# Developing an International Space Station Curriculum for the Bootstrapped Learning Program

Jeremy Ludwig, John Mohammed, and Jim Ong
Stottler Henke Associates, Inc.
951 Mariner's Island Blvd.
San Mateo, CA 94404
650-931-2700
ludwig, mohammed, ong @stottlerhenke.com

*Abstract*—DARPA's Bootstrapped Learning (BL) program is aimed at advancing the state of the art in *instructable computing*. Two objectives of this program are developing a general electronic student that makes use of machine learning algorithms to learn from the kind of focused instruction typically provided by a human teacher and creating a repository of automated curricula that can be taught to the student. This paper focuses on the second objective, describing a curriculum developed for the BL program to both instruct and test the student that places the electronic student (eStudent) in the role of an International Space Station (ISS) flight controller. The eStudent is taught how to detect and diagnose single-fault problems within the thermal control system of the ISS. During each lesson, the eStudent interacts with an ISS simulator to review alerts and access telemetry values. To obtain greater visibility into its diagnostic reasoning, the eStudent is trained to create an external representation of its reasoning about the current problem – a diagnostic rationale. This includes describing potential problems, hypothesizing possible events and states, positing possible causal explanations as rationale assertions, seeking evidence for or against these assertions, projecting possible risks, and using possible risks to focus attention when developing a rationale. In addition to describing the curriculum developed as part of the first year of the BL program, we also describe some of the future directions we will investigate as part of the second year. [1,2]

**TABLE OF CONTENTS**





## 1. INTRODUCTION

> "Wait! Haven't we learned by now that thinking machines combined with spacecraft are very, very bad?" [1]

The Bootstrapped Learning (BL) program is a DARPA initiative aimed at advancing the state of the art in *instructable computing*. One objective of this program is to create an *electronic student* (eStudent) that can learn from the kind of focused instruction typically provided by a human teacher to a human student.[2] The eStudent uses a variety of machine learning algorithms to learn how to solve tasks in an arbitrary domain, where the teacher instructs with a set of formally defined natural instruction methods (NIM). For example, in using the *teaching by example* NIM the instructor may make gestures at relevant problem attributes, demonstrate actions in the domain, and provide explanations for why actions were taken. For example, an unmanned aerial vehicle operator may instruct the eStudent on how to recognize truck-to-truck transfer by showing the system specific examples (both positive and negative) of the activity and annotating these examples to point out the most important aspects.[3] This type of machine learning has significant implications for the development of intelligent applications in nearly any domain.[4] Instead of learning from large amounts of data or having computer programmers work with subject matter experts to manually encode knowledge in an expert system, an eStudent would learn from a limited amount of instruction provided directly by the expert.

There are a number of research areas being investigated under the multi-year, multi-organization BL program. Roughly divided, one team is charged with developing the eStudent while another develops the curricula that the eStudent is expected to learn and a general electronic teacher for delivering the curriculum. The two teams work together to define the formal language and the natural instruction methods used by both teams (either in teaching the curricula or learning from them). This paper focuses on the curriculum development aspect, describing one curriculum in the curriculum repository.

The curricula fill two roles in the BL program – to instruct the eStudent and to test the eStudent to see if it learned the

[1] 978-1-4244-3888-4/10/$25.00 ©2010 IEEE
[2] IEEEAC paper#1076, Version 6, Updated 2009:11:30

concepts correctly. Each curriculum is organized in the same manner, called a ladder, where simpler concepts are taught in early rungs and more complex concepts taught in later rungs. Each rung in the ladder teaches a single concept, though this concept may be supported by multiple NIM lessons. For example, a single concept such as how to recognize that a truck is parked could be taught in four lessons: one by telling, two by example, and one by feedback. Additionally, each curriculum contains background knowledge that is provided to the student. The background knowledge contains the basic concepts and procedures from the domain that are required for learning to occur. That is, the basics are programmed and the eStudent is expected to learn the rest as it works its way up the ladder. In Years 1 and 2 of the BL program, curricula are formally defined and instruction is automated. This allows for the algorithms behind the eStudent to be developed and tested in an efficient manner.

