GLINT: AN INTERPRETER FOR GRAPH MODIFICATION OPERATORS OF THE GERONA LANGUAGE FOR CRISIS DECISION MAKING

BY

GREG R. DHUSE

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2004

Urbana, Illinois

To my parents

ACKNOWLEDGEMENTS

Many thanks to my advisor, Dr. David Wilkins, without whose guidance this project would not have been possible. Thanks to Dr. Lui Sha, whose support allowed me to pursue this research. Thanks also to the graduate students who both taught and collaborated with me on this and other projects – David Fried and Eugene Grois – as well as the entire undergraduate programming staff of the Knowledge-Based Systems Group at the University of Illinois, in particular David King who laid the groundwork for GLINT's abstract object system. Thanks to my family, who have been and continue to be my greatest supporters in all that I undertake. And finally, thanks to the University of Illinois at Urbana-Champaign for providing an abundance of opportunities to learn. We gratefully acknowledge support in part for this research by ONR grants N00014-95-1-0749 and N00014-00-1-0660.

TABLE OF CONTENTS

1. INTRODUCTION

The contribution of this thesis is the Gerona Language Interpreter (GLINT) version 1.0, an interpreter and parser for a major part of the Gerona Language for knowledge representation and inference. Gerona is method of knowledge representation and inference for the procedural skill of crisis decision making. The Gerona explicit knowledge representation facilitates the use of the encoded expert knowledge to support multiple dimensions of expertise, such as problem-solving, critiquing, explanation, question-answering, learning, and tutoring.

GLINT 1.0 parses and interprets Gerona Graph Modification Operators (GMOs), which update a Causal Story Graph (CSG). Parsing GMOs allows GLINT to exhibit expert decision-making and critiquing. The GLINT interpreter will be extended at a future time to support the full Gerona language, specifically Meta Graph Modification Operators (MGMOs). Support for MGMOs will allow GLINT to additionally exhibit question-answering and explanation behavior. The relationship between Gerona, GMOs, MGMOs, and the CSG is illustrated in Figure 1.

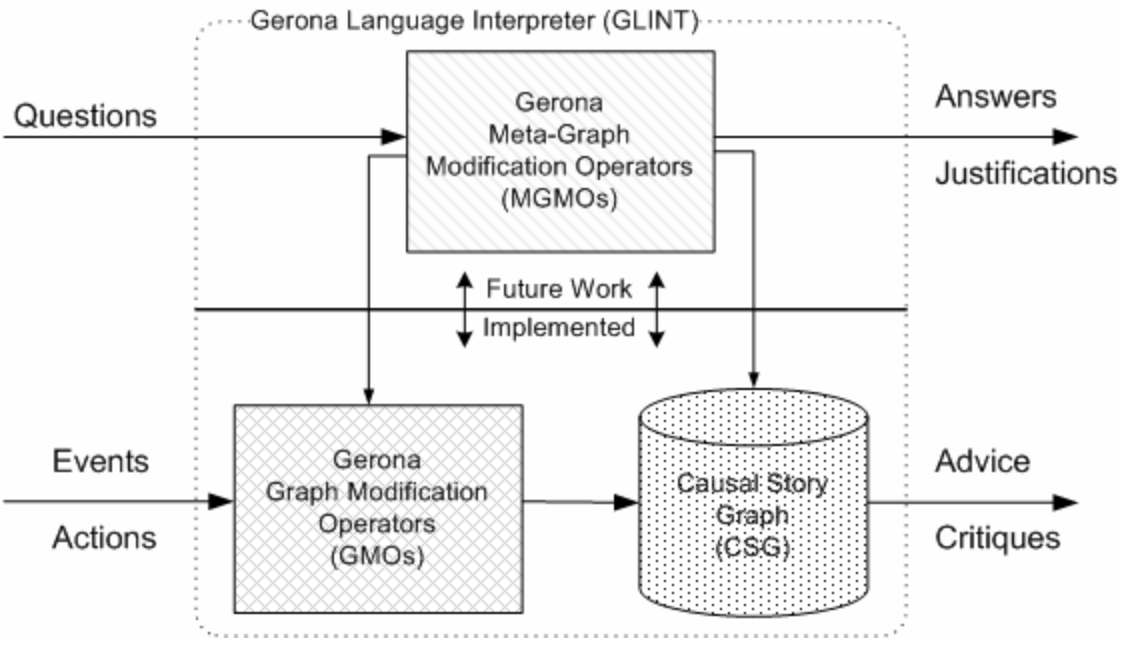

Figure 1 - GLINT Archetecture

Communication between GLINT components occurs via the Event Communication Language (ECL), a formalized language for communication to and from the external environment. Events, Actions, and Questions arrive in the form of ECLs which trigger the execution of GMO or Meta-GMO operations on the CSG. The resultant CSG, which is comprised of a structured set of ECL statements, then holds the entire dynamic state of the system including advice, action critiques, question answers, and justifications.

The rest of this document is organized as follows. Chapter 2 defines the Gerona language for the tasks of problem-solving, critiquing, explanation, question-answering, learning, and tutoring. Chapter 3 describes the syntactic and semantic details of the Gerona language. Chapter 4 describes GLINT 1.0, an implemented system which interprets a major part of the Gerona Language and generates Causal Story Graphs. Chapter 5 examines the performance of GLINT and illustrates the process of Causal Story Graph generation through an example scenario. Chapter 6 discusses future work to build upon the GLINT framework and extend its capabilities as an effective tutor. Finally, Chapter 7 provides a summary of this thesis.

1.1. Related Research

1.1.1. NEOMYCIN

Adapting the MYCIN expert system, NEOMYCIN is an intelligent tutor for the domain of medical diagnosis [3]. NEOMYCIN separates domain-specific expert knowledge from problem-solving strategies by incorporating a distinct strategy layer over a rule model. NEOMYCIN encodes both expert knowledge and teaching heuristics explicitly in its rule set. NEOMYCIN focuses on classification from a static set of symptoms, but does not address dynamic problems such as crisis decision making.

1.1.2. DC-Train and MINERVA-DCA

DC-Train, developed by the Knowledge-Based Systems Group at the University of Illinois, is a simulator and expert system framework for the domain of Navy shipboard

damage control [2, 8, 9, 10]. MINERVA was an early expert system for DC-Train that exhibited problem-solving and decision-making behavior [1, 11].

The MINERVA expert system generates correct solutions to DC-Train crisis scenarios, and critiquing is possible by comparing a student's actions to the ideal decisions as made by MINERVA. While effective for analyzing and scoring student performance, MINERVA is not ideal for more sophisticated critiquing, tutoring, and question-answering. MINERVA does not address the problem of qualifying and quantifying the differences between a student's decision and the correct action, which is itself a difficult reasoning task.

1.1.3. DCX 2.0

DCX 2.0, also developed by the Knowledge-Based Systems Group at the University of Illinois, is a replacement for the MINERVA expert system, and a direct predecessor to GLINT and Gerona. DCX 2.0 pioneered the use of the Causal Story Graph (GSG) and Graph Modification Operators (GMOs) for knowledge representation and inference [4]. As an agent in the DC-Train 4.0 expert system, DCX 2.0 exhibits basic critiquing and problem-solving capabilities.

The DCX 2.0 intelligent agent encodes expert knowledge as GMOs written in C++, making introspection and meta-program analysis difficult. This representation was found to be unwieldy for both researchers and domain experts, and lacking the flexibility and uniform representation of the Gerona language. Most importantly, DCX 2.0 does not provide a suitable environment for the implementation of Meta-GMOs, necessitating the development of Gerona.

2. THE GERONA LANGUAGE

2.1. Motivation for development of the Gerona language

Gerona is a method of knowledge representation and inference that facilitates the user of a knowledge base to support multiple dimensions of expertise, ranging from expert problem-solving, critiquing, explanation, question-answering, knowledge acquisition and learning, and intelligent tutoring. The general approach of all efforts aimed at multi-use knowledge-bases is to make the knowledge modular, uniform, and explicit, and able to be reasoned over by other programs.

Gerona is designed with the following goals in mind:

- The CSG contains the expert model output. It specifies the correct actions to take. The actions are correct even if other experts or students have previously issued erroneous actions.

- The CSG contains the critiquing model output of errors of omission and commission. These errors are shown within the problem-solving context of goals and subgoals, and are available for use by critiquing or tutoring programs

- The expert and critiquing models are encoded in GMOs. These can be read by experts without programming experience. These can also be directly executed by an interpreter, or translated directly into natural language.

- Meta-GMOs operate over GMOs and the CSG. Each Meta-GMO provides answers to questions of a particular class (eg. “how”, “why”, “when”, “what” questions)

Gerona achieves an explicit knowledge representation in a variety of ways. First, the knowledge associated with the procedural skill of crisis decision making is clustered

around actions rather than sub-procedures, which allows expert knowledge to be stored more modularly and in smaller chunks. Second, there is a separation of static and dynamic knowledge. Static domain knowledge is encoded in GMOs which modify the CSG, a structure containing all the dynamic knowledge in the system. Third, GMOs have a very uniform and easy to read syntax. In brief, they consist of if-then rules built from of G-Clause statements, which have a simple and uniform structure.

2.2. Agents and Event Messages

The Event Communication Language (ECL) ties components of the GLINT system together. ECL is a simple formal language for representing atomic information and messages. ECL Messages represent a range of actions, events, questions, and answers. Use of a common representation in all parts of GLINT allows uniform reasoning and inference over the entire space of static and dynamic knowledge.

The ECL language formalizes communication by defining an exact number of ECL message templates for a domain and further grouping these templates into classes according to number.

ECL Number	Message Class
2000-2999	World Representation
3000-3999	World State Predicates
4000-4999	World State Functions
5000-5999	Output from decision maker
6000-6999	Input to decision maker
7000-7999	Goals
8000-8999	Crises
9000-9999	Questions

Table 1 - ECL Message Classes

The ECL representation is designed to simplify translation between a machine-readable form and natural English language. Figure 2 shows an example of translation between the machine-readable and spoken language format of a DC-Train 4.0 message.

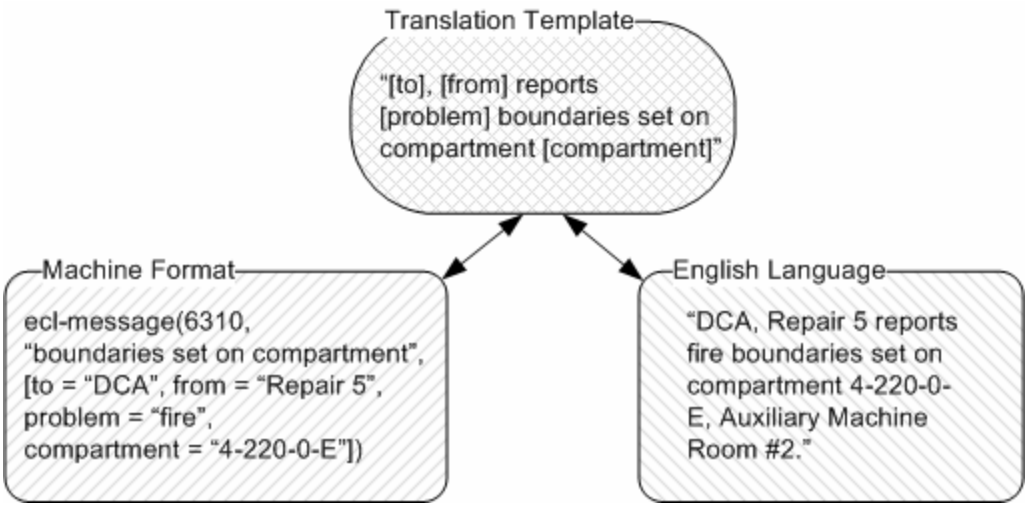

Figure 2 - Event Communication Language (ECL) Message Translation

One of the uses of Gerona and GLINT is to provide the intelligent reasoning component for the DC-Train 4.0 simulator-trainer for shipboard damage control. DC-Train consists of a physical crisis simulator, an intelligent agent simulator, and the Gerona-based expert critiquing system [2, 8, 10, 12]. DC-Train uses agents to simulate discrete entities and individuals in an environment for crisis decision making. In the DC-Train simulator, students assume the role of the DCA while intelligent agents represent the Phone Talkers, CO, EOOW and other entities on the ship. DC-Train 4.0 uses ECL Messages for Agent communication, allowing seamless integration with GLINT. Detailed discussion on the DC-Train 4.0 Event Communication Language can be found in [9], and a complete specification is provided in APPENDIX D – DC-TRAIN 4.0 ECL MESSAGES.

