AVERT
Cognitive Software Assistants for Preventing Human Errors

SOSTC - Improving Space Operations Workshop

April 7, 2021

Presented by: Emilio Remolina
remolina@stottlerhenke.com
650.931.2709
http://www.stottlerhenke.com

Stottler Henke
Smarter Software Solutions

Talk Overview

- Project Team
- Project Summary
- AVERT Architecture
- Implementation Techniques
- Prototypes
- Future Work

Project Team

Stottler Henke	Artificial intelligence tools and applications planning, scheduling, diagnosis intelligent user interfaces autonomous systems
Steven Noneman	NASA human and robotic spaceflight operations 40 years experience on Payload operations and as Flight Controller at MSFC
Michael Haddad	NASA human and robotic spaceflight operations 30 years of experience in NASA operations Worked directly with astronauts on ground processing tasks and support during flight missions

AVERT Summary

Problem

Even Experts make slips

Due to:

- Work overload, loss of **situation awareness (SA)**
- Interruptions, stress, fatigue
- Bad system designs (system behavior not matching user expectations /goals)

AVERT Goal

- Develop intelligent assistants that reduces the likelihood of crew task execution errors
  - Do interventions to improve user workload or SA
  - Support astronauts during deep space missions

AVERT Status

- Defined AVERT architecture to create automated assistants that:
  - Model user affective state, tasks and system situation awareness,
  - Use the above models to identify situations where the crewmember needs help,
  - Intervene to increase task SA during procedure execution and diagnosis tasks,
  - Suggest the automatization of procedures when procedure execution performance is degraded, and
  - Define SA displays to indicate state of task execution and in particular actions taken by the automation, tasks that require the operator intervention, and alerts generated during the task execution.

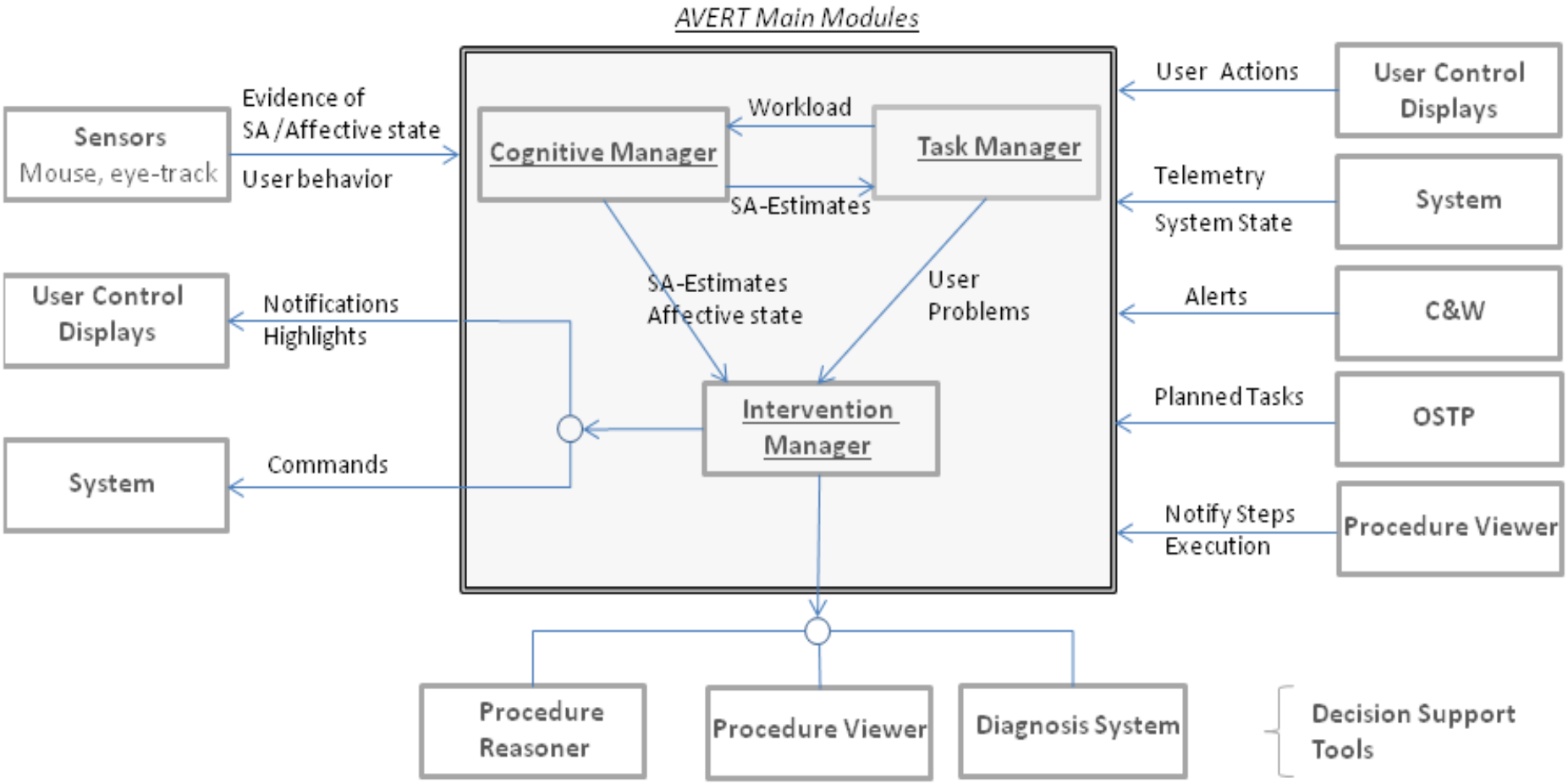
- Demonstrated software-based astronaut assistance concepts when earth-based controllers cannot help an astronaut with a systems problem identification, response, and resolution.
  - Two applications domains: MTV and ADAPT.

Fault Detection Isolation and Recovery (FDIR) CONOPs

- Developed working prototypes illustrating AVERT's FDIR process support:
  - Monitoring: AVERT tracks and monitors the user and the space system C&W system in order to recognize system and user states affecting the execution of operator tasks.
  - Recognition: AVERT assesses the operator's awareness of system status and events, determines the priority of tasks to be performed, and advises whenever the operator is not working on high priority tasks.
  - Reaction: AVERT advises on the criticality of impacted functions, suggests procedures to recover system functions, and monitors the operator's execution of such procedures.
  - Diagnosis: AVERT suggests possible explanations for the observed failures, monitors that the operator crosschecks the explanations for the failures.
  - Recovery: AVERT recommends recovery procedures and monitors the user's execution of the procedures.
  - Procedure execution: AVERT monitors the execution of procedures and produces visual cues when it detects that the operator seems confused, might be about to make a mistake or has not gathered information relevant to the procedure step.
  - Change to LOA: When the operator's task performance is degraded, AVERT suggests changes to the level of automation, for example, by automatizing the procedure execution.

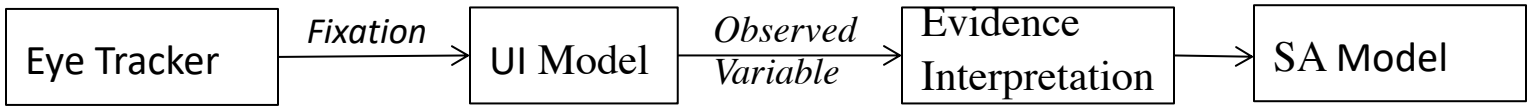
AVERT Architecture – Main Modules

- Cognitive Manager: maintains estimates of user knowledge in *short-term memory* as well as estimates of the user's affective and cognitive states.
- Task Manager: monitors user's actions, system telemetry, and the Caution and Warning (C&W) system in order to detect *problems* with a task execution.
- Intervention Manager: decides on which problems the user should focus on and makes *interventions* to help the operator.