The BL program currently contains five curricula. The first two, Blocks World and RoboCup [4], are domains commonly used as testbeds in artificial intelligence research. Three additional diversity domains have been constructed as part of the BL program to provide challenges for the eStudent more akin to those encountered in the real world. This paper focuses on the development of one of these diversity domains, which teaches the eStudent to recognize and diagnose faults in the thermal control system (TCS) of the International Space Station (ISS).

## 2. CURRICULUM OVERVIEW

Flight controllers are personnel who aid in the operations of a space flight at facilities such as NASA's Mission Control Center in Houston. They sit at computer consoles and use mission operations software to monitor telemetry data from the space vehicle and review automatically generated alert messages to detect and diagnose problems in real time. They also determine and execute maintenance and recovery actions by sending commands to computers onboard the space vehicle for execution. Each controller is an expert in a specific area and retains flight control responsibility for that area.

Diagnosis is one of the skills required of ISS flight controllers. ISS diagnosis is a complex skill that requires considerable knowledge, aptitude, and experience. Flight controllers detect possible problems by noticing symptoms in the form of alert messages and abnormal telemetry data values. Using their diagnostic reasoning skills and understanding of ISS systems, they hypothesize system state conditions and events that might have caused the symptoms, and they search for evidence, such as telemetry data and crew reports, that support or refute the hypotheses until they converge upon a best explanation or diagnosis.

A distinctive characteristic of ISS diagnosis is that it is carried out in real time in the context of a mission. That is, unlike a typical mechanic who diagnoses and repairs equipment that is taken out of service, a flight controller monitors and controls the ISS while it is in flight. He or she must prioritize diagnosis and recovery actions by considering the potential impact of each problem on the crew's safety, vehicle health, and mission objectives (in that order). Because the ISS is a very complex system, it is common for many off-nominal conditions and abnormalities to exist at any particular time, and not all problems can be or need to be diagnosed and resolved. The likelihood and severity of each potential problem affects the urgency and importance of diagnosing and recovering from it.

The ISS curriculum places the eStudent in the role of an ISS flight controller who must detect and diagnose problems within the thermal control system (TCS) of the International Space Station (ISS). During each lesson in this Bootstrap Learning curriculum, the eStudent interacts with an ISS simulator to review alerts and access telemetry values. To obtain greater visibility into the eStudent's "thinking", the eStudents are trained to create an external representation of part of their reasoning – a *diagnostic rationale.* This mirrors the practice of having students "think aloud" in front of the teacher as they perform each exercise. This curriculum draws upon instructional methods developed by Stottler Henke for NASA while designing ADEPT, an intelligent tutoring system for training NASA flight controllers (i.e. human students) to diagnose ISS problems by reasoning about ISS systems and their interactions.

Diagnostic rationale is the collection of assertions about observations, system states and the causal and evidentiary relationships among them that are asserted by students/eStudents to describe their diagnostic reasoning process. Each assertion corresponds to a hypothesized past, present or future state or event, such as a component fault, parameter value, or component state, or to an observation such as a telemetry reading, human observation, or alert. Each assertion has an associated *qualitative degree of belief* which can be *true* (strong belief in the assertion), *uncertain*, or *false* (strong belief in the assertion's negation).

Assertions can be *supported or refuted* by other assertions through link assertions that express causal relationships ("can lead to"), negative evidence ("refutes"), or positive evidence ("supports"). The antecedent and consequent of each *causal link* must be a state or event. If Assertion A specifies that a state or event of type *a* might have occurred, and states or events of type *a* are known to tend to cause events or states of type *b,* and Assertion B specifies that a state or event of type *b* might have occurred, then a belief in A supports a belief in B (via projection), and a belief in B supports a belief in A (via abduction, or explanation). For example, if a coolant pump failure (*a*) causes a drop in coolant pressure (*b*), a belief that the pump failure occurred (A) supports a belief that the coolant pressure drop has or

will occur (B), and a belief that a coolant pressure drop has occurred (B) supports the belief that the coolant pump failure has occurred(A). Note that support for an assertion does not necessarily imply strong belief in the assertion.

A *positive evidential link* represents support for the occurrence of a state or event provided by a piece of evidence. For example, a telemetry reading that indicates low coolant pressure is evidence for the state of low coolant pressure. A *negative evidential link* represents a partial refutation for a state or event by a piece of evidence. For example, a telemetry value that indicates nominal power availability refutes a hypothesis that there was a power generation failure.