2.3. The Causal Story Graph (CSG)

A Causal Story Graph (CSG) is an organized collection of ECL Messages [4]. It contains the entire dynamic state of the scenario at any point in time, as well as a log of past contexts and a roadmap for the next set of correct actions to take according to domain-

specific doctrine. It may also contain critiques of student actions and answers to domain or scenario-specific questions.

The CSG contains all dynamic knowledge. Execution of GMOs creates a causal structure that encodes an expert model and a critiquing model. Execution of MGMOs over GMOs and the CSG allows question-answering, explanation, and justification. By executing MGMOs or inspection of a CSG, learning or tutoring programs have access to an expert model, critiquing model, and question-answering model.

The CSG shows the causal relation between events, agent reports, crises, and their correct solutions. By searching the CSG, it is possible to determine the cause of an event and, by applying expert domain rules, the correct decisions to address it. The CSG reflects only the events that the student DCA has knowledge of, therefore making it an effective structure for reasoning in the same context as the student.

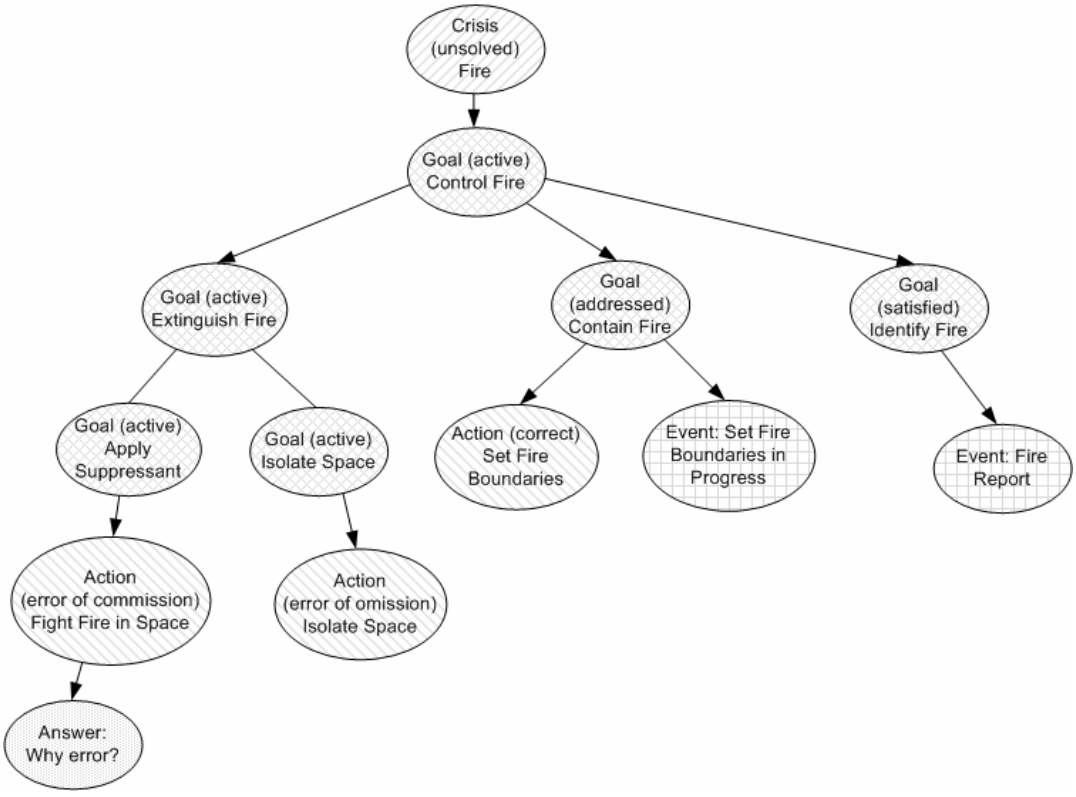

Figure 3 - Example Causal Story Graph (CSG)

A CSG has a tree-like structure, consisting of nodes and the edges that join them. There are 17 types of CSG nodes, organized into 7 categories:

- Scenario
  - ScenarioNode:
    Root of the CSG containing general scenario information.
- Event
  - EventNode:
    Major scenario events that require student action or cause later Crises.
- Crisis
  - CrisisNode:
    An unsolved Crisis that the student must address.
  - SolvedCrisisNode:
    Once a CrisisNode is addressed by the student, it becomes a SolvedCrisisNode.
- Goal
  - GoalNode:
    Domain-dependent goal that the student must achieve. Goals may be hierarchical, in which case the satisfaction of child goals is necessary before a parent goal can be addressed.
  - AddressedGoalNode:
    When a student has carried out the responsibilities necessary to address a goal, the GoalNode is changed to an AddressedGoalNode.
  - SatisfiedGoalNode:
    If the actions taken by a student are confirmed to be correct, a GoalNode or AddressedGoalNode becomes a SatisfiedGoalNode.
- Report
  - ReportNode:
    Information and reports that the student receives from other agents is stored in a ReportNode under the GoalNode it affects.
- Action
  - Unordered
    - PendingActionNode:
      Action that the student is responsible for taking, which has not yet been taken.
    - ExpiredActionNode:
      Action that the student was responsible for, but did not take within the allowed length of time.
    - ErrorOfOmissionNode:
      Action that a student failed to take when necessary.
  - Ordered
    - Non-error
      - CorrectActionNode:

Correct action taken within the optimal timeframe.

- LateActionNode:
  Correct action taken after the optimal timeframe.
- SubOptimalActionNode:
  A sub-optimal action was taken within the optimal timeframe, but the Goal was still satisfied.
- LateSubOptimalActionNode:
  A sub-optimal action was taken after the optimal timeframe, but the Goal was still satisfied.

- Error
  - ErrorOfCommissionNode:
    An incorrect action was taken to address the Goal.

- Answer
  - AnswerNode:
    Stored answer to student question pertaining to the scenario.

2.4. Graph Modification Operators (GMOs)

Graph Modification Operators (GMOs) store all static knowledge for a domain in a uniform representation that is both machine and human readable. Execution of a GMO adds, deletes, or modifies nodes within the CSG. The set of GMOs for a domain defines the entire space of expert domain knowledge as well as the information needed to advise and critique decisions and solve problems.

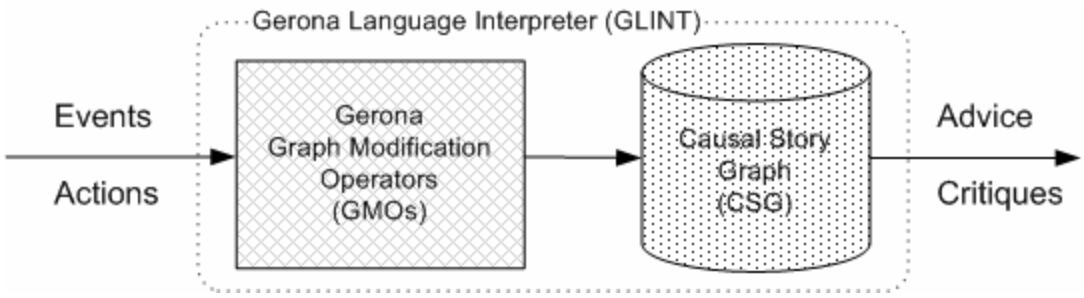

Figure 4 - GMO Execution

GMOs specify discrete operations on a CSG, and are the correct domain-dependent actions to take given a CSG and a new ECL event message. In other words, when a new report or event arrives, a GMO may be executed to determine the action that should be taken in the scenario's current context according to the expert model. A GMO might, for

example, create a new PendingActionNode for the DCA to investigate a fire alarm when ECL 6820 is received from the DCCO. It might also modify an existing PendingActionNode to become a CorrectActionNode once the student DCA correctly issues the ECL 5105 order to investigate. Since a CSG tells the story of a training scenario, running a scenario amounts to building a CSG through the execution of GMOs. Thus, all domain and expert knowledge is encoded in GMOs.

In DCX 2.0 [4], GMOs were implemented as C++ objects within the DC-Train agent framework. This approach had a number of shortcomings that lead to the development of Gerona and the GLINT interpreter. Most importantly, expert knowledge was not stored in an easily machine-readable format, making it impractical to do meta-reasoning about the content of the domain rules without maintaining a second representation. The C++ rule code was also verbose, difficult to debug, and difficult for Navy domain experts to read and write.

The purpose of the Gerona language is to provide a more concise, uniform, and readable way to write GMOs. Through an interpreted framework Gerona facilitates introspection and meta-operations at runtime, making the domain-dependent expert model available in an explicit, machine-readable representation. This allows not only reasoning within the domain, but also about the domain rules themselves. For more information, see Section 2.6 below.

A complete list of Gerona GMOs can be found in APPENDIX E – GERONA GRAPH MODIFICATION OPERATORS.

2.5. Automated Critiquing

Automated expert critiquing is a major use for the current implementation of GLINT. The CSG is the backbone for automated critiquing, which is accomplished by applying expert domain rules to incoming ECL events and the current CSG, in the form of GMOs. When a new ECL event arrives, a GMO is executed to update the CSG with any new information the student may have about the scenario. Expert-coded domain rules are

applied to determine if new Goal conditions exist and whether any existing Goals have been addressed.

2.6. Question Answering Strategies (MGMOs)

The final major design goal of the Gerona language is support for meta-reasoning over expert rules for the purpose of automated question-answering. As described above, GMOs are discrete operators that are triggered by an ECL Message to create or modify nodes in the CSG. GMOs are written in the Gerona language, encode all expert knowledge for the domain, and are executed via the GLINT 1.0 Gerona interpreter. During execution, a GMO can analyze the incoming event, the current state of the CSG, and any static information available about the domain.

Meta-GMOs are similar to GMOs, but with one important difference: Meta-GMOs can also reason about the expert rules encoded in GMOs. Both Meta-GMOs and GMOs are written in the Gerona language. While GMOs execute within the context of the domain, Meta-GMOs are outside observers that examine the GMO parse tree.

Execution of MGMOs over GMOs and the CSG allows Gerona to exhibit question-answering and explanation. Learning and tutoring programs can reason about a scenario by either examining GMOs through the execution of MGMOs, or inspecting the CSG. The CSG provides a critiquing model in conjunction with the GMOs for a particular domain, while MGMOs provide a question-answering model.

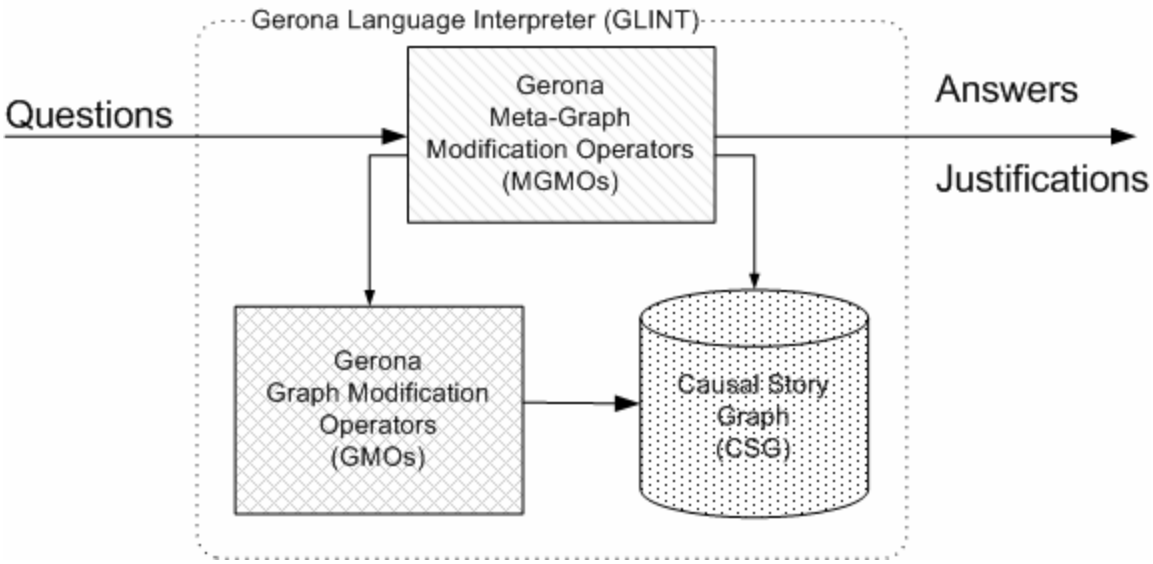

Figure 5 - Meta-GMO Processes

The purpose of a Meta-GMO is to answer student questions about either the general domain doctrine or specific situations in a scenario. For a general scenario-independent question such as: "What is the correct action to take when a report of Fire alarm in compartment 4-174-0-E is received?" this is accomplished by examining the parse tree of the GMO for a relevant ECL Message as well as stored static domain information like ship layout. In this case, the answer would be: "Send a party from Repair Locker 2 to investigate the compartment." To answer a scenario-specific question such as: "What action should I have taken 2:30 into the scenario?" the Meta-GMO must examine domain rules, static ship data, and the CSG at that point in the scenario.