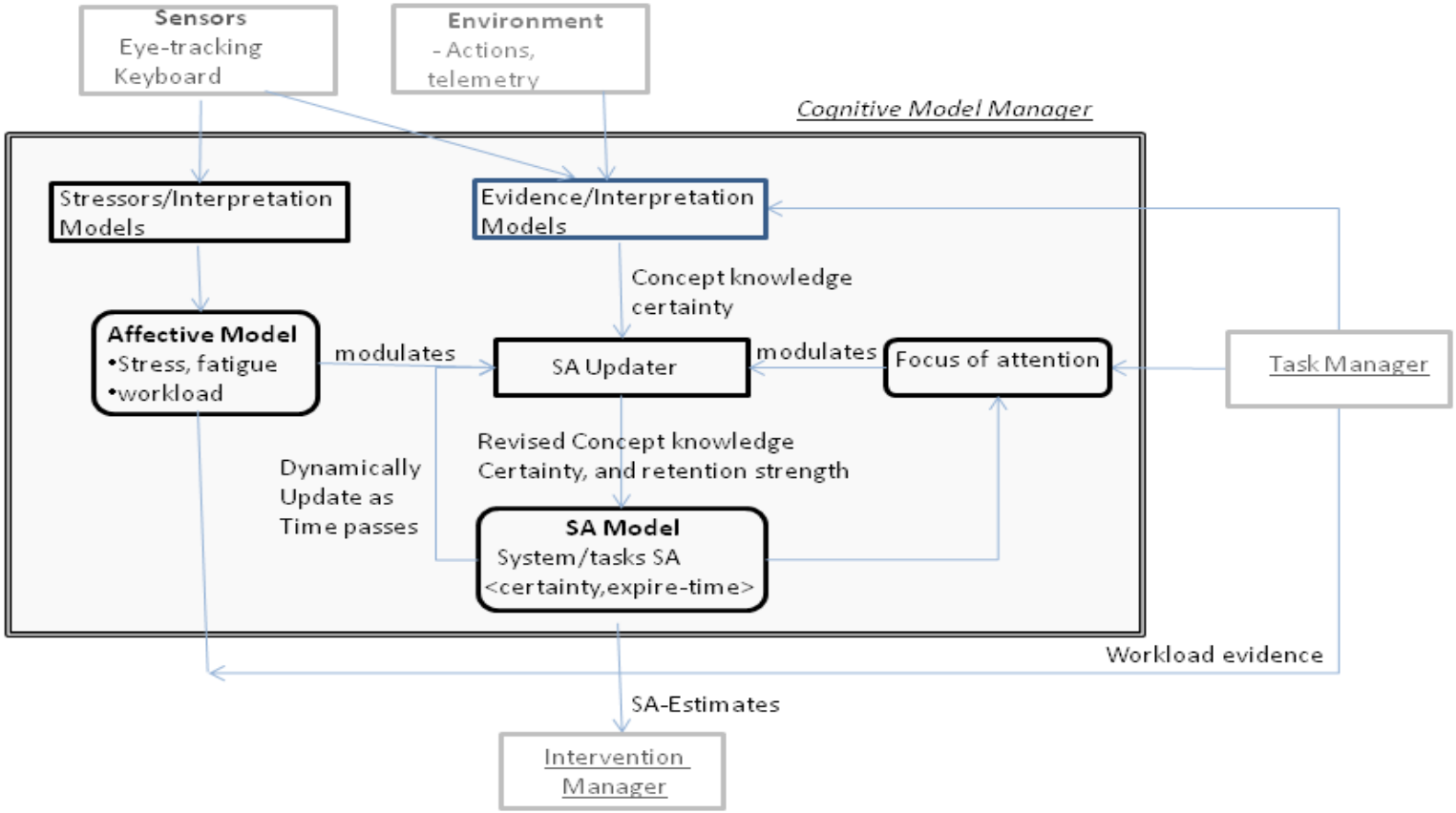


Cognitive Manager

- Uses sensor inputs and UI models to gather evidence about the concepts known by the user.



- Uses sensor input to estimate affective state.
- Uses short-term memory model to dynamically estimate user's SA decay over time.
- Uses affective model to modulate SA estimates.



Task Representation

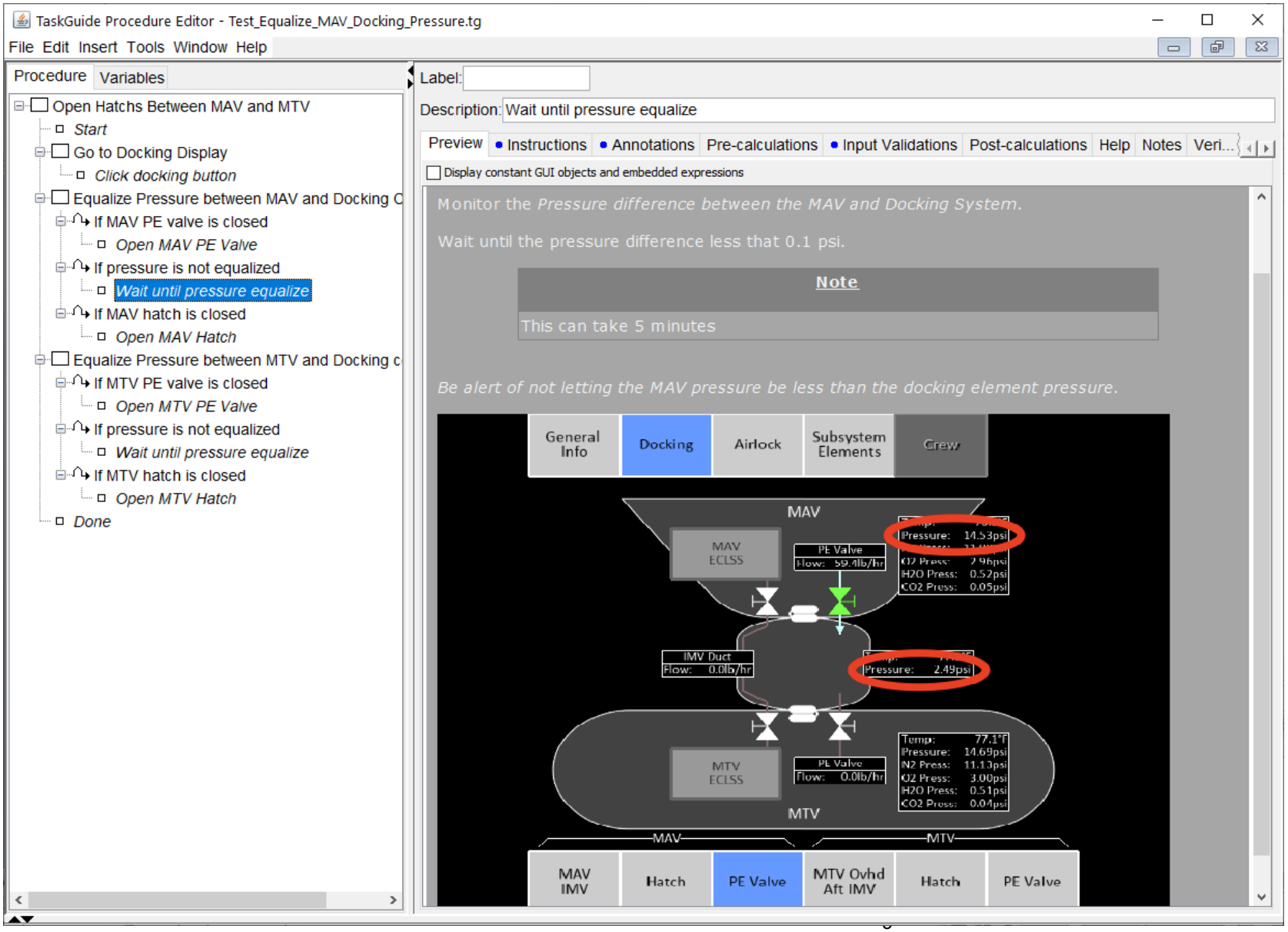
- Task representation:
  - Hierarchical representation of tasks with temporal constraints and deadlines.
  - Tasks have associated possibly multiple procedures.
  - Tasks and procedures are annotated with:
    - Workload estimates along different dimensions (motor, visual, auditory, frustration level, etc.), workload dimension as in the NASA-TLX,
    - Situation awareness requirements, cost of interrupting a task,
    - Actions to be taken by the operator.

- Knowing what the operator should be doing at a particular time is a difficult problem.

- Task monitoring assumptions:
  - The automated assistant knows of the scheduled tasks (OSTP).
  - Crewmembers use a *procedure viewer* to execute procedures.
  - ➢ The automated assistant knows what the user should be doing at any given time.

Procedure Viewer

- AVERT uses TaskGuide as a Procedure Viewer.



Task Annotations

- Task annotations are used by the task manager to detect problems during task execution.