The NASA ADEPT system enabled the student to construct a rationale by drawing and configuring nodes and links that represented assertions and their relationships. Assertions are displayed as node icons in the graphical display, and links are displayed as line icons that connect related assertion (node) icons. Thus, during each exercise, the student carries out interleaved actions that (a) request and view telemetry data, using simulated mission operations software, to assess the state of the ISS, and (b) incrementally express his or her rationale by drawing and configuring node and link icons.

Figure 1 shows an example rationale in the ADEPT system, illustrating how information about assertions and links is encoded graphically as node and link icons. It describes chains of causally dependent states starting with an over-current condition in an Remote Power Control Module (RPCM; the circular node labeled "LA2A C I High" in the diagram) leading to a circuit breaker trip ("CB Open"), loss of power to the Moderate Temperature Loop (MTL) nitrogen interface assembly ("MTL-NIA Off"), abnormally low coolant pressure ("Low Coolant P"), shutdown of the coolant pump ("Pump Off") and finally, inoperation of the MTL ("No MTL").

These state or event nodes are linked by causal links (dark grey lines) and are supported by positive evidential links (blue dashed lines) from telemetry assertions (square grey nodes) and alert assertions (yellow/orange triangle nodes). The rationale also shows an alternate possible explanation for the coolant loss alert (the "Coolant Leak" node). This node is drawn as a hollow circle with an "X" through it to