Meta-GMOs are not yet implemented in GLINT 1.0, however the system and the Gerona language were designed with this in mind. Much of the design [5, 6] and framework already exists, and the implementation of Meta-GMOs will most likely be the next major step in this path of research. Details about future work on Meta-GMOs can be found in Section 6.1 below.

3. SYNTAX AND SEMANTICS

The purpose of this chapter is to detail the syntax and semantics of the subset of the Gerona language for knowledge representation and inference implemented in GLINT 1.0. The main constructs of Gerona are GMOs, G-Clauses, Rules, and Subroutines.

The production grammar for each Gerona element implemented by GLINT 1.0 is described in this chapter. For further details, the entire GLINT 1.0 Gerona production grammar is provided in APPENDIX C – GLINT 1.0 GERONA GMO PRODUCTION GRAMMAR.

3.1. Variables

Gerona's variables are weakly typed and any type-casting is done behind the scenes. As a result, domain experts with no previous programming experience can use Gerona without knowledge of variables or types. Since Gerona is case-sensitive, care must be taken not to confuse the case of variables. For example, 'compartment' and 'Compartment' are two distinct variables. Case-sensitive variables are nearly necessary for this application, since many ECL fields are automatically bound in GMO context, drastically reducing the available namespace for intuitive variable names. Examples of possibly conflicting ECL fields from DC-Train 4.0 include: compartment, station, target, source, loop, and casualty. Without case-sensitive variables, users would have to specifically avoid those descriptive words when writing GMOs. A useful convention to distinguish between ECL field variables and user-defined variables is to always begin user-defined variables with a capital letter, while ECL fields are all lowercase. Thus, 'Compartment' is a local, user-defined variable while 'compartment' is the bound value of an ECL field.

Variables do not need to be declared, and are bound at the first time they are used in a binding context. There are three binding contexts for variables:

- In the LET block of a GMO header (see Section 3.4 below)

- In the LET block of a SUBROUTINE (see Section 3.5 below)
- As an unbound argument to a G-Clause (see Section 3.2 below)

It is an error to use an unbound variable outside of a binding context.

3.2. G-Clauses

The main type of statement in a Gerona program is the G-Clause, an imperative function or predicate with syntax similar to Prolog Horn clauses. G-Clauses offer an abstraction to underlying operations like CSG search, and a fixed syntax for all Gerona statements.

G-Clauses provide a highly uniform form for knowledge representation in Gerona, which facilitates reasoning and meta-reasoning. Every Gerona operation, including creating, modifying, and locating a node or other information is written using the same basic G-Clause syntax. G-Clauses also allow Gerona to have a very small amount of specialized syntax that a domain expert is required to learn in order to read and modify Gerona scripts.

Every G-Clause has the same structure and exactly 7 arguments. Arguments are either a single value or, for arguments 2-5, an explicit set of values separated by commas and surrounded by brackets. Arguments may also be an underscore ('_'), which signifies a "Don't care" value, as in Prolog.

The syntax for a G-Clause is detailed below:

gclause: *InfoType(Operation, ECLState, ECLNumber, ECLName, Fields, Parent, Variable)*

InfoType: Type of information or node to find, create, or modify.

Values: goal, action, report, crisis, world-state

Operation: Operation to execute.

Values: find, find-all, modify, create

ECLState: State of Action, Goal or Crisis node to match. Other values for InfoType have no state.

Values:

action: pending, correct, late, sub-optimal, late-sub-optimal, error-of-commission, error-of-omission, expired

goal: unaddressed, addressed, satisfied, overridden

crisis: unsolved, solved, overridden

ECLNumber: ECL Number of node to find, locate, or modify.

Values:

2000-2999 – World Representation

3000-3999 – World State Predicates

4000-4999 – World State Functions

5000-5999 – Output from decision maker

6000-6999 – Input to decision maker

7000-7999 – Goals

8000-8999 – Crises

ECLName: English-language name for ECL, for readability.

Fields: Key = value pairs to match against the fields of any found nodes. The key is always the name of the ECL field to match, and the value may either be a constant, a bound variable, or an unbound variable. If the value is a constant or a bound variable, the key of a found node must match the value to be included. If the value is an unbound variable, it is not included in matching but will be bound to the corresponding value of the returned node or executed predicate. **This is a binding context.**

Parent: For goals, actions, reports and crises, the parent node of that node in the CSG. If an unbound variable is passed, it will be bound to the parent of any found node. **This is a binding context.**

Variable: The returned value, if any, of a statement will be bound to this variable. Returned values may be a node, a set of nodes, or a literal value. **This is a binding context.**

Following is a list of allowed single-value and set relations within G-Clause field lists:

Relation	**Operator Semantics**		
	Number	**String**	**Set**
=	Number A is equal to Number B	String A is identical to String B	N / A
<	Number A is strictly less than Number B	N / A	N / A
>	Number A is strictly greater than Number B	N / A	N / A
<=	Number A is less than or equal to Number B	N / A	N / A
>=	Number A is greater than or equal to Number B	N / A	N / A
<>	Number A is not equal to Number B	String A is not identical to String B	N / A
>~	N / A	N / A	Set B contains Value A
<~	N / A	N / A	Set A contains Value B
!~	N / A	N / A	Set B does not contain Value A

Table 2 - G-Clause Field List Operator Semantics

Set operations are transparent to the user in most cases. If a variable is bound to a set and later passed as an argument to a G-Clause, each member of the set will be evaluated in that clause. Thus, it is not necessary for the programmer to consider whether multiple nodes might match search criteria, or to treat a set of values any differently than a single value. One exception is in the G-Clause field list, where generally a subset relation is desired rather than strict set equality. Since ECL field parameters only have a single value, the field comparison [compartment = CompartmentSet] does not make sense. Instead, a set relation should be used instead of '=' in this case.

3.3. Control Structures

Gerona has only two simple control structures. Since set operations are implicit within the language, iterative control structures found in many other languages are not necessary. WHILE, FOR, FOREACH, and DO…WHILE are examples of structures that do not exist in Gerona.

A complete list of Gerona's lexical tokens and reserved words can be found in APPENDIX B – GLINT 1.0 GERONA GMO TOKENS AND RESERVED WORDS.

3.3.1. Either…Or

EITHER…OR blocks are the first and simplest control structure in the Gerona Language. They are no more than a verbose English-language logical OR of predicates. If either the first or second blocks evaluates to TRUE, the EITHER…OR statement also evaluates to TRUE. The syntax is formalized below:

either_or: *EITHER logical_block OR logical_block END EITHER*

3.3.2. Logical and Procedural Blocks

A code block is a list of statements that are evaluated together. Two types of code blocks exist in Gerona: Logical and Procedural. A logical block always evaluates to TRUE or FALSE and consists of a series of G-Clause predicates or EITHER…OR blocks joined with AND. A predicate can be negated using the NOT reserved word. The syntax is shown below:

logical_block: [logical_block AND] [NOT] gclause
| *[logical_block AND] [NOT] either_or*

A procedural block is a list of G-Clauses, Rules, or special operations such as CALL, which are executed serially. If the procedural block is executed, all statements are guaranteed to be run in order.

procedural_block:
[procedural_block] gclause
| *[procedural_block] call*
| *[procedural_block] rule*

3.3.3. Rules

Most expert domains can be subdivided into a list of rules to apply in particular situations. In Gerona, the space of domain knowledge is divided in two ways:

- By event; an ECL Message going to or from the DCA decision maker
- By rules within an event that can be executed when certain criteria match

Rules are a special kind of control structure that permits conditional execution in the Gerona language. Syntactically, a Rule contains an identifier and an IF...THEN block. Rule identifiers are dotted strings with the following syntax:

identifier: *number[.string][.string][.number][.number] [“string”]*
| *string[.string][.number][.number] [“string”]*

For example: 5150.valve.critique.1 “Open/Close firemain valve is correct”.

Rules are sub-parts of a GMO that contain a logical block and a procedural block. The logical block is the IF part of the statement. If the logical block evaluates to TRUE, the Rule’s procedural block is executed. The procedural block is the THEN part of the statement. Rule syntax is formalized below:

rule: *RULE identifier IF logical_block THEN procedural_block END RULE*

Rules carry a local scope, and variables that are bound inside either the logical or procedural block of a Rule leave scope and disappear when the Rule has finished executing.

The following is an example Rule with a nested Rule within its procedural block:

```
RULE 5150.valve.critique.1 "Open/Close firemain valve is correct"
IF
    world-state(find, _, 4105, "Valve to Firemain Loop", [valve =
       Valve, loop = Loop], _, _)
    AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
       [loop = Loop], _, G)
    AND action(find, pending, 5150, "Open/Close Firemain Valve",
       [valve = Valve, valve_action = ValveAction], _, A)
THEN
    action(modify, correct, 5150, "Open/Close Firemain Valve",
       [valve = Valve, valve action = Valve Action], _,A)

    RULE 5150.valve.critique.1.1 "action satisfies restore firemain
       pressure goal"
    IF
      NOT world-state(find, _, 3901, "Goal Has Pending Actions",
       [goal = G], _, _)
    THEN
      goal(modify, addressed, 7560, "Restore Firemain Pressure",
       [loop = Loop], _, G)
    END RULE
END RULE
```

The outside Rule, 5150.valve.critique.1, has 3 predicates inside its logical block and 2 statements inside its procedural block. The first procedural statement is an action G-Clause and the second is a nested Rule, 5150.valve.critique.1.1. The procedural block will be executed if and only if world-state() AND goal() AND action() evaluate to TRUE.

If the procedural block is executed, both the action() statement and the Rule are guaranteed to be executed. The THEN portion of the nested Rule will only execute if its IF portion evaluates to TRUE.

3.4. GMO Syntax

Graph Modification Operators (GMOs) store all static expert knowledge for a domain. The syntax for GMOs in Gerona reflects the event-driven nature of DC-Train 4.0's agent framework. Each GMO is associated with a type of ECL message that triggers its evaluation. A GMO is fundamentally a list of Rules that may be applied to the CSG when the corresponding event occurs, and other expert-specified scenario criteria exist. The GMO syntax appears below:

gmo: *GMO FOR ECL identifier [WHERE condition_list] [LET assighment_list] rule_list END GMO*

condition_list: [condition_list AND] key relation value

assignment_list:

[assignment_list,] value -> Variable

| *[assignment_list,] Field -> Variable*

| *[assignment_list,] field set -> Variable*

field_set: *[field_set,] Field*

rule_list: *[fule_list] rule*

GMOs consist of a header and a body, which is simply a block of Rules. The GMO header always contains a FOR ECL statement, which associates the GMO with an ECL event number. Following, the header may contain an optional WHILE block of conditions under which the GMO should be executed. Conditions are 'key relation value' statements, and a list of possible relations can be found in Table 2. Each time an event of this type arrives and all WHILE conditions are met, the GMO will be executed.

The final piece of a GMO header is the LET assignment block. The LET block allows fields from the triggering ECL message or literal values to be bound to local variables. A

set of values may be bound to a single variable by enclosing the values in brackets and separating each with a comma.

When a GMO is executed, each Rule is executed in order. An excerpt from GMO 5101 is shown below to illustrate GMO syntax:

```
GMO 5101
FOR ECL 5101 "Flood Magazines"
LET compartment -> Compartment

   RULE 5101.flood-magazine.critique.1 "flood magazines order is
        correct"
      ...
   END RULE

   RULE 5101.flood-magazine.critique.2 "flood magazines incorrect -
        no permission"
   ...
   END RULE
   ...
END GMO
```

3.5. Subroutines

Subroutines are convenience structures within Gerona that facilitate code reuse. Subroutines are defined with the SUBROUTINE reserved word, and executed within a procedural block with a CALL statement. The syntax for a subroutine is formalized below:

subroutine: *SUBROUTINE name [LET assignment list] rule list END SUBROUTINE*

Examples of Gerona subroutines can be found in Appendix E.1. Subroutines.