Annotation Types	Examples
SA Requirements	*@sa-condition* range(combustionChamber.getVariable("temperature"),45,55)
User Actions	*@step-action* push_button('GeneralInfo','Docking')
	@step-action wait_for(equal_variables('MavCabin.Pressure', 'DockingModule.Pressure'))
Expected Responses	*@if-system-failure* PE_BLOCKAGE execute_procedure('Docking_Pressure_MAV_Blockage')
Termination Conditions	*@step-completed-condition* isOpen(systemModel.getValue("MavVestibulePeValve.Indicator"))

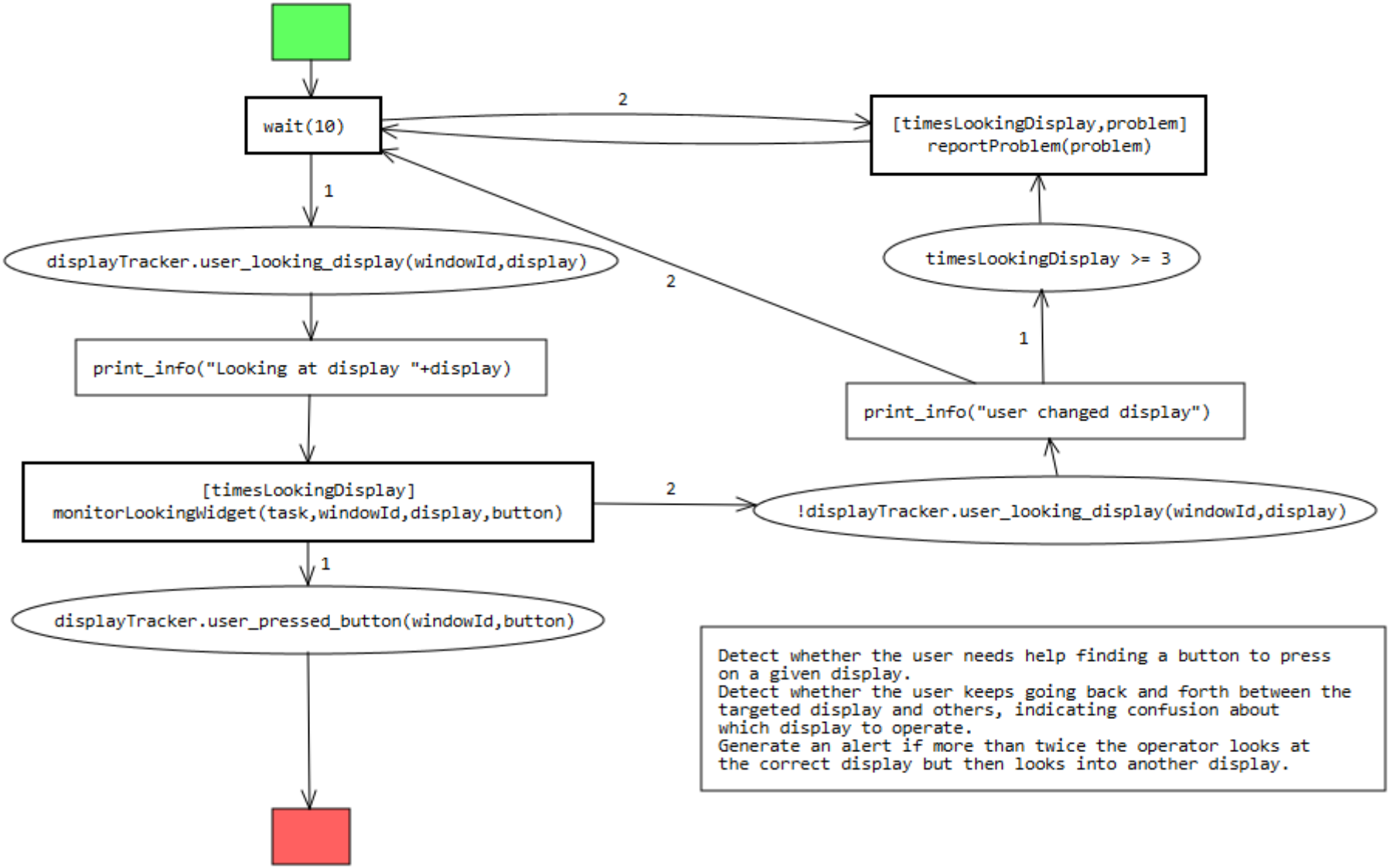
Task Monitoring

- Each task annotation triggers the creation of software monitors that detect user execution problems related to the annotation.

Annotation type	Problems
@sa-condition	Operator lost SA Operator is looking for related but incorrect information
@step-action push-button	Operator confused on what button to press Operator goes back and forth between displays Operator is about to press the wrong button Operator not doing work
@step-action wait-for C	Operator does not know that the wait-for condition C is true Operator is looking for related but incorrect information
@if-system-failure F do X	Operator does not recognize the occurrence of failure F Operator does not start task X Operator starts incorrect task Y
@step-completed-condition C	Operator does not recognize condition C Operator does not move to the next procedure step

Task Monitoring – Behavior Transition Networks

- Used Behavior Transition Networks (BTNs) to define the logic to monitor user behavior.
- BTNs: graphical programming language representing **hierarchical finite state machines**.
- The BTN below implements the logic to monitor a push-button action.



Monitoring Diagnosis Tasks

- Diagnosis tasks do not have a pre-scribed procedure to follow.

- AVERT's goal during a diagnosis task is to make sure that the operator crosschecks different failure hypotheses provided by an automated diagnosis system.

- We assume that there is a diagnosis system such that:
  1. For a set of C&W alarms, the diagnosis system automatically generates a set of fault hypotheses,
  2. For each hypothesis, the diagnosis system provides the following information:
     Indicators of the failed component,
     Indicators/symptoms supporting the hypothesis,
     Indicators/symptoms refuting the hypothesis,
     Other data pattern of interest associated with the faulty component, and
     The likelihood of the hypothesis being the root cause of the given alarms.

- The operator should check the hypotheses by looking at the hypotheses' indicators.
  - We assume that there is a time limit for the cross-checking task.

- The AVERT assistant ensures that the operator makes adequate progress during the cross-checking of the failure hypotheses.

Monitoring Diagnosis Tasks

AVERT recognizes three main states the operator is at during the diagnosis process:

- The operator is not diagnosing the failures:
  - The operator is not looking at any display showing any of the indicators associated with the alert explanations.
  - The operator's attention is on something other than diagnosing the alerts.
  - ➢ The AVERT assistant will call the operator's attention to one of the explanations for the alerts, opening displays as needed to show the explanation indicators.

- The operator is focused on an unexpected hypothesis:
  - The operator is working on indicators associated with a explanation but, given the current time and work done in crosschecking, AVERT determines that the operator should consider other explanations.

- The operator is not making progress crosschecking a hypothesis:
  - The operator is looking at the correct displays for an explanation but is not looking at the explanation indicators shown on those displays.
  - ➢ The AVERT assistant will highlight the indicators or give explicit hints as to what to look at.
  - ✓ A special case is considered when the operator looks at all of the explanation indicators—and then continues looking at those same indicators: we interpret that as the operator not having yet concluded whether the indicators confirm or rebut an explanation.