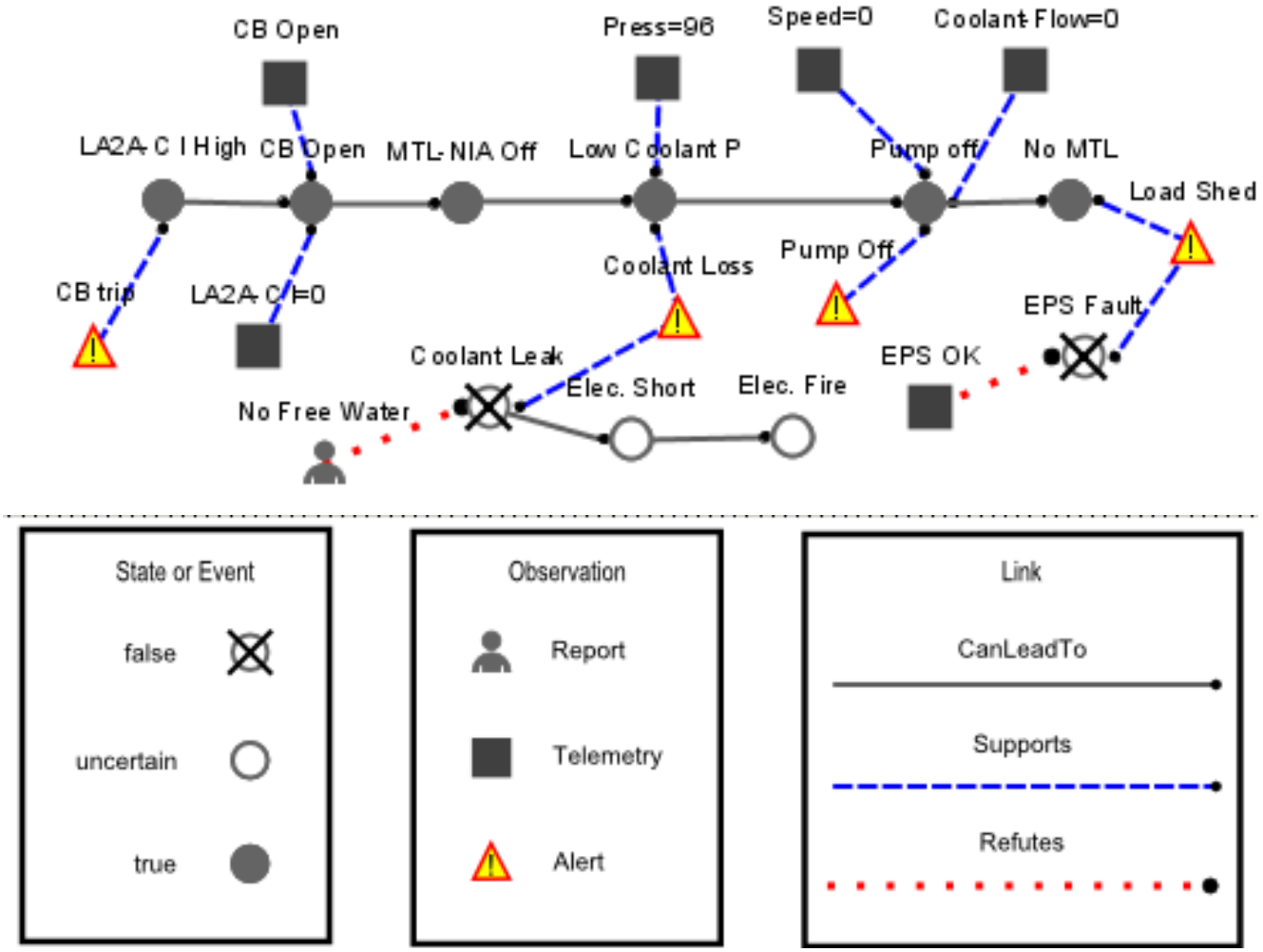


**Figure 1.Graphical Display of a Rationale for a Specific Scenario**

indicate that this explanation is not believed to be true. This node also has causal links to potentially dangerous consequences of a leak, namely electrical short circuits ("Elec. Short") and electrical fire ("Elec. Fire").

A report assertion (the stylized human upper body icon labeled "No Free Water") provides negative evidence (the red dotted "Refutes" link) for the coolant leak hypothesis. Similarly, an assertion regarding a fault in the Electrical Power System (EPS) Generation system ("EPS Fault") provides an alternate explanation (that is not believed) for the load shed alert for which a telemetry assertion ("EPS OK") provides negative evidence. Reports, telemetry readings, and alerts are observations which, by definition, are assumed to be true and require no evidential support of their own to be believed. That is, the student can assume that any perceived report really did occur, although determining the import of each report requires analysis and interpretation. This diagram represents the scenario author's description of the diagnostic rationale for the scenario.

Using an analogous, though non-graphical rationale framework, the BL ISS curriculum teaches the eStudent how to carry out a variety of primitive diagnostic tasks and communicate its diagnostic rationale. Each primitive diagnostic task requires the eStudent to interact with an ISS simulator to observe the current state and to assert different types of rationale. An overview of the entire BL ISS curriculum is given in the following section.

## 3. ISS CURRICULUM

The ISS curriculum is comprised of four parts, each of which is discussed below. The first is a simple ISS simulator that generates alerts and provides observation about the system state. The second part is the background knowledge given to the eStudent. This knowledge is pre-encoded in the BL representation language and allows the curriculum to teach an eStudent that already has some knowledge of the problem domain. The third part of the curriculum is the instruction itself. That is, the ladder rungs that teach the student how to use the material encoded in the background knowledge to create a diagnostic rationale. The fourth part of the curriculum is the graduation exam, which describes what the eStudent should be able to do after the entire curriculum is learned.

*Simulator*

The four major capabilities that the ISS simulator provides for the initial curriculum are: signaling alerts, providing access to telemetry (system parameter) values, providing methods for creating a rationale object, and providing methods to add assertions to a rationale. Alerts, requested telemetry values, and the developed rationale are sent to the eStudents as *percepts*, which are computer representations of the observations made by the eStudent. The simulator commands available to the eStudent are limited to requesting telemetry, creating rationale objects, and creating assertions of various types.

In the initial version of the ISS Simulator, the alerts and telemetry values are fixed for each lesson. That is, at the beginning of the lesson, all alerts for that lesson are presented to the student as percepts. The telemetry values are also fixed at the start of the lesson, though the student will need to request these percepts individually. The only aspect of the simulation that changes over time is the rationale constructed by the eStudent.

The eStudent is taught to create a record of the specific pieces of knowledge that pertain to the problem being diagnosed, in the form of a network of assertions constituting a diagnostic rationale. This rationale representation, as part of the simulator, is external to the eStudent and perceptible by the eStudent – much like writing on a whiteboard. Thus, the automated curriculum instructor can also manipulate the rationale as a means of demonstrating the process that the eStudent should be learning, can describe the actions that the eStudent should be making in communication acts, and can gesture at parts of the rationale, as well as at parts of the information available to the eStudent from the simulation (e.g., point to parts of an alert).