The syntax for a subroutine is very similar to the syntax for a GMO. The primary difference is that subroutines are called with an argument list, whereas GMOs are triggered by an ECL event. While a GMO's LET block binds ECL fields to local variables, a subroutine's LET block binds arguments from its argument list to local variables.

Within a GMO, subroutines are invoked with the CALL statement. CALL takes an arbitrary number of key=value bindings that are made available to bind against in the subroutine's LET block. When a CALL statement is executed, a new local variable context is created, which lasts until the subroutine completes execution.

call:	*CALL subroutine-name(binding_list)*
binding_lsit:	*[binding_list,] key = value*
|	*[binding_list,] key = Variable*

An example CALL statement from GMO 5156.start is shown below:

```
CALL get-new-firemain-orders(gmo-name = "5520.request-pump",
   goal = G, loop = Loop)
```

4. INTERPRETING GERONA

The GLINT 1.0 interpreter is responsible for parsing a subset of the Gerona language, creating an executable parse tree, and executing Gerona GMOs and Rules in response to scenario events in the form of ECL Messages. GLINT can be divided into 4 logical components:

Lexical tokenizer:	Subdivides raw Gerona code into semantically useful chunks.
Gerona parser:	Constructs a Gerona parse tree from a stream of lexical tokens.
Abstract object system:	Constructs an abstract hierarchy of C++ objects that represent semantic concepts in the Gerona parse tree.
Evaluation system:	Executes abstract Gerona statements at run-time.

Each piece of the interpreter is covered in detail, and will be illustrated using the excerpt of code from GMO 5105 shown with numbered lines below:

```
1  GMO 5105
2  FOR ECL 5105 "Investigate Compartment"
3        LET compartment -> Compartment,
4         target -> Station

5        RULE 5105.investigate.critique.1 "investigate order is
                correct"
6        IF
            goal(find, unaddressed, [7105, 7205, 7305],
                  ["Identify Fire", "Identify Flooding",
                  "Identify Smoke"], [compartment = Compartment],
                  _, G)
7           AND action(find, pending, 5105, "Investigate
                  Compartment", [compartment = Compartment], _,
                  A)
8           AND world-state(find, _, 4302, "Best Repair Locker
                  for Compartment", [compartment = Compartment,
                  station = Station], _, _)
9        THEN
10          goal(modify, addressed, [7105, 7205, 7305],
                  ["Identify Fire", "Identify Flooding",
                  "Identify Smoke"], [compartment = Compartment],
                  _, G)
```

```
11          action(modify, correct, 5105, "Investigate
                  Compartment", [compartment = Compartment,
                  target = Station], _, A)
12       END RULE
         ...
13 END GMO
```

Figure 6 details the architecture of the GLINT interpreter and parser. During the Parse Phase, the Lexical tokenizer, Gerona parser, and Abstract Object System analyze Gerona GMOs and Meta-GMOs, building a parse tree. During execution, an incoming ECL message triggers the Evaluation system which executes Gerona in the current context to produce a CSG.

4.1. The Lexical Tokenizer

Gerona's lexical tokenizer, or lexer, is arguably the simplest part of the Gerona interpreter. Its job is to read raw Gerona code from a file, memory, or other input and subdivide the text into meaningful chunks for parsing. As the input is read, text in the stream is matched against regular expressions to determine its meaning in the Gerona language. The lexer was implemented with GNU Flex [7], a common open-source tool for building lexical analyzers. An example helps to illustrate the job of the lexical tokenizer:

```
1  GMO 5105
2  FOR ECL 5105 "Investigate Compartment"
3        LET compartment -> Compartment,
4         target -> Station
```

When passed through the lexer, the above raw Gerona text would tokenize to:

```
1  START_GMO INT_NUM
2  FOR_ECL INT_NUM STRING_LITERAL
3  LET STRING ARROW STRING ‘,’
4         STRING ARROW STRING
```

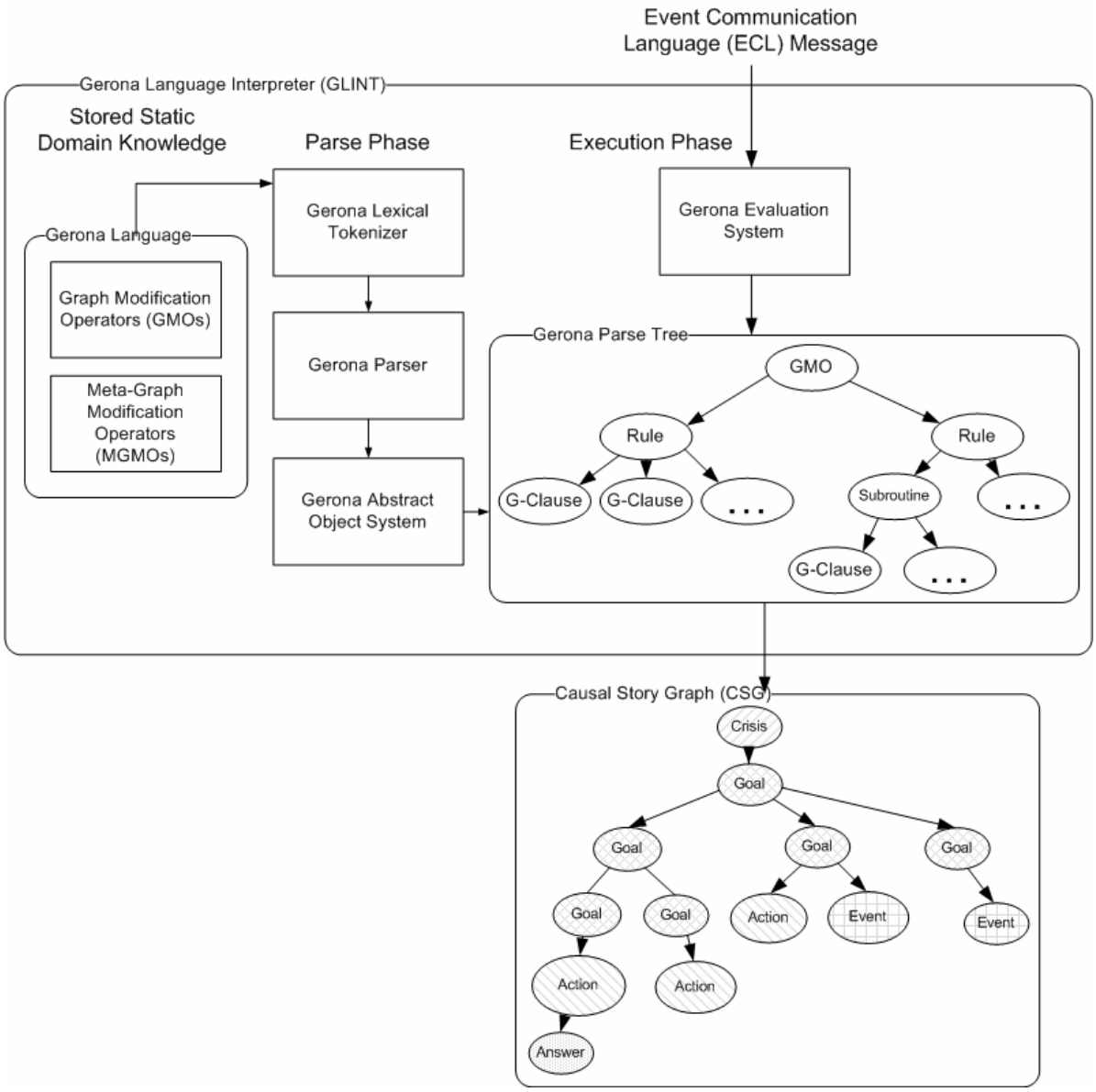

Figure 6 - GLINT 1.0 Architecture

Whitespace and line breaks are ignored in Gerona, but they have been preserved in the above block for readability. Unlike the raw GMO text, these tokens begin to have meaning as elements of the Gerona language. Notice that keyphrases such as “FOR ECL” are grouped and translated into a single meaningful language token, FOR_ECL. “FOR ECL” is a reserved phrase, and a complete list of Gerona’s tokens is available in APPENDIX B – GLINT 1.0 GERONA GMO TOKENS AND RESERVED WORDS.

4.2. The Gerona Parser

The next step in interpreting Gerona is the parser. The Gerona parser takes tokens from the lexical tokenizer and assembles them into semantic components according to the language grammar. Raw tokens are reduced to meaningful symbols, which are often further reduced into other symbols, until the whole program comprises one root-level symbol. In Gerona, the root-level symbols are *gmo* and *subroutine*, which are described in Sections 3.4 and 3.5 above.

The Gerona parser is a Shift/Reduce parser, a common type of parser for programming languages, and is implemented using GNU Bison. Bison is an open-source parser compiler based on the ubiquitous tool Yacc [7]. As an overview, the premise of Shift / Reduce parsing is that each new token can either be pushed onto a stack of symbols already in the program, or combined with other symbols on the stack to reduce into a new symbol. This process is best shown with an extension of our previous example. The tokens below would be parsed in the following abbreviated steps, with symbols shown in *italics*:

```
1  START_GMO INT_NUM
2  FOR_ECL INT_NUM STRING_LITERAL
3  LET STRING ARROW STRING ‘,’
4         STRING ARROW STRING
```

Step 1: Shift(2) + Reduce

```
1  START_GMO identifier
```

Step 2: Reduce + Shift(3) + Reduce

```
1  gmo_start
2  FOR_ECL identifier
```

Step 3: Shift(3)

```
1  gmo_start
2  FOR_ECL identifier
3  LET STRING ARROW STRING
```

Step 4: Reduce + Shift(4) + Reduce

```
1  gmo_start
2  FOR_ECL identifier
3  LET assignment ‘,’
4         assignment
```

Step 5: Reduce

```
1  gmo_start
2  FOR_ECL identifier
3  LET assignment_list
```

Step 6: Reduce

```
1  gmo_header
```

Often, languages are specified as a context-free grammar, which defines the production rules that allow reductions from one or more symbols to another. Some of Gerona's production rules are formalized in Section 3, however a complete specification of the grammar used in the GLINT 1.0 parser is provided in APPENDIX C – GLINT 1.0 GERONA GMO PRODUCTION GRAMMAR.

As the parser makes reductions, it also creates abstract language objects in C++ and adds them to the parse tree. The term parse tree generally refers to a tree representation of the symbols in a piece of code, and the reductions made in the parsing process. This representation is interesting to language researchers, but is not of practical relevance to this thesis on knowledge representation and AI. Therefore, when the Gerona parse tree is discussed in this thesis, it refers to the collection of abstract objects that represent a Gerona program, which are arranged in a tree-like hierarchy.

4.3. Abstract Object System

Gerona's abstract object system is a collection of C++ classes that implement underlying language functionality. Each language feature must be represented in a C++ object, which will be executed when Gerona programs are run. The object system is where all behind-the-scenes language operations take place. In the interest of making Gerona a simple, concise, and powerful language for non-technical domain experts, much functionality that is normally left to the user in a language like C/C++ has been moved inside Gerona to a lower level. This simplifies the language itself, but mandates a large C++ support back-end.

Gerona's object system is abstract because it makes extensive use of C++ polymorphism. This allows representation of language features in a logic-centered way, and facilitates code reuse by removing most special cases. An EITHER…OR object, for example, can be treated like any other logical statement, with no distinction necessary between it and a predicate G-Clause in a logical block.

The definition for the C++ object implementation of Gerona's EITHER…OR block is shown below for illustration:

```
class EitherStatement : public LogicalStatement
{
   public:

      EitherStatement( LogicalBlock* either1Block,
                       LogicalBlock* either2Block ) :
        either1Block_(either1Block), either2Block_(either2Block) { }

      virtual ~EitherStatement();

      bool execute();

   private:

      LogicalBlock* either1Block_;
      LogicalBlock* either2Block_;
};
```

The class EitherStatement inherits from LogicalStatement, which also includes predicate G-Clauses. An EITHER…OR statement approximates a logical OR in Gerona by first evaluating the first block, and if the result is FALSE, evaluating the second block. The statement as a whole evaluates to TRUE if and only if one or more of its two blocks evaluates to TRUE. The implementation of this from the EitherStatement object is shown below:

```
bool
EitherStatement::execute()
{
   if ( either1Block_->execute() == true )
      return true;
   else
   {
      if ( either2Block_->execute() == true )
        return true;
      else
```

```
            return false;
        }
    }
```

The Gerona parse tree is a Program object which contains a list of all parsed GMO and Subroutine objects. Note that these correspond to the root parser symbols of *gmo* and *subroutine*. Many, but not all, abstract objects mirror counterpart symbols in the parser. This is because fundamental language structures, like EITHER...OR statements, both constitute a language symbol and a piece of language logic that needs to be implemented in the underlying framework. It is also convenient to create a concrete object instance each time the parser makes a reduction to the corresponding symbol.