Diagnosis Performance Estimations

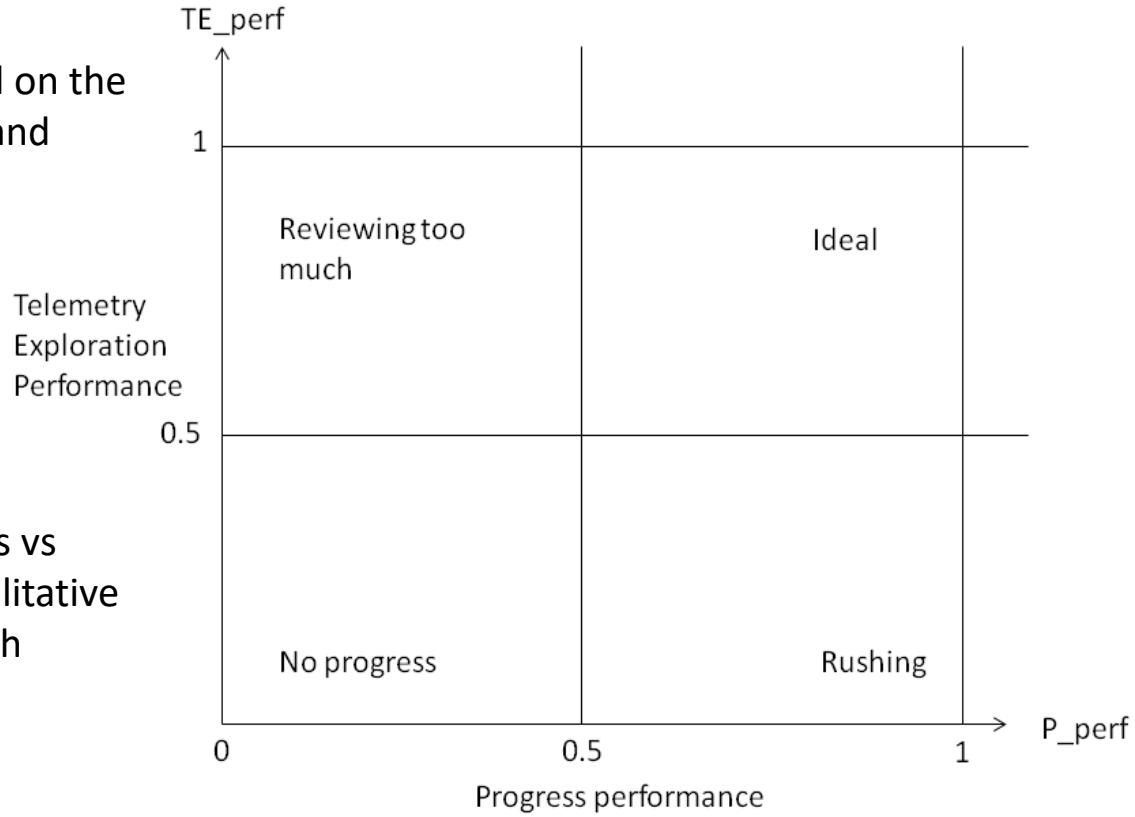
- AVERT uses a combination of two metrics to estimate the overall diagnosis performance:
  - Progress performance: A time-based metric where given a maximum time to diagnose N components, the diagnosis performance at a time t is based on the number of diagnosed components out of N.
  - Telemetry exploration: a function of the number of telemetry data pattern to explore

    $TE\_perf(n) = \max(1.0, (1/\alpha)\, n/N)$,

    where n is the number of reviewed patterns, N maximum number of data patterns to review, and α is a value in (0,1] representing the ideal percentage of data patterns to review.

- AVERT uses rules to intervene based on the problems revealed by the progress and telemetry exploration metrics.

- The figure shows a graph of Progress vs Telemetry Exploration and gives qualitative descriptions to the areas in the graph denoting performance problems.



AVERT Prototypes

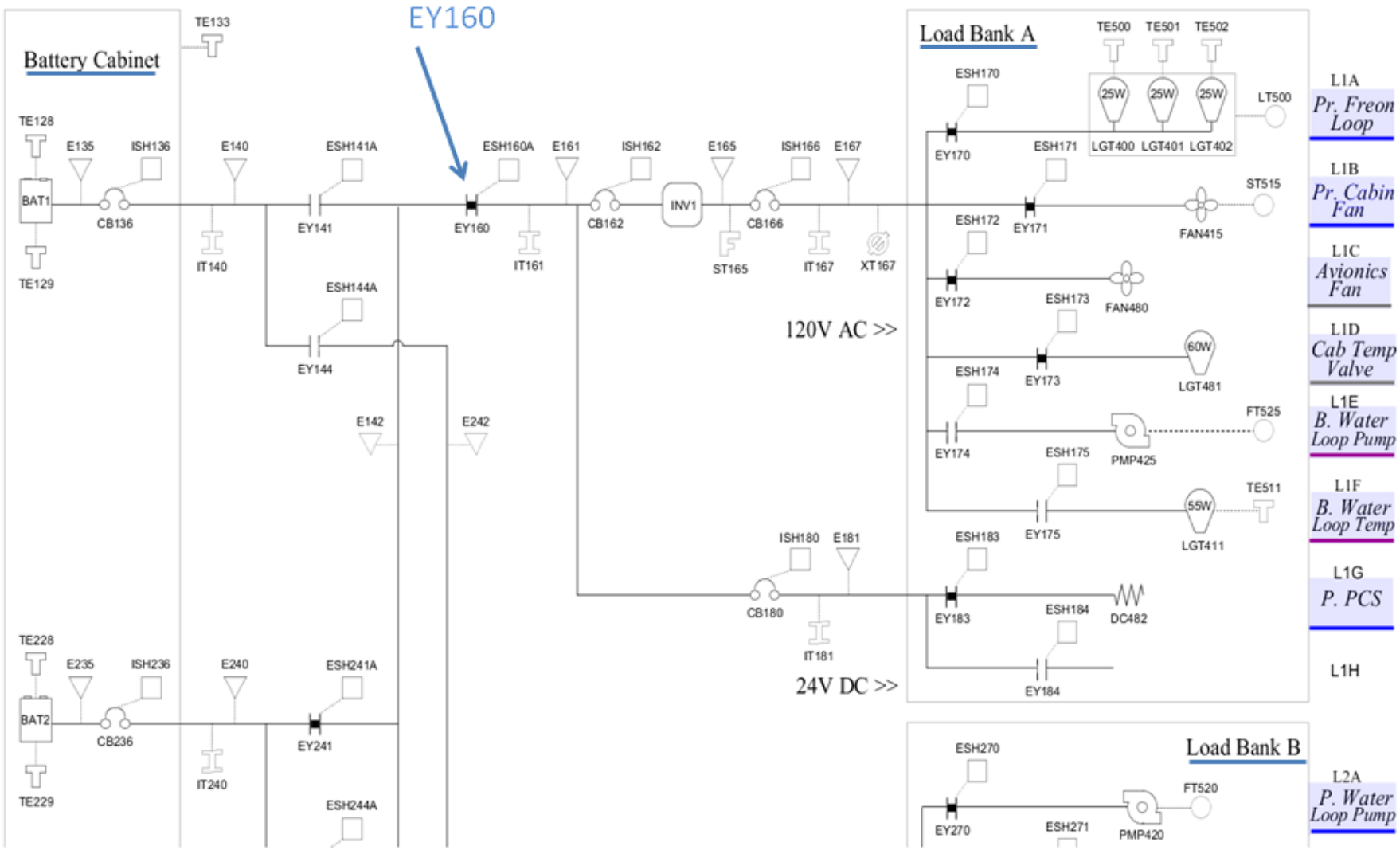
- Created AVERT assistants supporting the FDIR process in two domains:
  - MTV: Mars Transit Vehicle,
  - ADAPT: Advanced Diagnostics and Prognostics Testbed.

- For the MTV Domain:
  - Used NASA's JSC MTV simulator evaluated by NASA during the NEEMO-21 experiments.
  - Defined UI Model to:
    - Relate display coordinates to system variables,
    - Use eye-tracking data to derive SA, and
    - Virtually enhance the MTV simulator to highlight widgets in the display.
  - Demonstrated AVERT support in scenarios illustrating:
    - Procedure execution,
    - Multitasking,
    - Using pre-planned responses to failures

- For the ADAPT Domain:
  - Demonstrated AVERT assistant supporting the FDIR workflow in scenarios where the operator should respond to multiple C&W alerts.
  - Implemented procedure-reasoner to suggest procedures to respond to failures,
  - Implemented diagnosis system,
  - Implemented system re-configuration expert

ADAPT- EY160 Failure Scenario

- Relay EY160 fails: this relay controls the power distribution of Bank A.
- Multiple C&W alerts are received about lost functions, for example, cabin's pressure and temperature outside nominal limits.