The ISS Simulator is implemented as a Java program and integrated with the Bootstrap Learning Framework through the construction of Java interfaces and bean objects. The simulator presents alerts, telemetry data, and the eStudent's rationale as percepts, and accepts telemetry requests and rationale actions from the student. Each percept type defined in this curriculum, and any complex subtypes, has been implemented as a Java bean class. Additionally, all simulator commands have been defined in the ISS Simulator Java interface. Any non-primitive arguments for simulator commands are also specified as Java bean objects. The ISS Simulator is able to run with no UI or with a UI designed for monitoring eStudent progress visually.

*Background Knowledge*

Unlike actual flight controllers who begin training with a college-level engineering education, we designed the curriculum to teach basic principles that could be learned by a high school student who possessed a lay understanding of diagnosis and devices acquired through ordinary life experiences. This decision eliminated the need to encode large amounts of college-level engineering knowledge as eStudent background knowledge, reducing the amount of background knowledge to a manageable level. Instead of encoding a quantitative model of the ISS components and their states and relationships, we encoded a qualitative model that describes relationships such as *A can lead to B* and *X refutes Y.* This decision was made based on the goals of the Bootstrap Learning program for the ISS curriculum. This tradeoff allowed us to focus resources on developing

the curriculum to be taught to the eStudent rather than focusing on encoding a more complex model of the relevant ISS systems. The goal was to advance scientific knowledge in the area of instructable computing, not to create a real-world diagnostic assistant for the ISS, so this loss of realism was deemed acceptable.

That said, the simulator provides the eStudent with the same kinds of indications of system health and operation that are available to crew and/or flight controllers. This primarily consists of Caution & Warning Alerts and telemetry displays. Since the focus of the curriculum is on the task of diagnosis, we assume that the eStudent's background knowledge includes information about causal relationships among system states and events (i.e. the student is not required to learn models of how the system is expected to work). For example, the background knowledge contains statements such as an (i) *open check valve can lead to a failed pump*, (ii) *alert A0001 supports failed pump* or (iii) *pump speed should be between 12000 and 18000 rpm* . These statements are used to create the diagnostic rationale based on the given alerts and the telemetry observations that the student requests. The background knowledge also contains information that is used by the student to prioritize diagnostic search, such as (i) *a payload shutdown of a certain type could lead to loss of mission function* while (ii) *the loss of an avionic subset could lead to loss of vehicle*.

*Curriculum Overview*

This section describes each of the units and rungs in the curriculum. Some of the rungs include lessons that present an initial rationale and require the eStudent to add or modify the rationale, based on the eStudent's information gathering and analysis. The overall structure of the curriculum is shown in Figure 2. Each rung is taught to the eStudent using formal NIM (natural instruction method) contracts, where most lessons are presented using more than one NIM. Additionally, each rung contains a performance test to verify that the eStudent learned the concept presented in the rung.

Alerts: Rungs 1.1-1.3 teach basic concepts about ISS alerts. Specifically, these rungs teach how to recognize abnormal alerts and how to create an assertion in the rational object about abnormal alerts.

Evidence: Rungs 2.1 and 2.2 teach the eStudent to hypothesize that certain states or events have (or have not) occurred. That is, if an observation (alert, telemetry) O has been made and the background knowledge contains a statement that O supports/refutes state E then create a state/event in the rationale. Rungs 2.3 and 2.4 teach how to create the supports/refutes link between the observation and the state.

Causality: Rung 3.1 teaches the eStudent how to assert that two state assertions are causally linked, based on the *can lead to* statements in the background knowledge. Rung 3.2 provides instruction on projecting causality, where if A can lead to B but no assertion has been made about B then the learner should assert state B and search for evidence that supports/refutes this hypothesized state. Rung 3.3 teaches the abductive version of this, where B is an asserted state and A is the state to be asserted.