Figure 7 is an excerpt from the Gerona parse tree that corresponds to our GMO 5105 example. Each node in the figure represents an abstract object in the GLINT interpreter. The arrows show that more abstract objects like GMO are built from the lower-level objects Rule and LogicalBlock.

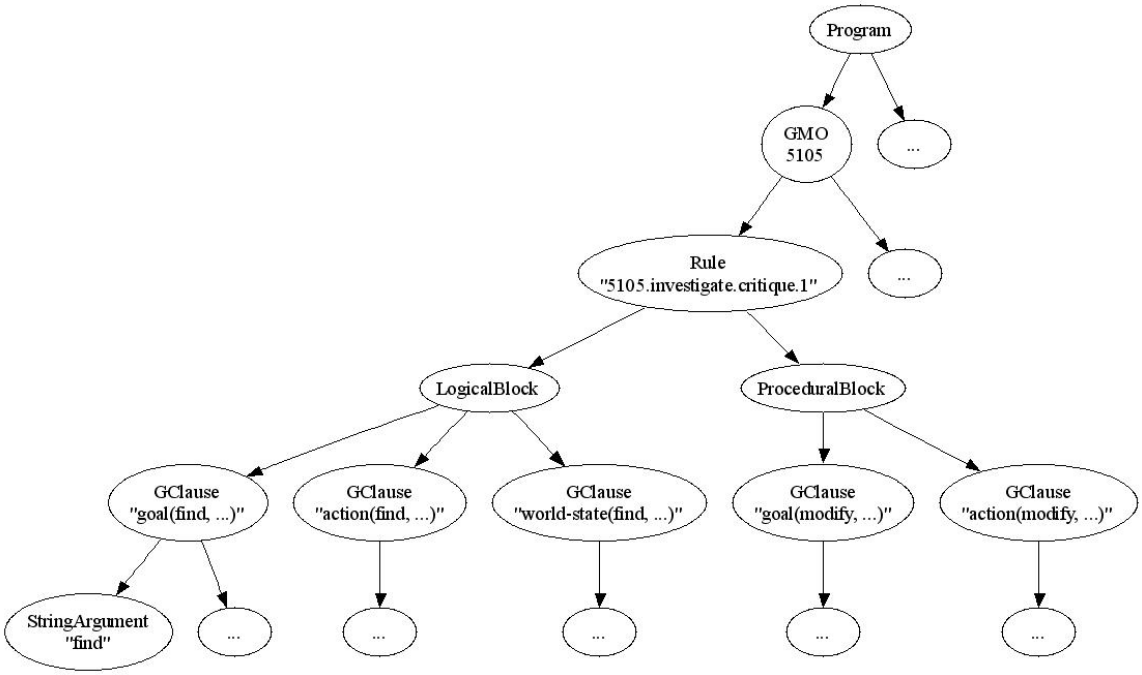

Figure 7 - Partial Example Parse Tree

4.4. Evaluation System

The final major component of the Gerona interpreter is the evaluation system. This system is responsible for run-time evaluation of Gerona language statements. Each object in the Gerona parse tree is conditionally evaluated by calling its execute() method in the correct context. Flow of execution is a pre-order traversal of the parse tree, ending at leaf objects.

The pseudo-code below shows a rough outline of the process of execution when a new ECL Message arrives:

```
Program.execute(ECL 5101)
{
   foreach (GMO)
      if ( (GMO.ecl == 5101) AND (GMO.while_block.matches() )
         GMO.execute()
         {
            // Do LET bindings

            foreach (Rule)
               Rule.execute()
               {
                  if (LogicalBlock.execute())
                     ProceduralBlock.execute()
               }
         }
}
```

When an ECL Message 5101 arrives, the Program attempts to find one or more GMOs that match the ECL and its parameter Fields. If a match is found, that GMO is executed. When a GMO is executed, the assignments in its LET assignment list are bound in a local scope. Next, each Rule is executed by first evaluating its LogicalBlock and, if successful, its ProceduralBlock. This flow of execution continues to G-Clauses and other statements within those blocks, and even to individual variables and arguments.

Each set of braces in the above section of pseudo-code indicates a local variable scope. That means that any bindings that occur within that scope do not back-propagate when the scope is exited. Programs, GMOs, and Rules all have local scopes.

4.5. CSG and World-State Operations

As discussed throughout this thesis, the execution of Gerona and individual GMOs performs modifications to the Causal Story Graph. Therefore, the Gerona interpreter must support language-level operations for CSG node manipulation. This is done through G-Clause operations, which are implemented in C++.

Each call to a G-Clause executes the corresponding low-level operation which is abstracted in user-space at a conceptual level. For example, calling goal(create…) conceptually creates a new GoalNode in the CSG and fills its parameters. In actuality, the creation of this node may trigger creation of an entire sub-tree of Goals. These details are taken care of behind-the-scenes to reduce load on the user.

Another type of provided functionality is world-state operations. These predicates and functions correspond to ECLs 2000-4999 and provide information about static areas of the domain, such as Repair Locker jurisdiction and the ship's layout.

It may be possible to implement many CSG and world-state operations directly as subroutines in the Gerona language. Since it could be argued that certain static aspects of domain knowledge are implicit in these operations, implementing them in Gerona would extend the static domain knowledge, and permit Meta-GMO reasoning over those parts of the domain. Since this knowledge does not change or bear direct relevance to training in our domain, this has not been a high priority, however it is a task that may be undertaken at a later time.

5. RESULTS

This chapter will illustrate the execution of the GLINT 1.0 interpreter in an example crisis decision making scenario from the domain of Navy shipboard damage control. The graphs shown are actual Causal Story Graphs (CSGs) generated by GLINT in response to world events, human decisions, and the actions of agents.

In the domain of shipboard damage control, the Damage Control Assistant (DCA) is a naval officer responsible for addressing crises onboard a ship. This scenario highlights GLINT's critiquing capabilities by suggesting and analyzing the actions of a human DCA. Communication between the human DCA decision maker and the simulation occurs via orders issued according to Navy doctrine, and encoded as ECL Messages in the range 5000-5999. Similarly, messages from agents to the DCA are encoded as ECL Messages in the range 6000-6999.

Complete source code for Gerona GMOs appears in APPENDIX E – GERONA GRAPH MODIFICATION OPERATORS. It may be helpful to refer to the relevant GMOs throughout the example scenario.

5.1. Example Scenario

This scenario was chosen to be a typical crisis decision making situation that Navy DCAs are trained to handle. There are 6 key events in the scenario:

1. **No ECL** Scenario begins
2. **ECL 6820** DCCO reports a fire alarm in compartment 4-174-0-E to the DCA
3. **ECL 6801** Report from crew in compartment 4-174-0-E confirms the fire
4. **ECL 5170** DCA orders the electrical and mechanical isolation of compartment 4-174-0-E, but sends the wrong repair party
5. **ECL 5114** DCA correctly orders fire boundaries on compartment 4-174-0-E
6. **ECL 5120** DCA incorrectly orders crew to move into compartment 4-174-0-E to fight the fire before ordering desmoking

5.2. GLINT Results

Event 1: Scenario begins

At the beginning of the scenario, the Causal Story Graph (CSG) contains only one node, the Scenario Node. This node contains general information about the scenario, such as its title, and is the root of the CSG.

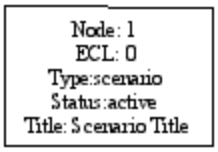

Figure 8 - CSG After Event 1

Event 2: DCCO reports a fire alarm in compartment 4-174-0-E to the DCA

GLINT receives an ECL 6820 "Alarm in compartment from DCCO" message, indicating that the Damage Control Console Operator identified an alarm in compartment 4-174-0-E. This message triggers the execution of GMO 6820.fire.

Each rule in GMO 6820.fire is evaluated, and rule 6820.fire-alarm.interpret.3 "Fire alarm triggers new crisis" is applied to the CSG

```
RULE 6820.fire-alarm.interpret.3 "Fire alarm triggers new crisis"
IF
    NOT crisis(find, unsolved, 8100, "Fire", [compartment =
        Compartment], _, _)
THEN
    crisis(create, unsolved, 8100, "Fire", [compartment =
        Compartment], scenario-node, C)
    goal(create, unaddressed, 7100, "Control Fire", [compartment =
        Compartment], C, _)
    …
END RULE
```

First, a new unsolved CrisisNode is added to the CSG for this possible fire (Node 2). Then, GoalNode "Control Fire" (Node 3) is created as a child of the CrisisNode.

Similarly, the sub-goals “Extinguish Fire” (Node 4), “Identify Fire” (Node 5), and “Contain Fire” (Node 6) are added to the CSG. In turn, child sub-goals, “Apply Fire Suppressant” (Node 7) and “Isolate Compartment if Necessary” (Node 8) are included.

The rule 6820.fire-alarm.interpret.4 “fire alarm report attached to identify fire goal” also matches, and is evaluated. This small rule simply creates a report to represent the ECL 6820 alarm received (Node 9). Figure 9 shows the CSG after GMO 6820.fire executes.

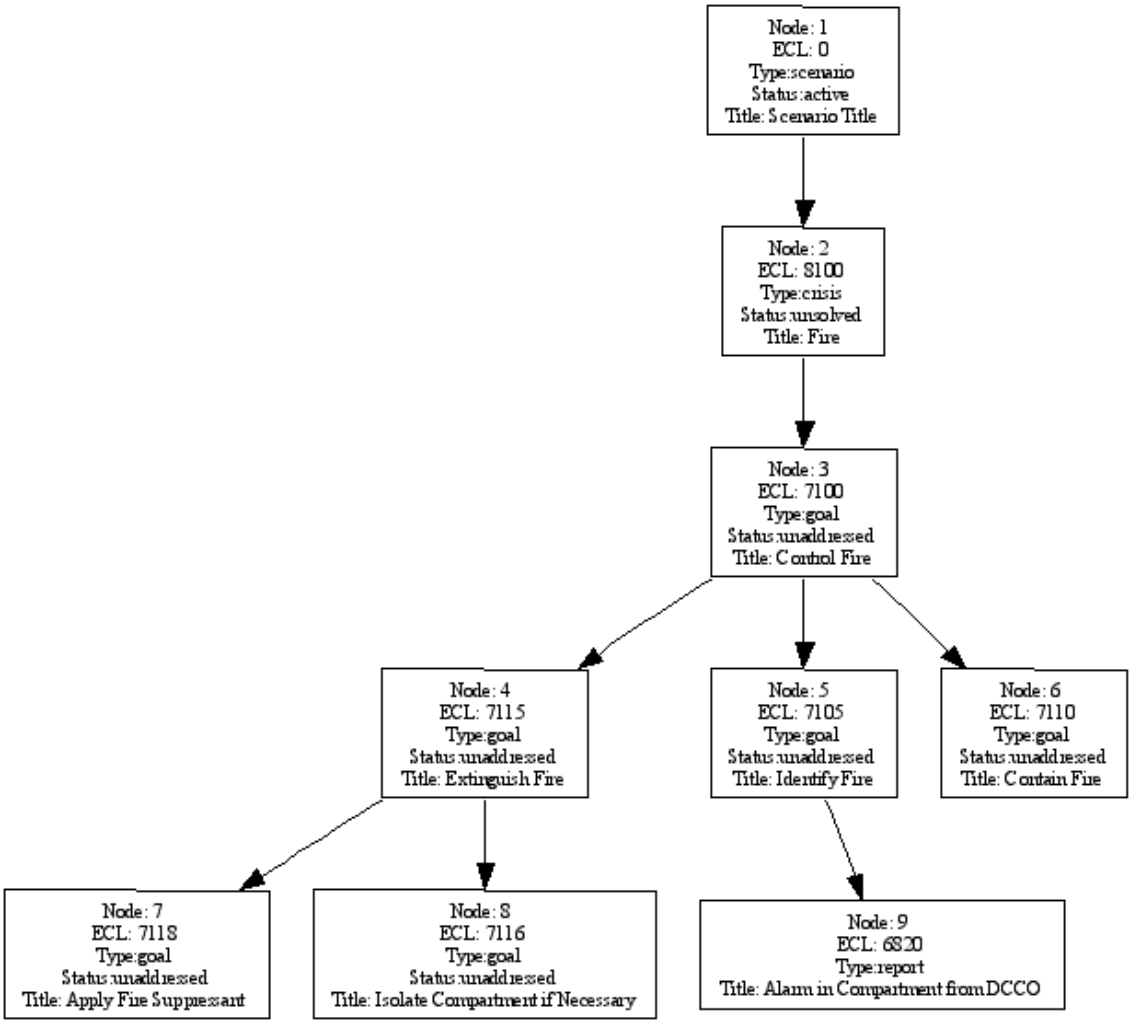

Figure 9 - CSG After Event 2

Event 3: Report from crew in compartment 4-174-0-E confirms the fire

At this point in the scenario, an ECL 6801 “Damage report: fire” message arrives from compartment 4-174-0-E, triggering evaluation of GMO 6801.fire. The IF blocks of two rules and two sub-rules evaluate to TRUE, prompting the execution of those rules.