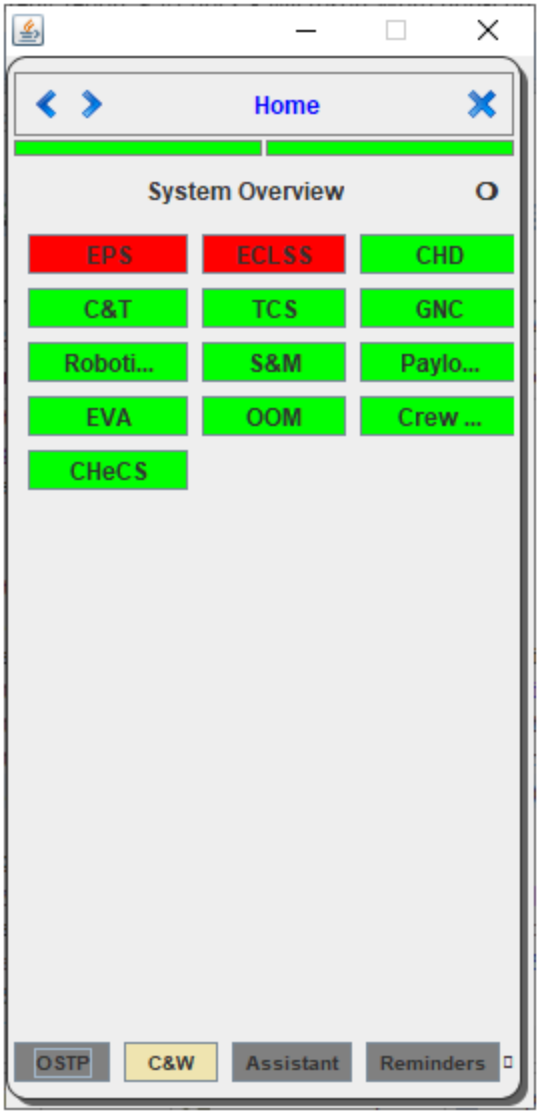


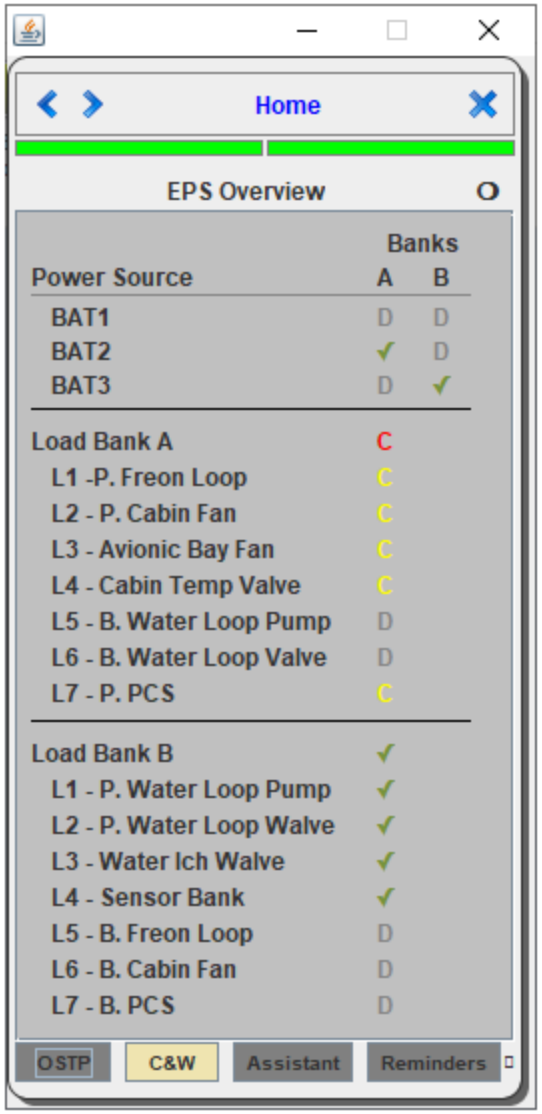
EY160 Failure Scenario

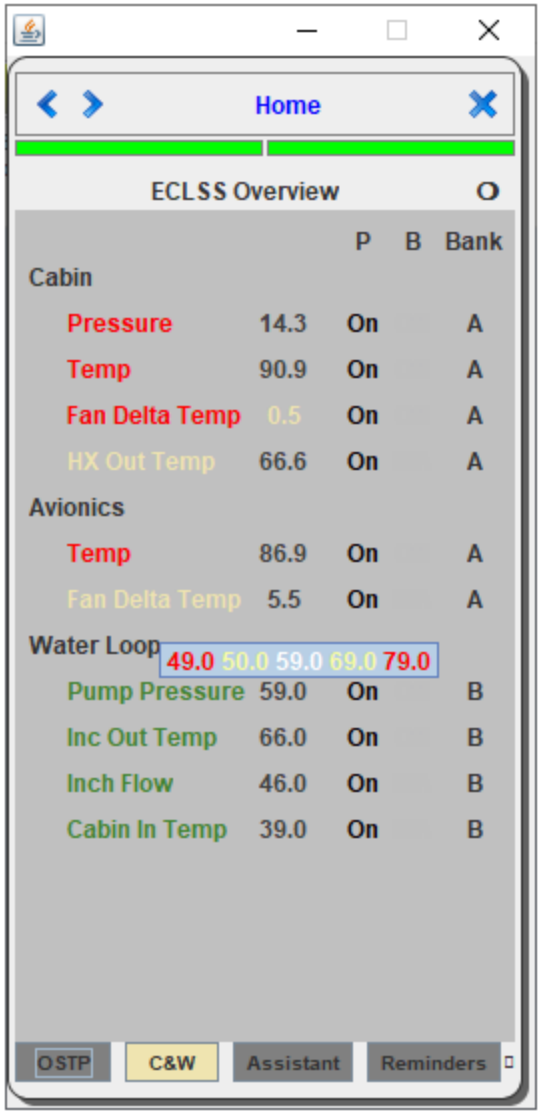
Operator Workflow	AVERT Assistant
1. Acknowledge C&W alerts	Monitors the operator interactions with the UI in order to detect whether the operator is aware of the alerts.
2. Bring the system to a safe configuration	Suggests that the operator execute the procedure to run critical loads using the backup loads. Checks that the operator is not working on **expected alerts** that might have occurred because of a procedure execution. Monitors execution of the 'migrate-critical-loads' procedure.
3. Diagnose failures	Monitors the system state and user tasks to decide whether the operator should initiate the diagnosis task. Suggests components that might have failed given the observed alerts. Makes sure that the operator reviews the relevant telemetry.
4. Schedule EY160 repair	Automatically creates a 'reminder' to repair EY160.
5. Repair EY160	Reminds operator to fix EY160.
6. Re-configure to nominal configuration	Detects that the system can be reconfigured to nominal configuration. Suggests procedure to do the re-configuration. Monitors the execution of the procedure.

Monitoring System Status

- User phone-like UI to monitor the state of the ADAPT System.