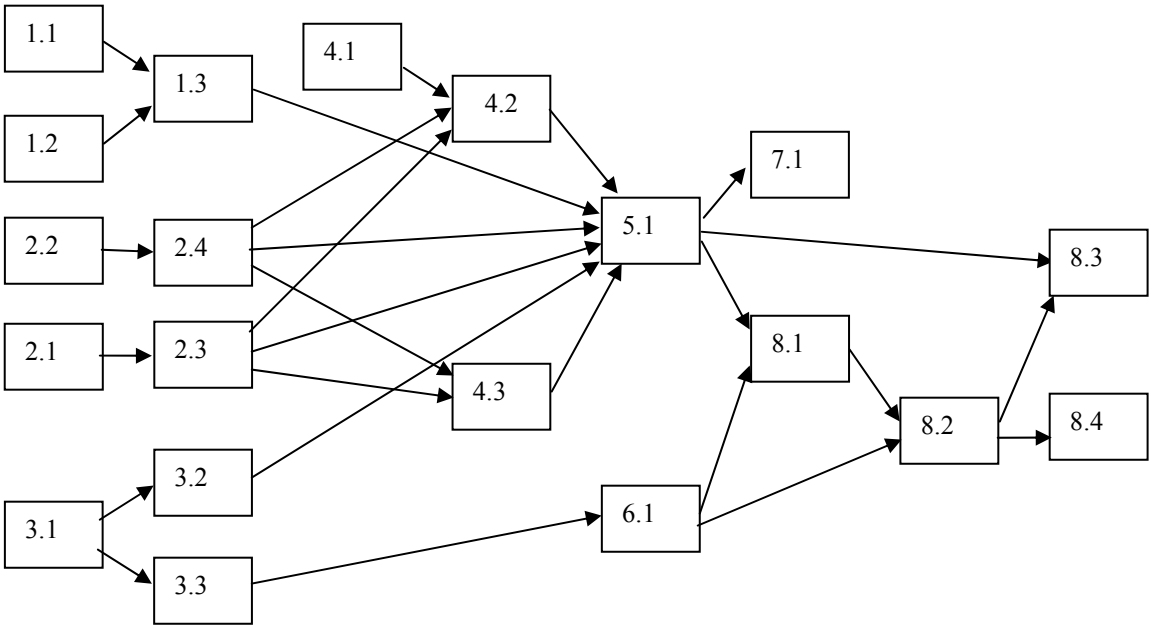


**Figure 2. Curriculum prerequisite structure.**

Seek Evidence: Rung 4.1 teaches the eStudent to request telemetry values relevant to the hypothesized states or events, and to link these observations. Rungs 4.2 and 4.3 teach how to create supports and refutes assertions based on telemetry observations and abnormal alerts (respectively).

Diagnosis: Rung 5.1 brings all of the preceding rungs together, teaching the eStudent how to create a complete diagnosis based on an initial set of alerts given by the simulator.

Risks: Rung 6.1 teaches the eStudent to extend its rationale to identify risks by adding causal consequents of a state or event, where some of the states / events are marked as undesirable (i.e. lead to loss of mission, vehicle, or life).

Belief: Rung 7.1 teaches the eStudent to modify the certainty associated with the states or hypothesized events in its rationale.

Domain specific heuristics: Rung 8.1 teaches the eStudent that diagnosis (as taught in Rung 5.1) should be modified so that the progeny of risky hypothesized states / events should be identified before searching for supporting evidence or possible causes. Rung 8.2 instructs the student on how to assign a qualitative notion of risk to a diagnostic hypothesis, based on the maximum risk associated with its progeny. Rung 8.3 modifies the diagnosis algorithm yet again, this time so that the algorithm investigates the most risky outcomes first. Rung 8.4 presents a very specific heuristic that further focuses risk-based evidence seeking based on the immediacy of the risk.

The structure of this curriculum requires that the student apply knowledge gained in earlier rungs to learn the concepts in later rungs. For example, learning diagnosis in rung 5.1 requires that the eStudent apply knowledge about creating alert assertions for abnormal observations learned in rungs 1.x (i.e. rungs 1.1-1.3), identify and link supported causes as learned in rungs 2.x, identify and link possible causes as learned in rungs 3.x, and seek additional evidence as learned in rungs 4.x.