First the simple rule 6801.fire-report.interpret.4 “associate new fire report with Identify Fire goal” is executed, placing a ReportNode for the new damage report (Node 10) into the CSG.

Next, rule 6801.fire-report.interpret.5 "fire report satisfies identify fire goal" modifies the GoalNode “Identify Fire”, changing its status from “unaddressed” to “satisfied”. Two sub-rules of 6801.fire-report.interpret.5 also match and are executed. Rule 6801.fire-report.suggest.1 "if isolation has not already been achieved, propose it if it is possible" determines that the DCA decision maker must electrically and mechanically isolate compartment 4-174-0-E before firefighting can take place, and places a PendingAction node (Node 11) into the CSG. Finally, rule 6801.fire-report.suggest.3 "set fire boundaries on compartment if necessary" executes and adds a PendingAction node “Set Boundaries on Compartment” (Node 12) as a child of the “Contain Fire” goal. This explicitly states that the DCA is responsible for containing the fire with fire boundaries before it can be effectively fought.

Event 4: DCA orders the electrical and mechanical isolation of compartment 4-174-0-E, but sends the wrong repair party

At this time, the human DCA decision maker orders isolation of compartment 4-174-0-E by sending a party from Repair Locker 2 via ECL 5170. According to Navy doctrine, Repair Locker 5 has jurisdiction over this compartment, so the DCA has just performed a sub-optimal action. GLINT evaluates GMO 5170 in response to this ECL message, and rule 5170.isolate.critique.2 "wrong repair locker for isolate space - sub-optimal action" is run.

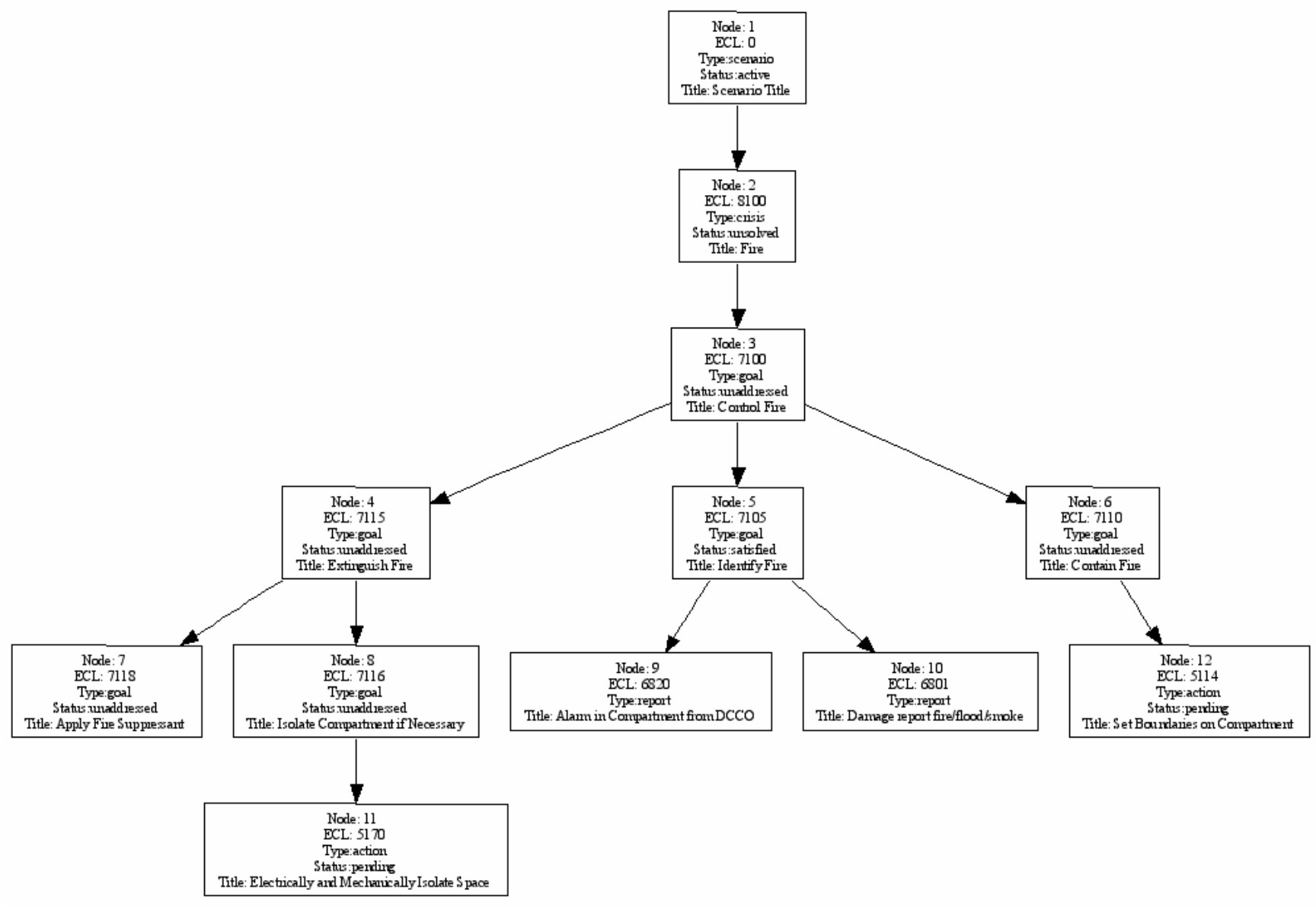

Figure 10 - CSG After Event 3

```
RULE 5170.isolate.critique.2 "wrong repair locker for isolate space
- sub-optimal action"
IF
   goal(find, unaddressed, 7116, "Isolate Compartment if Necessary",
        [compartment = Compartment], _, G)
   AND action(find, pending, 5170, "Electrically and Mechanically
        Isolate Space",
        [compartment = Compartment], P, A)
   AND world-state(find, _, 3360, "Can Isolate Compartment",
        [compartment = Compartment], _, _)
   AND NOT world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
        [compartment = Compartment, station = Station], _, _)
   AND world-state(find, _, 3401, "Has Personnel", [station =
        Station], _, _)
THEN
   goal(modify, addressed, 7116, "Isolate Compartment if Necessary",
        [compartment = Compartment], _, G)
   action(modify, expired, 5170, "Electrically and Mechanically
        Isolate Space",
        [compartment = Compartment], _, A)
   action(create, sub-optimal, 5170, "Electrically and Mechanically
        Isolate Space",
        [target = Station, compartment = Compartment], P, _)
END RULE
```

First the goal “Isolate Compartment if Necessary” is marked “addressed”, and then the original PendingAction “Electrically and Mechanically Isolate Space” is changed to an ExpiredAction. Finally, the rule creates a new SubOptimalActionNode (Node 13) to represent the sub-optimal isolation order from the DCA.

GMO 5170 provides a critique of the decision maker’s action by identifying a sub-optimal decision and storing this critique within the CSG. The CSG stores the entire dynamic state of the scenario. Not only is each event in the simulation recorded, but the execution of GMOs adds expert critiques and suggested optimal decisions to the CSG in an explicit form.

Event 5: DCA correctly orders fire boundaries on compartment 4-174-0-E

The next event is an order ECL 5114 from the DCA to set fire boundaries on the compartment. This order correctly addresses the “Contain Fire” goal. When the ECL message is received, GLINT evaluates GMO 5114.fire. Rule 5114.set-fire-boundaries.critique.1 "correct fire boundaries" is then executed, which marks the