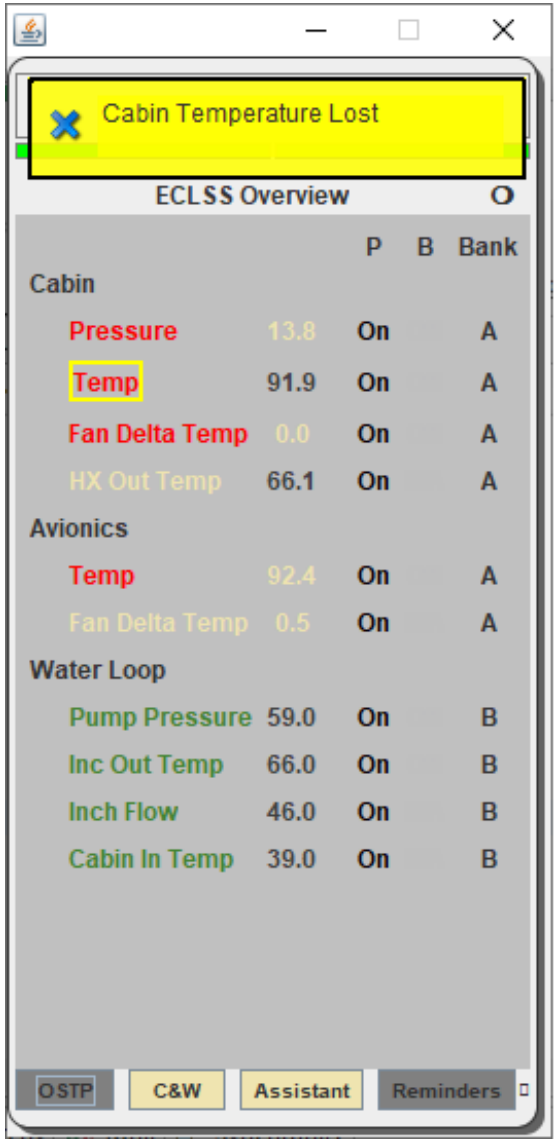






Monitoring Operator Alerts Recognition

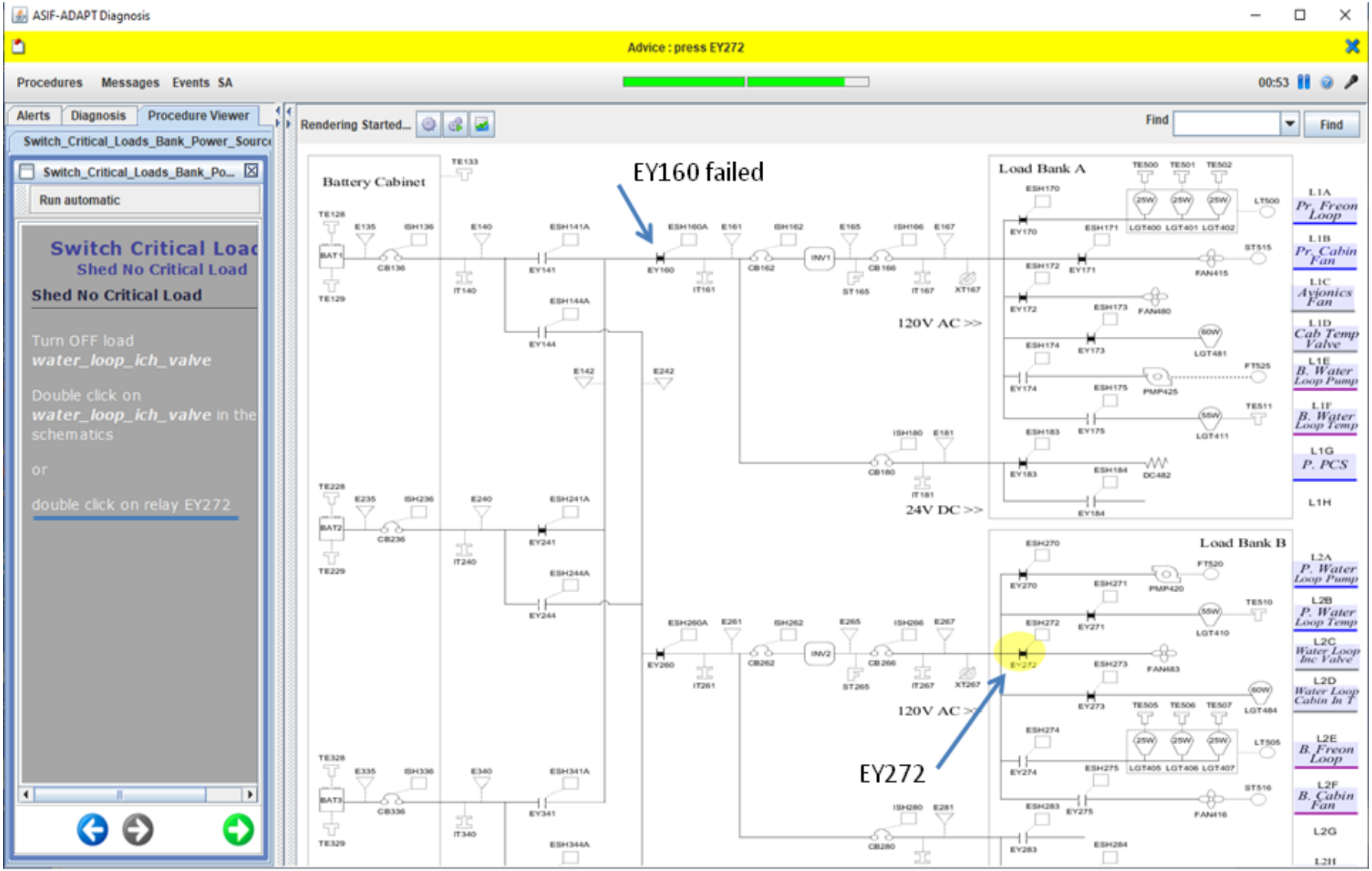
- If the operator does not acknowledge C&W alerts, AVERT informs the operator of a problem by using textual notifications and highlighting components in the UI.



Stottler Henke
Smarter Software Solutions

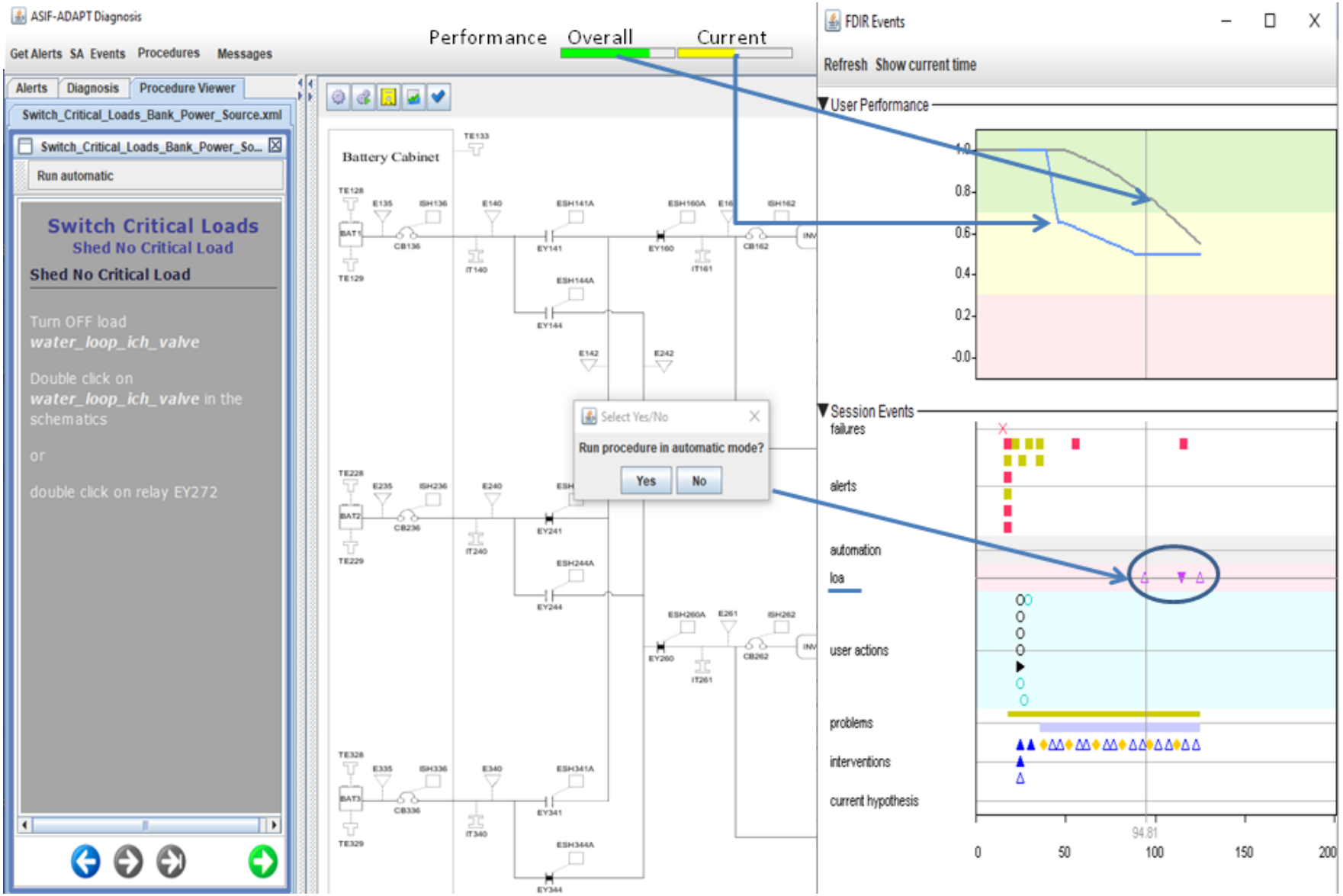
Responding To Failures: Procedure Selection and Execution

- AVERT suggests a procedure to execute, starts the procedure viewer, and monitors the execution of the procedure.
  - Procedure Interventions: textual advice, highlight controls to use, cross out controls not to use