The curriculum also has several instances in which to learn a new concept, the eStudent must reevaluate the representation of concepts learned earlier. For example, the concept of abnormal observation learned in rung 1.1 must be modified when rung 1.2 is learned. More significantly, the concepts learned in Rungs 1 through 4 will likely need to be modified when Rung 5 is learned, as it is unlikely that the eStudent will have represented these concepts in a fashion that directly supports the way they will need to be applied in rung 5.1. Similarly, the eStudent will have to modify the representation of the diagnostic algorithm learned in Rung 5.1 when later rungs are learned, to insert the subtask of identifying risks as learned in rung 6.1 (rung 8.1), and again to change the order in which states are processed in rung 8.3. The algorithm for identifying risks learned in Rung 6 must also be modified to incorporate the heuristic learned in rung 8.4.

*Graduation Examination*

Once the eStudent has completed the entire curriculum, it is expected to be able to create a complete diagnostic rationale. The rationale identifies explanations for the abnormal observations presented in a scenario in the form of causal chains of hypothesized states or events along with the observations that support and refute the various hypotheses. The rationale, in some cases, will also identify the potentially dangerous states or events that might ensue from the current situation. This capability is limited primarily by the content of the background knowledge: the eStudent's diagnostic rationale will extract a subset of the causal network represented in the background knowledge. The eStudent is expected to have identified the greatest risk associated with each hypothesized state, and to have used that notion of risk to determine the order in which hypotheses are examined to obtain additional evidence and to determine more remote causes.

The eStudent will be presented with an initial state consisting of several alerts. The eStudent is required to identify which of the alerts are abnormal and create alert assertions for them. The eStudent must then create state or event assertions identifying states/events that represent possible explanations for the abnormal alerts, and link them with supports assertions. The eStudent then repeatedly expands the rationale by "processing" the states or events in the rationale. This consists of identifying possible consequent states until risks have been identified, seeking relevant observations, and seeking possible causes. In each case, the eStudent creates assertions in the rationale for the identified states or events and observations, and links them with the appropriate relation assertions.

The eStudent's performance is measured by two criteria: the degree to which the eStudent's rationale matches the rationale expected for the scenario, and the degree to which the order in which the eStudent inserted assertions into the rationale observed the assessment of risk. The overall assessment on the final examination is thus a combination of the assessments associated with each of the individual rungs.

## 4. CONCLUSION

The BL ISS curriculum as described in this paper has not yet been fully tested against the most recent versions of the eStudent. The evaluation process is expected to begin in the near future, as all elements of the program become ready – the representation language shared between the eStudent and the curricula, the formal NIM contracts in which lessons are given, and the further development of the algorithms that make up the eStudent.

While we cannot present performance results that provide evidence for the soundness of the curriculum or the capabilities of the eStudent, there are two other areas of discussion. The first area describes some of the simplifications of the real world that were made in order to contain the scope of this curriculum. The second area outlines future work in this domain being carried out as part of Year 2 of the BL program.

*Simplifications of the Real World*

For this first year of the Bootstrapped Learning project, we have developed a curriculum that focuses on basic flight controller diagnostic skills, situated within the thermal control system of the ISS. These skills include identifying potential problems and diagnosing them through the construction of a problem rationale. Rationale construction involves hypothesizing possible events and states, asserting possible causal explanations, seeking evidence for or against assertions, and projecting possible risks. Explicit rationale construction is designed to obtain visibility in the diagnosis process, but is not a task that a flight controller would actually perform explicitly as part of their job. Controllers do, however, have to undertake rationale construction as a cognitive task during performance of their jobs.

The ISS Simulator provides the eStudent with a simplified user interface to the mission operations software. Specifically, all alerts are presented directly to the eStudent. Additionally, the eStudent can retrieve the value of any telemetry variable by name rather than having to request or navigate to a telemetry screen on which the variable is displayed and find the variable name and value on the screen. Finally, as the curriculum focus for year one is on teaching diagnostic skills, the ISS Simulator makes significant simplifications with respect to modeling the TCS of the ISS.

The ISS Curriculum teaches a simplified version of the diagnostic skills possessed by flight controllers, in that the eStudent:

(1) Is presented with diagnostic scenarios that contain only small and simple sets of interconnected system components that span a few types of components.