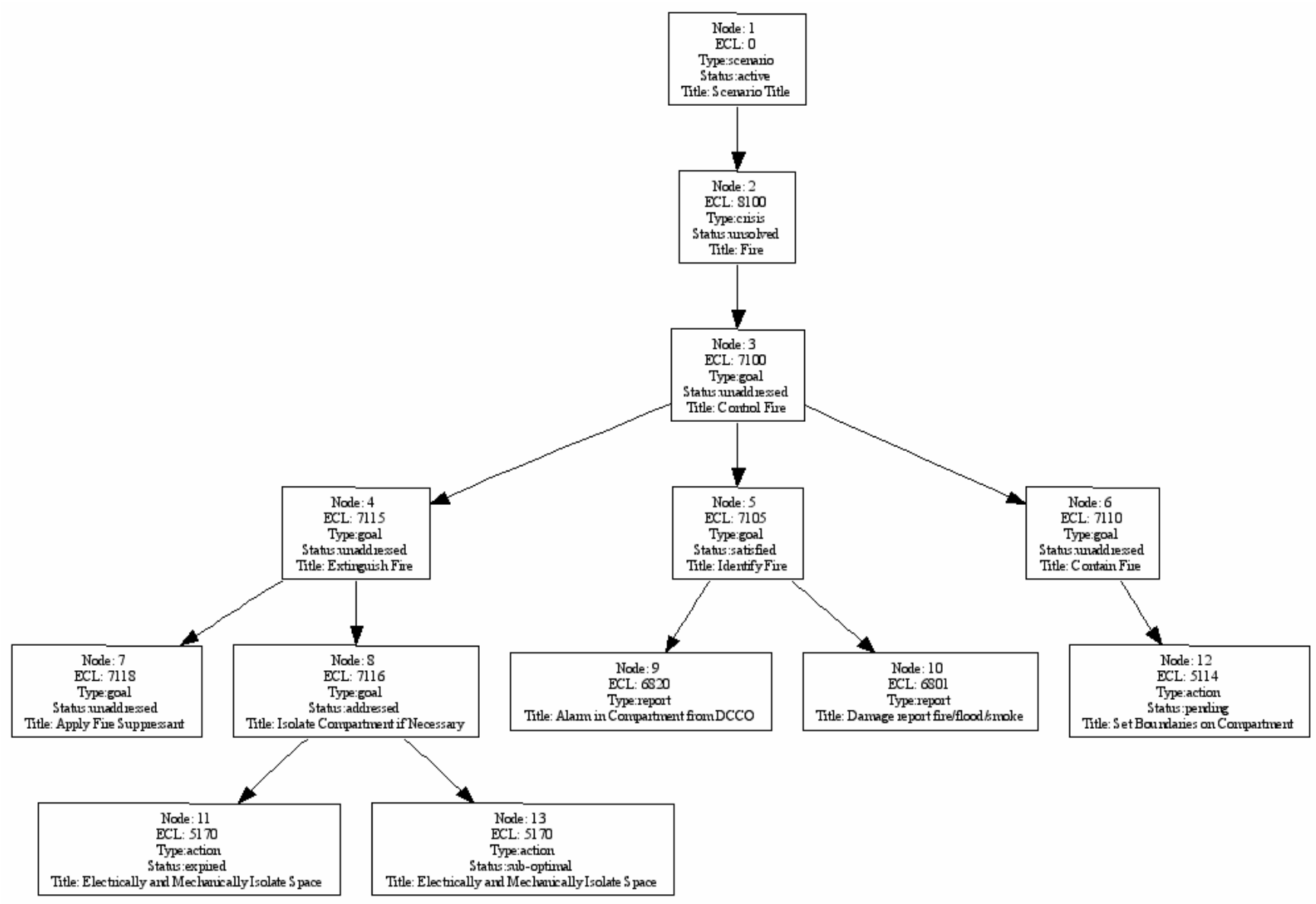

Figure 11 - CSG After Event 4

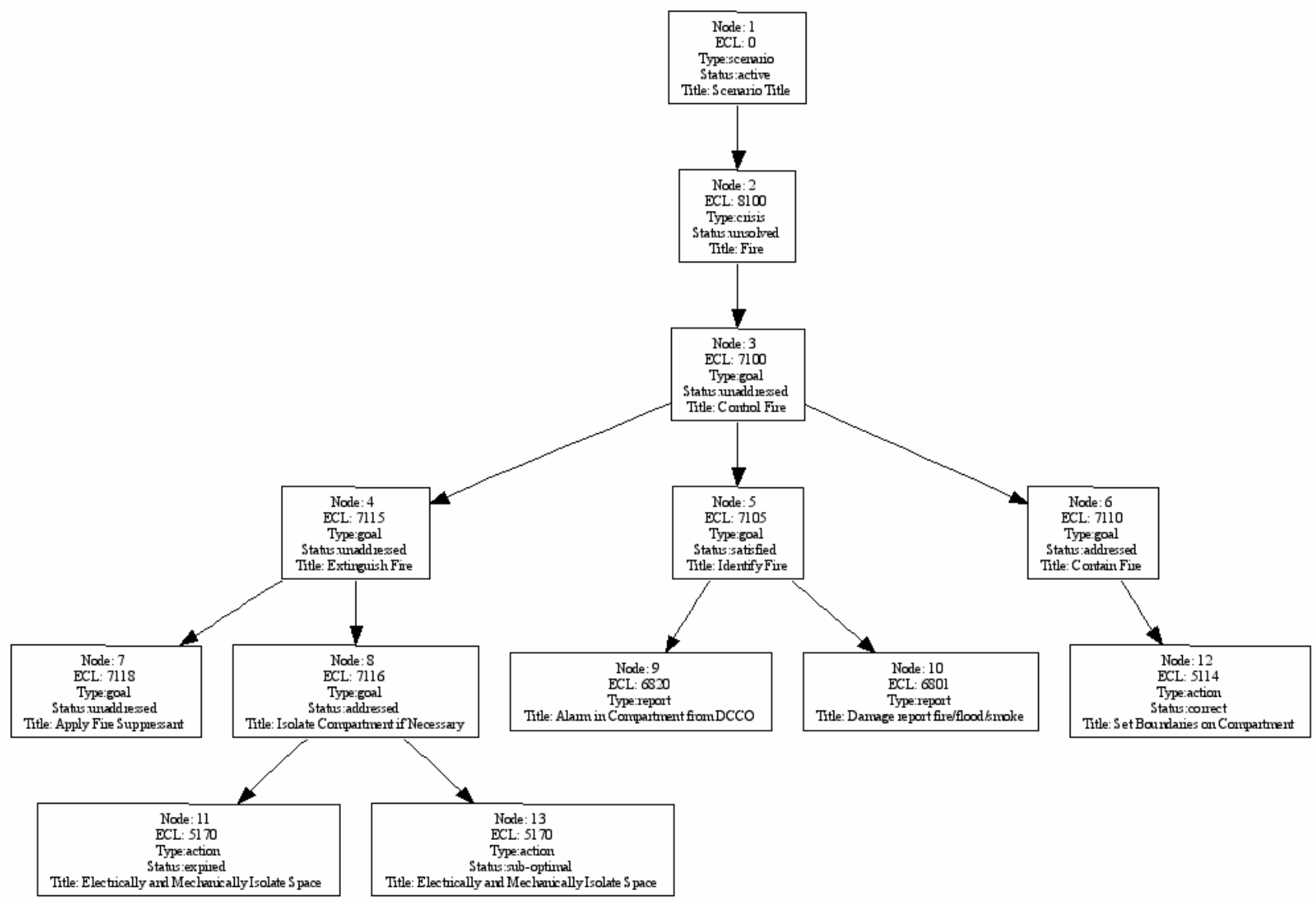

Figure 12 - CSG After Event 5

"Contain Fire" Goal (Node 6) addressed and the "Set Boundaries on Compartment" PendingAction (Node 12) as correct. For this event, the DCA decision coincides with the ideal expert action.

Event 6: DCA incorrectly orders crew to move into compartment 4-174-0-E to fight the fire before ordering desmoking

The last event of this scenario is a decision by the DCA to send fire fighters into compartment 4-174-0-E to fight the fire, given through ECL 5120 "Fight fire in space". There is no PendingAction node in the CSG for ECL 5120, so it is apparent that this was an incorrect action.

```
RULE 5120.fight-fire.critique.2 "preconditions for firefighting not
       met"
IF
    goal(find, unaddressed, 7118, "Apply Fire Suppressant",
       [compartment = Compartment], _, G)
    AND EITHER
      NOT goal(find, satisfied, 7116, "Isolate Compartment if
       Necessary", [compartment = Compartment], _, _)
    OR
      NOT goal(find, satisfied, 7117, "Active Desmoke if Necessary",
       [compartment = Compartment], _, _)
    END EITHER
THEN
    action(create, error-of-commission, 5120, "Fight Fire in Space",
           [compartment = Compartment, station = Station], G, _)
END RULE
```

When GLINT executes GMO 5120, the IF block of rule 5120.fight-fire.critique.2 "preconditions for firefighting not met" evaluates to TRUE. The first statement in the IF block searches the CSG for the Goal "Apply Fire Suppressant", which exists (Node 7). The following EITHER…OR block evaluates to TRUE if either of two Goals have not been satisfied. The first Goal, "Isolate Compartment if Necessary" is addressed in Event 3 (Node 8). The second Goal, "Active Desmoke if Necessary", cannot be found in the CSG, indicating that the preconditions for firefighting have not been met. Upon execution, this rule creates a new ErrorOfCommission node, as shown in Figure 13.

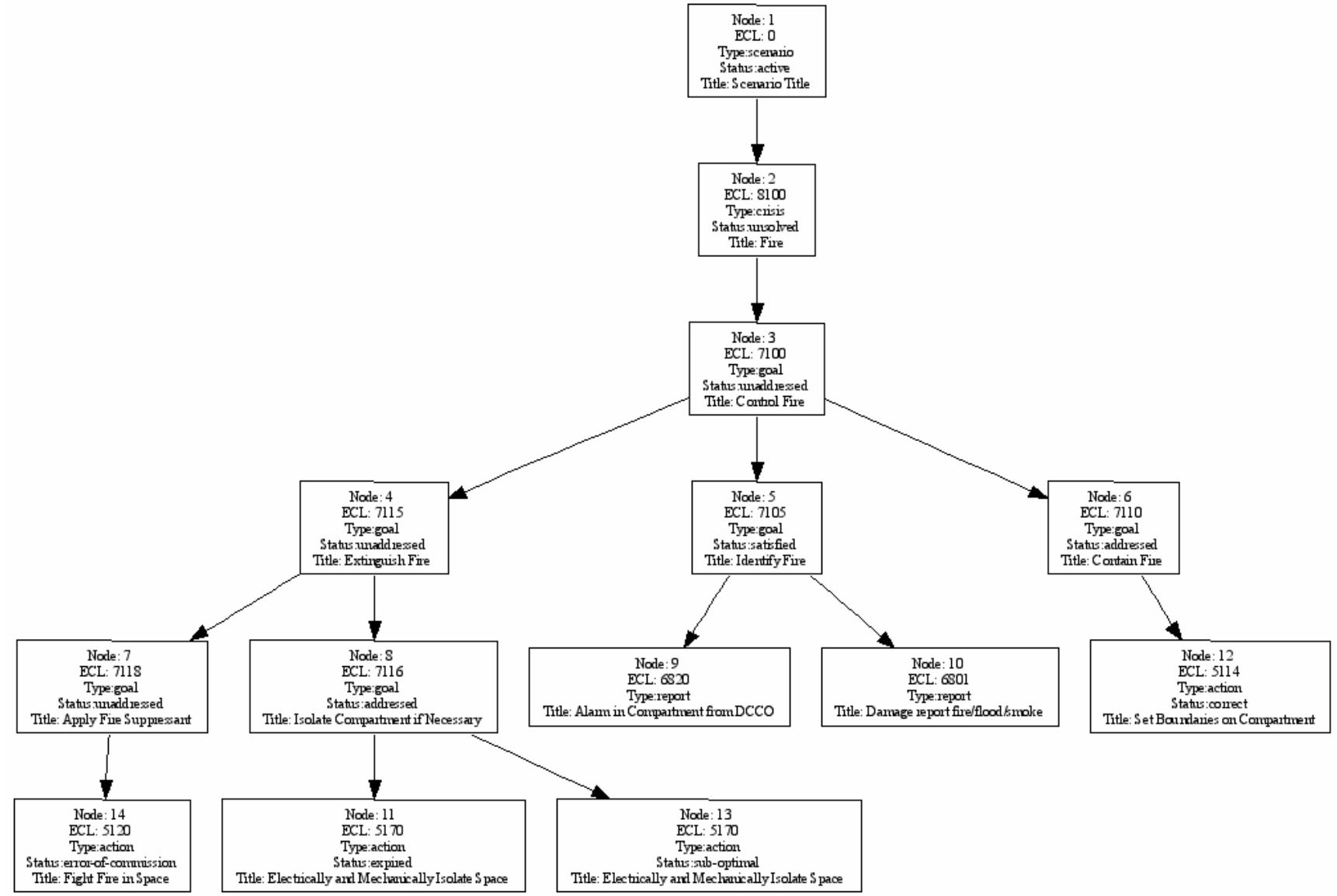

Figure 13 - CSG After Event 6

6. FUTURE WORK

The Gerona language and GLINT 1.0 implementation discussed in this thesis are building blocks for many avenues of upcoming expert and tutoring system research. This section outlines some future work planned around Gerona and GLINT.

6.1. Meta-GMOs

The concept of Meta-GMOs was introduced in Section 2.6 above. To recap, Meta-GMOs are much like GMOs, with the primary difference that Meta-GMOs also have the capability to reason over the Gerona GMO parse tree in addition to the CSG and any static domain information that GMOs can access.

Meta-GMOs are written in Gerona, and thus share nearly common syntax with GMOs. Therefore, the majority of the Gerona interpreter framework needed to build a Meta-GMO parse tree has already been implemented. There is significant infrastructure necessary to enable this search of the GMO parse tree that is, however, yet to be implemented.

The main remaining research challenge left in the implementation of Meta-GMO support is the development of techniques for search and reasoning over the semantics of GMOs and the GMO parse tree. Meta-GMO routines require the capability to search the parse tree for both GMOs and Rules that match certain criteria, and use the parsed instructions within those structures to generate natural, English-language, descriptions of the actions taking place. There are technical hurdles to overcome before Meta-GMO support can be completed, however initial research into this area is extremely promising.

6.2. Question Answering and Answer Justification

As previously discussed, Meta-GMOs are tools for answering student questions. Their goal is to search the CSG, static domain data, and GMO parse tree for answers to questions posed by students during a training scenario. In order to both accept questions and return answers in natural language, future research in the field of Natural Language Processing (NLP) is required.

The first NLP task at large is mapping from a student question utterance in natural English to a corresponding Meta-GMO, which can be executed to search for and build the answer. A variance of the DC-Train 4.0 system in use at Stanford University already has much of the technical capability to do this. Student speech is recorded via a microphone, mapped to plain text, and then converted to a Logical Forms representation for further NLP. [6] The task remaining is to learn a mapping from the Logical Form of a question to the correct answer Meta-GMO. Machine learning techniques will likely be used to learn this representation, although the actual solution requires future research.

The second NLP task is compiling answers to student questions into a natural language response. Meta-GMOs have the difficult job of locating answers within the CSG and GMO-encoded static expert knowledge, however work is required to convert the individual pieces of an answer into convincing text that addresses the question that was asked. A simple initial approach to this is suggested in [5].

6.3. Automated Tutoring and Student Models

The final major avenue of related future work is in the area of automated tutoring. A primary tenet of tutoring theory is the construction and estimation of student models. That is, the student's internal representation of the task that needs to be learned, what concepts are well-understood, and what gaps in knowledge exist. With an accurate student model, automated systems can direct questioning, review, and critiquing with the intent of better teaching the student about topics they do not understand.

The DC-Train 4.0 simulation and GLINT interpreter and parser make an excellent framework for long-term automated tutoring research. The CSG provides a strong beginning for research into student models, because it already contains the entire dynamic state of the scenario at any point in time. Decisions made by the student during a scenario are reflected in the CSG, and estimation about what topics a student does or does not understand can be made from examining critiquing trends within the CSG.

Tutoring theory is a large field in both Psychology and Artificial Intelligence, and much additional preliminary research must to be done before serious treatment of this long-term goal can begin. Gerona and GLINT were designed with this eventual use in mind.

7. CONCLUSION

In this thesis the Gerona Language Interpreter (GLINT) version 1.0 has been presented as a platform for knowledge representation and inference for the task of crisis decision-making. GLINT 1.0 parses and interprets GMOs for the Gerona language, a novel programming language that supports multiple dimensions of expertise within an expert system, including problem-solving, critiquing, explanation, question-answering, knowledge acquisition and learning, and intelligent tutoring.