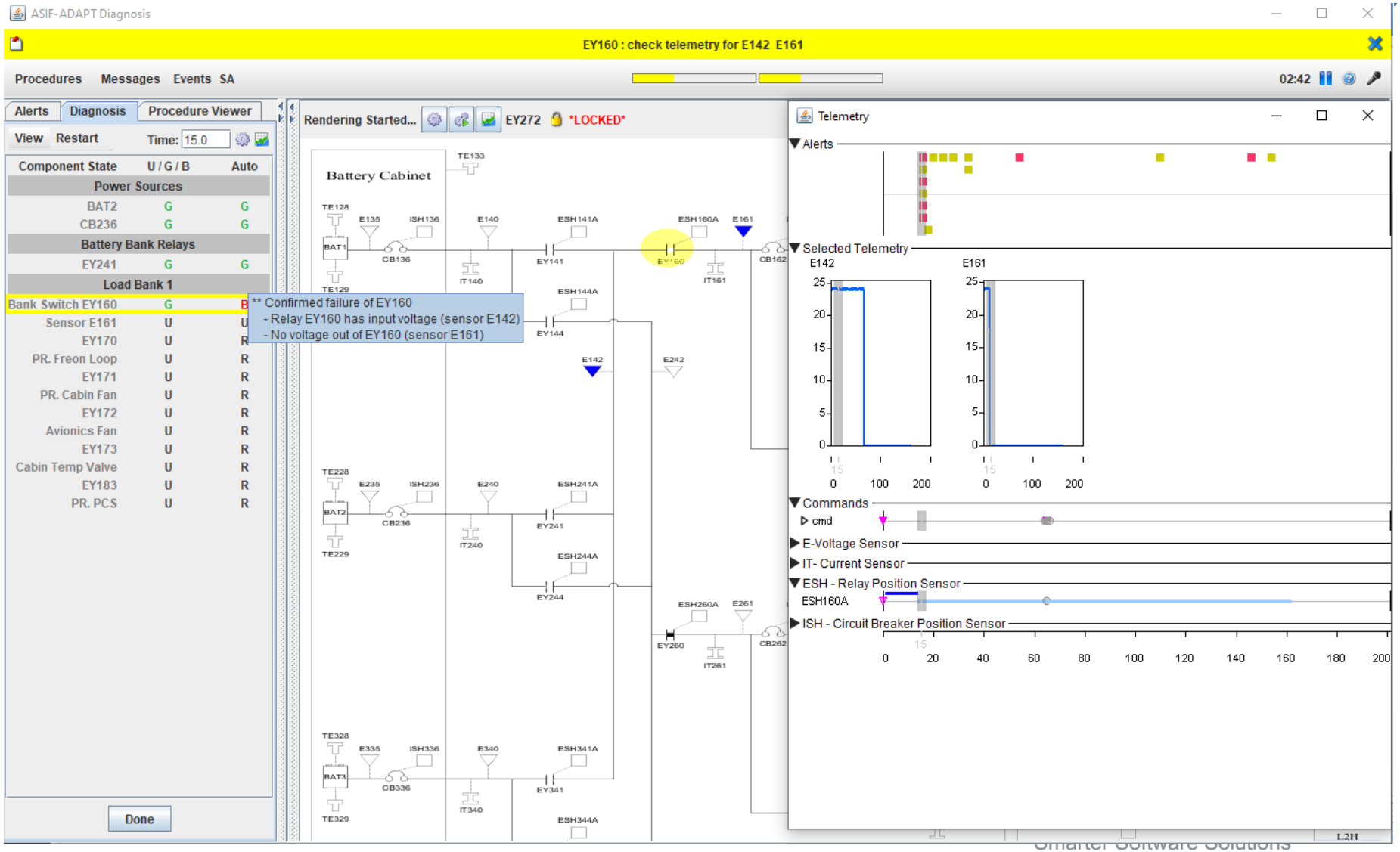
Changes to LOA

- If procedure execution performance is degraded, AVERT offers to execute the procedure automatically.



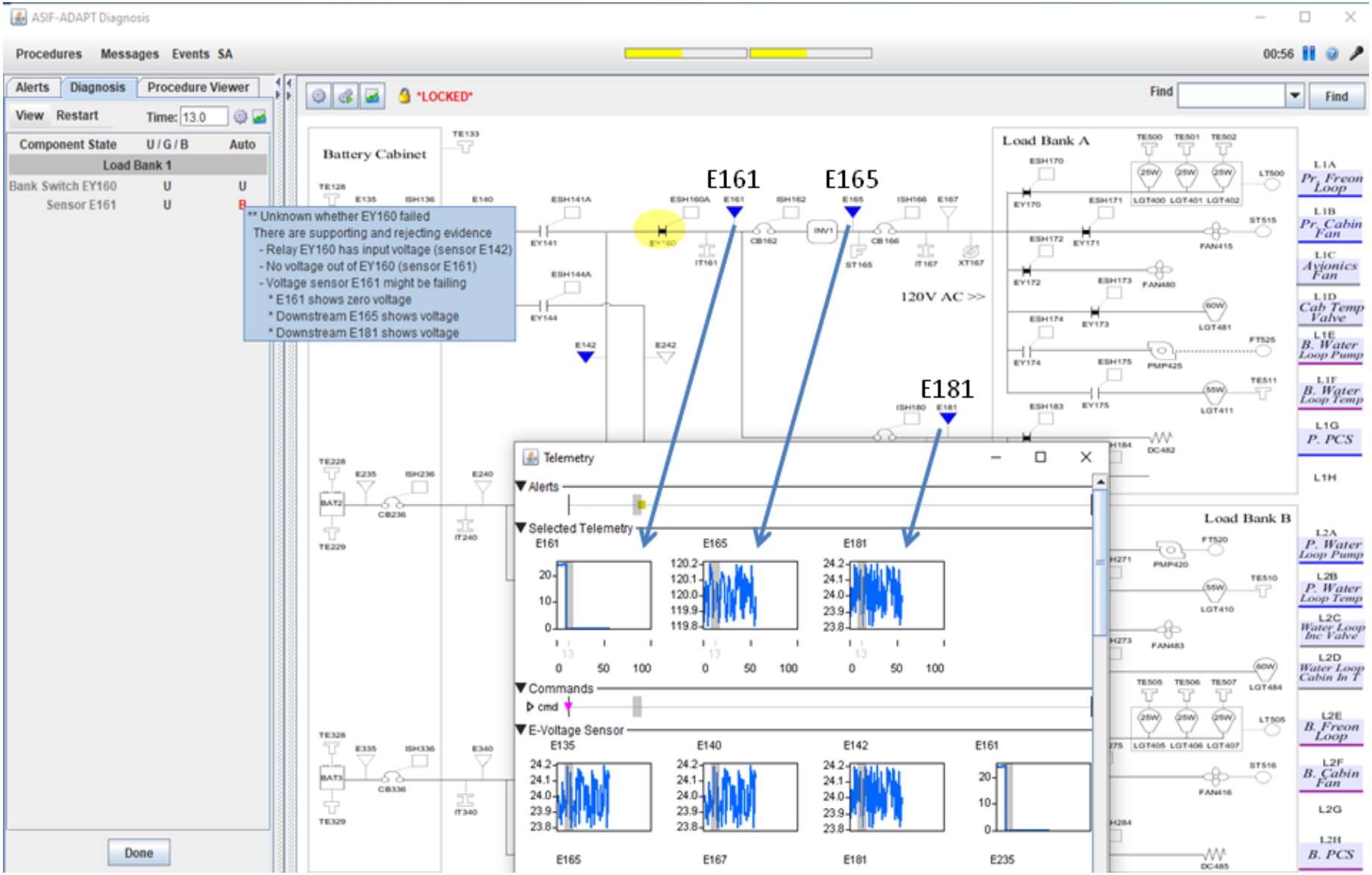
Diagnosing failures

- AVERT suggests components that might have failed.
- AVERT makes sure the operator reviews telemetry, specially when the operator decision disagrees with the automation decision (yellow row in the Figure).



Diagnosing Sensor Failures

- AVERT makes sure the operator considers sensor failures, and reviews a rationale indicating why a sensor might have failed.