(2) Uses knowledge that is represented as fully-instantiated (not axiomatized) rules that relate specific observation types and state types to support the eStudent's causal and evidential inferences among beliefs and observations. This eliminates the need for reasoning from abstract models of the system components. In reality, flight controllers apply a combination of model-based, heuristic, and case-based knowledge about many kinds of devices and physical phenomena to generate possible inferences.

(3) Is asked to compare competing diagnoses, weigh evidence or compare explanations only in a simple, qualitative way. In reality, flight controllers apply (usually qualitative) probabilistic reasoning, combined with their understanding of systems and their causal relationships, to weigh evidence and compare possible explanations.

*Future Work*

In Year 2, we plan to extend the initial ISS curriculum, with the goal of identifying and teaching additional domain specific diagnostic strategies and expanding the coverage of the curriculum:

(1) Risk-based Focus of Attention: At each stage of the general diagnostic algorithm, the learner must make a choice regarding what part of the rationale to expand further. The learner can make use of domain specific heuristics on the risk (including severity, likelihood, and likely time before manifestation) associated with the current conditions.

(2) Likelihood-based Focus of Attention. The learner can also make use of domain specific information on the perceived likelihood of possible explanations gathered from the instructor and use this heuristic knowledge to explore the parts of the rationale most likely to explain the current problem.

(3) Termination. Terminating a search for a diagnosis is important for two reasons. First, when a situation engenders potential dangers it is important to perform recovery and mitigation activities even before the diagnosis is complete. Second, if the expansion of the rationale is focused, then it is possible to terminate the diagnosis before the eStudent exhausts all possibilities.

(4) Expand Coverage: Expand the curriculum to contain additional components of the ISS in order to demonstrate diagnosis over multiple systems.

The first three additions are aimed at further demonstrating how new lessons can be taught to the eStudent to refine its definitions of key concepts – an important goal of an instructable system. The fourth addition will allow for a gradual expansion of scope within the ISS curriculum.

## ACKNOWLEDGEMENTS

This material is based upon work supported by BAE Systems and the United States Air Force Research Laboratory (AFRL) under Contract No. FA8650-07-C-7722. We want to thank Candy Sidner (at BAE Systems) and Ben Rhode (at Cycorp) for their significant input throughout the curriculum development process. The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.

DISTRIBUTION STATEMENT A: Approved for Public Release, Distribution Unlimited

## REFERENCES

[1] N. Shachtman. (2007). DARPA wants software students. [Online]. Available: http://www.wired.com/dangerroom/2007/08/can-you-teach-a/.

[2] DARPA IPTO. [Online]. Available: http://www.darpa.mil/ipto/programs/bl/bl.asp.

[3] D. Oblinger. (2006). Bootstrapped Learning: Creating the electronic student that learns from natural instruction. [Online]. *2009(July 23)*. Available: http://www.darpa.mil/ipto/programs/bl/docs/AAAI_Briefing.pdf.

[4] J. Beal, P. Robertson, and R. Laddaga, "Curricula and metrics to investigate human-like learning," in *Papers from the AAAI 2009 Spring Symposium: Technical Report SS-09-01*, Stanford, CA, 2009.

## BIOGRAPHY

**Jeremy Ludwig** is a research scientist / project manager at Stottler Henke Associates. His research areas include intelligent training systems, behavior modeling, and machine learning, focusing on a number of research projects that utilize both game and simulation technology for training. Jeremy was a co-chair for the 2008 AAAI Fall Symposium on Adaptive Agents in Cultural Contexts and has been involved in several tutorials on the use of artificial intelligence techniques in serious games at I/ITSEC conferences over the past four years. He joined Stottler Henke in the fall of 2000 and holds a PhD in computer science from the University of Oregon.

**John Mohammed** is a research scientist / project manager at Stottler Henke Associates. His research interests include model-based reasoning, case-based reasoning and intelligent training systems. He has worked on and/or led many projects pertaining to reasoning about space-based assets for NASA and for the US Air Force. He joined Stottler Henke in the spring of 1996 and holds a PhD in computer science from Stanford University.

**Jim Ong** is a group manager at Stottler Henke Associates. He leads researchers and software engineers developing artificial intelligence systems and directly manages applied research and product development projects in the areas of intelligent tutoring systems and data visualization.