ability
- SE Plan

Design Review

- Development Plan & Schedule
- Control Flow: A roadmap of how individual steps will occur
- Operational Scenarios: A description of how the product will be used
- Architecture: Top-level design of the Prognostic Tool
- Context: How it fits into the greater product
- Test Plan

Test Readiness Review

- Detailed Code Review
- Testing Plan
- Review Design Documents
- Documentation

Unit, Validation, and Verification Tests

Iteration 1: EFT-1

Dr. Kai Goebel, Dr. Matt Daigle, Chris Teubert
Dr. Indranil Roychoudhury, Dr. Abhinav Saxena

89% SOC

- First flight test for Orion space capsule
- Also first full test case for the product
- Battery Models developed previously for other projects
  - Detailed and Exhaustive V&V Study
  - Validated on other models initially
  - Then validated against real data on varying conditions
- Software running on the ground monitoring batteries onboard Orion Space Capsule in real time.
- Model was validated against Orion test data for application-specific validation

EFT-1: How did it go?

Challenges:

- 80% of time used for SE activities
  - These activities are important- but it did not leave enough time for development.
- Communication Issues
  - Missing requirements or “requirement clarifications”
  - Miss-communication about what information would be available

Metric for success:

- Ideal metric would be using ground truth
  - Not Possible in this case
- Metric 1: Was the information consistent? Does it make sense?
- Metric 2: Was the information useful for the operators?

So was it a success?

- Yes, for the most part- We worked together to create a good verified product that
  1. Worked efficiently in real time
  2. Provided information that makes sense, and
  3. The operators found useful and interesting

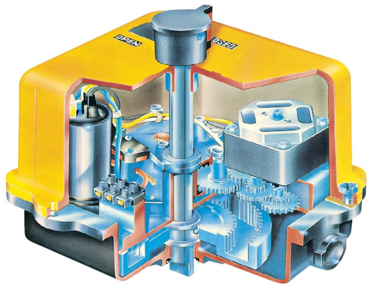
Takeaway

1. Communication
2. Spend time to make sure everyone understands requirements, expectations, needs of each group

Iteration 2: IDU

Dr. Kai Goebel, Dr. Matthew Daigle, Chris Teubert, Dorothy Zoledziowska

- IHM Demonstration for UPSS
- Second Full-Test Case
- Conducting Prognostics on a Rocket Refueling System
- Had team member at KSC to improve communication
- Target component: valve
  - Models developed from similar valve models that our group had previously developed
- Model Validation:
  - Validated against two independently developed simulations
  - Could not get data to validate against



IDU: How did it go?

- Better balancing of systems engineering and development time
- Better communication
- Metric for success:
  - See degradation over time, compare with actual component
  - If failure occurs- compare with data from prognostics
- Has not occurred yet, but we have a much better product that's modular and well-documented

Afterthoughts

Questions

Has anyone here had similar experiences with applying PHM research to create a releasable product? If so, what was it like? What challenges is do encounter?