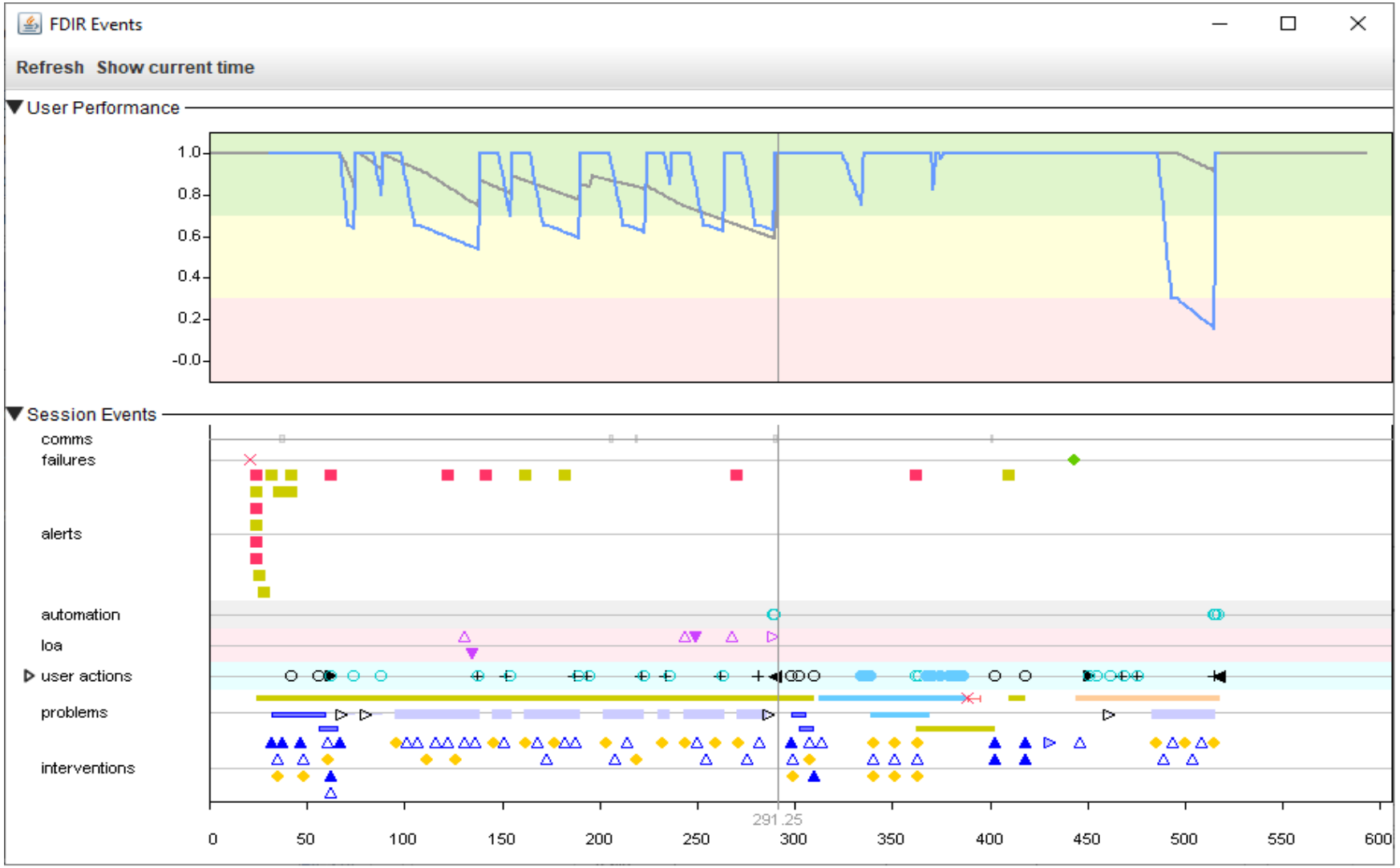


System Re-Configuration

- After relay EY160 has been repaired, AVERT detects that it is possible to bring the system to a nominal configuration, finds a procedure to do so, and suggests the astronaut to execute such procedure.
- AVERT uses a '*reconfiguration-expert*' software module that helps define the correct steps for the re-configuration procedure:
  - Depending on the system configuration, the reconfiguration procedure may be a simple one that only requires using some battery powering bank A and then turning loads on or off.
  - In some special circumstances, the procedure may require switching batteries before turning loads on and off.
    - In our example, Battery 3 is powering load bank B. Suppose that the relays connecting BAT-1 and BAT-2 to load bank A fail.
    - In this case, the only battery that could be used to power bank A is BAT-3.
    - The operator will need to connect BAT-1 to bank B and then connect BAT-3 to bank A, after which the loads can be turned on or off as required.
- Using expert modules is a way to gradually incorporate knowledge into the system and modularize such knowledge.
- Using rules was appropriate in our prototype, but constraint-satisfaction techniques might be the way to solve system configuration queries for larger systems.

Session Event Visualizations

- AVERT provides a session logging and replay facility to log telemetry, screenshots of the operator control displays, sensor data (e.g., eye-tracking, mouse movements), communications, operator actions, automation actions, user problems, and AVERT's interventions.



Summary

Developed working *prototypes* illustrating AVERT's FDIR support:

- Monitoring: AVERT tracks and monitors the user and the space system C&W system in order to recognize system and user states affecting the execution of operator tasks.
- Recognition: AVERT assesses the operator's awareness of system status and events, determines the priority of tasks to be performed, and advises whenever the operator is not working on high priority tasks.
- Reaction: AVERT advises on the criticality of impacted functions, suggests procedures to recover system functions, and monitors the operator's execution of such procedures.
- Diagnosis: AVERT suggests possible explanations for the observed failures, monitors that the operator crosschecks the explanations for the failures.
- Recovery: AVERT recommends recovery procedures, and monitors the user's execution of the procedures.
- Procedure execution: AVERT monitors the execution of procedures and produces visual cues when it detects that the operator seems confused, might be about to make a mistake or has not gathered information relevant to the procedure step.
- Change to LOA: When the operator's task performance is degraded, AVERT suggests changes to the level of automation, for example, by automatizing the procedure execution.

Future Work

- Prioritizing alerts:
  - It is not easy to determine automatically the importance and urgency of the problem signaled by an alert.
  - Astronauts use some knowledge of the system and interpretation of the alerts to decide in which order to respond to multiple alerts.
  - ➢ Identify knowledge sources and criteria that will enable an AVERT assistant to help astronauts prioritize their responses to alerts.
- Changes to LOA:
  - Our prototypes illustrate logic to change the level of automation when the user's procedure execution performance is degraded.
  - ➢ More elaborated logic is desirable for the assistant to take action based not only on the user's performance, but also the risk if no action is taken, and the time available to respond.
- Dialogs between the operator and the assistant:
  - In the current prototypes, there is not much communication between the operator and the assistant. The assistant gives advice based on its estimate of problems the operator is having with a task execution.
  - ➢ In a more realistic setting, the operator should be able to use a spoken-language interface to ask the assistant for help, request the assistant to do some tasks or show some relevant information, and ask for an explanation for any information provided by the assistant.

Future Work

- Suggesting procedures to respond to multiple failures:
  - The prototypes use a table lookup method to find procedures to respond to failures.
  - This method has limitations especially when single-failure procedures have to be sequenced to respond to multiple failures, or when an existing procedure needs to be adapted to respond to an unexpected situation.
  - ➢ Use more generic techniques based on model-based reasoning, like Stottler Henke's MFRP (Multi Failure Response Procedure), to automatically create procedures to respond to failures and better support the FDIR process.
- Use sensors to estimate workload or affective state:
  - We did not work on using sensor inputs to estimate user's affective state (e.g., fatigue, stress, boredom, confusion), and then relate task performance to affective state in order to determine an appropriate intervention.
  - ➢ Operator's affective and physical state provides further discriminators to explain operator's task performance. An AVERT assistant should use this information to find the appropriate responses to improve work efficiency and safety.
- Diagnosis support:
  - Our prototype illustrates how AVERT could help an astronaut to review automated diagnosis hypotheses for failures in the ADAPT's EPS system.
  - This EPS system has many components but it is relatively small when compared to systems used by NASA (e.g., ISS, Orion Crew Vehicle).
  - ➢ How will an assistant help the astronauts to diagnose failures in complex systems, where failures may impact different subsystems (e.g., EPS, ECLSS, ATCS, GNC)?.