The Gerona language was introduced as a means to encode expert domain knowledge and provide an explicit, powerful, and uniform knowledge representation as well as improved readability for non-technical domain experts. When combined with the Causal Story Graph, Gerona encodes all static and dynamic knowledge within a domain. Gerona and a CSG together provide an expert model, critiquing model, and question-answering model for learning and tutoring.

The architecture of GLINT was introduced as the engine behind Gerona, and is the result of over a year of research in the Knowledge-Based Systems group at the University of Illinois at Urbana-Champaign.

Ultimately, Gerona and GLINT 1.0 are tools for future work in the fields of automated critiquing, question-answering, and tutoring. The framework presented here is only an early step in a series of new research endeavors in these fields. With luck, this work will be used both for improved training of Damage Control Assistants on Navy ships, and promising research in the years to come.

REFERENCES

1. Bulitko, V. V., "MINERVA-5: A Multifunctional Dynamic Expert System," Masters Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 1998.

2. Carbonari, et al., "Knowledge Ontology Structures for DC-Train 4.0," Knowledge-Based Systems Lab Report UIUC-BI-KBS-2001-0026, January 2001.

3. Clancey, W. J., "NEOMYCIN: Reconfiguring a Rule-Based System with Application to Teaching," in *Readings in Medical Artificial Intelligence*. Reading, MA: Addison-Wesley, 1984.

4. Fried, D. M., "Design of an Expert Critiquing System for Crisis Decision-Making," Masters Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2002.

5. Fried, D. M., and Wilkins, D. C., "A Knowledge Ontology that Supports Expert, Critiquing and Student models: DCX 3.0," Knowledge-Based Systems Report UIUC-BI-KBS-2003-001, University of Illinois at Urbana-Champaign, 2003.

6. Fried, D. M., and Wilkins, D. C., Grois, E., Peters, S., Shultz, K., Clark, B., "The Gerona Knowledge Ontology and Its Support for Spoken Dialogue Tutoring of Crisis Decision Making Skills," in *The 3rd Workshopon Knowledge and Reasoning in Practical Dialogue Systems*, 2003.

7. GNU Flex / Bison Projects, http://www.gnu.org/software/{flex, bison}

8. Grois, E., and Wilkins, D. C., "Comprehensive Intelligent Agent Descriptions for DC-Train 4.0," Knowledge-Based Systems Lab Report UIUC-BI-KBS-2001-0038, May 2001.

9. Hamman, M., and Wilkins, D. C., "Design of a Communications Layer for the Intelligent Reasoning Module of DC-Train 4.0," Knowledge-Based Systems Lab Report UIUC-BI-KBS-2001-0032, April 2001.

10. Hoemmen, M., Carbonari, R., and Wilkins, D. C., "DC-Train 4.0 Users Manual," Knowledge-Based Systems Lab Report UIUC-BI-KBS-2001-0041, November 2001.

11. Park, Tan, Donoho, and Wilkins, "MINERVA: A Knowledge-Based System Shell with Declarative Representation and Flexible Control," Technical Report UIUC-KBS-001, Rev. 3, Department of Computer Science, University of Illinois at Urbana-Champaign, 1992.

12. Ramachandran, S., "KBS Orientation to the Damage Control Domain: The DDG-5 and the DCA," Knowledge-Based Systems Lab Report UIUC-BI-KBS-2001-0027, Beckman Institute, University of Illinois at Urbana-Champaign, January 2001.

APPENDIX A – GLOSSARY OF NAVY TERMS

A.1. Abbreviations

AFFF – Aqueous Film-Forming Foam (fire suppressant)

CSG – Causal Story Graph

CO – Commanding Officer

CSMC – Combat Systems Maintenance Central

DCA – Damage Control Assistant

DCCO – Damage Control Console Operator

DCX – Damage Control Expert (automated critiquing software for DC-Train)

ECL – Event Communication Language (DC-Train message protocol)

EOOW – Engineering Officer of the Watch

GMO – Graph Modification Operator

GQ – General Quarters

HCI – Human-Computer Interface

MRZ – Manned and Ready and Zebra (readiness condition for ship personnel)

RL – Repair Locker

A.2. Definitions

Boundary – Mechanism for cordoning off an area of the ship to prevent spread of fire, flood, or smoke.

Damage Control Central (DC Central) – Compartment where the DCA is located.

Deck – Level on the ship. The deck on the water line is 0. Each deck above that is numbered beginning with a zero—01, 02, 03, etc. Each deck below is numbered with no zero—1, 2, 3, etc.

Firemain – System of pumps, pipes, and valves that supplies water for firefighting and fire control operations.

General Quarters – Order given by the commanding officer (CO) which causes personnel to achieve "manned and ready" status and to set Zebra on all systems.

Halon – An automatic fire-suppressant system used in some compartments on a ship to protect vital equipment. Halon is a gas that smothers fire. Personnel must be evacuated from a compartment before halon can be used, so it is not instantaneous.

Isolation – Cutting off power (electrical isolation), ventilation (mechanical isolation), or both to a compartment to make it safer for firefighting

Jurisdiction – Each compartment has a repair locker assigned to it, based on its location in the ship. The repair locker is said to have jurisdiction over the compartment, though other RLs can be assigned to handle crises there.

Magazine – Ammunition stored in a compartment. If magazines heat up, they may explode, potentially sinking the ship. Magazines have their own special high temperature alarms, and may be flooded with water to prevent explosion.

Manned and Ready – A station is manned and ready when all of its personnel are in the proper compartment, and have achieved a state of preparedness for a possible crisis.

Net80 – General purpose phone talker who can contact anyone on the ship (see: Phone Talker)

Phone Talker – Person in DC Central who has a direct communications link with a particular repair locker (see also: Net80)

Port – The left side of the ship, facing the front. Usually denoted with even numbers (for instance, fire pumps 2, 4, and 6 are on the port side of the DDG51).

Repair Locker – Group of repair personnel on a ship; contains investigators, fire teams, and general-purpose repair personnel. On the DDG51, there are three repair lockers, Repair2, Repair3, and Repair5.

Starboard – The right side of the ship, facing the front. Usually denoted with odd numbers (for instance, fire pumps 1, 3, and 5 are on the starboard side of the DDG51).

Zebra – Material condition of maximum readiness for crises on all systems. During Zebra, the firemain is segregated into two loops, each with a separate firemain pump on, for maximum redundancy.

APPENDIX B – GLINT 1.0 GERONA GMO TOKENS AND RESERVED WORDS

Gerona Reserved Words:

GMO	AND
"END GMO"	IF
RULE	THEN
"END RULE"	EITHER
SUBROUTINE	OR
"END SUBROUTINE"	"END EITHER"
"FOR ECL"	"->"
WHERE	NOT
LET	CALL
TRUE	

Gerona Token Regular Expressions:

Regular expression	**Token**
\+-]?[0-9][0-9]*	INT_NUM
\"[a-zA-Z0-9_\-\\\/\(\)\;\.\,]*\"	STRING_LITERAL
[a-zA-Z][a-zA-Z0-9_\-\\\/]*	STRING
"_"	WILDCARD

Gerona Operators:

“=”	“<~”
“>”	“>~”
“<”	“!~”
“<=”	“<>”
“>=”	

APPENDIX C – GLINT 1.0 GERONA GMO PRODUCTION GRAMMAR

```
rule 1    program -> dgmo
rule 2    program -> subroutine
rule 3    program -> program dgmo
rule 4    program -> program subroutine
rule 5    dgmo -> dgmo_start FOR_ECL identifier rule_list END_GMO
rule 6    dgmo -> dgmo_start FOR_ECL identifier WHERE
           where_condition_list rule_list END_GMO
rule 7    dgmo -> dgmo_start FOR_ECL identifier LET assignment_list
           rule_list END_GMO
rule 8    dgmo -> dgmo_start FOR_ECL identifier WHERE
           where_condition_list LET assignment_list rule_list END_GMO
rule 9    dgmo_start -> START_GMO identifier
rule 10   subroutine -> sub_start rule_list END_SUBROUTINE
rule 11   subroutine -> sub_start LET assignment_list rule_list
           END_SUBROUTINE
rule 12   sub_start -> SUBROUTINE STRING
rule 13   call -> CALL STRING '(' binding_list ')'
rule 14   rule_list -> rule
rule 15   rule_list -> rule_list rule
rule 16   rule -> rule_start IF logical_block THEN procedural_block
           END_RULE
rule 17   rule_start -> RULE identifier
rule 18   logical_block -> TRUE
rule 19   logical_block -> gclause
rule 20   logical_block -> NOT gclause
rule 21   logical_block -> control_either
rule 22   logical_block -> NOT control_either
rule 23   logical_block -> logical_block AND gclause
rule 24   logical_block -> logical_block AND NOT gclause
rule 25   logical_block -> logical_block AND control_either
rule 26   logical_block -> logical_block AND NOT control_either
rule 27   procedural_block -> gclause
rule 28   procedural_block -> control_for
rule 29   procedural_block -> call
```

```
rule 30   procedural_block -> rule
rule 31   procedural_block -> procedural_block gclause
rule 32   procedural_block -> procedural_block control_for
rule 33   procedural_block -> procedural_block call
rule 34   procedural_block -> procedural_block rule
rule 35   gclause -> STRING '(' STRING ',' string_set ',' int_set
           ',' string_set ',' gclause_bindings ',' wild_var ','
           wild_var ')'
rule 36   gclause_bindings -> '[' ']'
rule 37   gclause_bindings -> '[' condition_list ']'
rule 38   control_either -> EITHER logical_block or_list END_EITHER
rule 39   or_list -> OR logical_block
rule 40   or_list -> or_list OR logical_block
rule 41   control_for -> FOR_EACH STRING FOR_IN STRING
           procedural_block END_FOR
rule 42   where_condition_list -> condition
rule 43   where_condition_list -> where_condition_list AND condition
rule 44   condition_list -> condition
rule 45   condition_list -> condition_list ',' condition
rule 46   condition -> STRING COMPARISON STRING_LITERAL
rule 47   condition -> STRING COMPARISON STRING
rule 48   condition -> STRING COMPARISON INT_NUM
rule 49   condition -> INT_NUM COMPARISON INT_NUM
rule 50   assignment_list -> assignment
rule 51   assignment_list -> assignment_list ',' assignment
rule 52   assignment -> STRING_LITERAL ARROW STRING
rule 53   assignment -> INT_NUM ARROW STRING
rule 54   assignment -> assignment_var_set ARROW STRING
rule 55   assignment_var_set -> STRING
rule 56   assignment_var_set -> '[' assignment_var_list ']'
rule 57   assignment_var_list -> STRING
rule 58   assignment_var_list -> assignment_var_list ',' STRING
rule 59   identifier -> INT_NUM
rule 60   identifier -> INT_NUM STRING_LITERAL
rule 61   identifier -> INT_NUM '.' STRING
rule 62   identifier -> INT_NUM '.' STRING STRING_LITERAL
rule 63   identifier -> INT_NUM '.' STRING '.' INT_NUM
```

```
rule 64   identifier -> INT_NUM '.' STRING '.' INT_NUM
          STRING_LITERAL
rule 65   identifier -> INT_NUM '.' STRING '.' STRING
rule 66   identifier -> INT_NUM '.' STRING '.' STRING STRING_LITERAL
rule 67   identifier -> INT_NUM '.' STRING '.' STRING '.' INT_NUM
rule 68   identifier -> INT_NUM '.' STRING '.' STRING '.' INT_NUM
          STRING_LITERAL
rule 69   identifier -> STRING
rule 70   identifier -> STRING STRING_LITERAL
rule 71   identifier -> STRING '.' INT_NUM
rule 72   identifier -> STRING '.' INT_NUM STRING_LITERAL
rule 73   identifier -> STRING '.' STRING
rule 74   identifier -> STRING '.' STRING STRING_LITERAL
rule 75   identifier -> STRING '.' STRING '.' INT_NUM
rule 76   identifier -> STRING '.' STRING '.' INT_NUM STRING_LITERAL
rule 77   identifier -> INT_NUM '.' STRING '.' INT_NUM '.' INT_NUM
rule 78   identifier -> INT_NUM '.' STRING '.' INT_NUM '.' INT_NUM
          STRING_LITERAL
rule 79   identifier -> INT_NUM '.' STRING '.' STRING '.' INT_NUM
          '.' INT_NUM
rule 80   identifier -> INT_NUM '.' STRING '.' STRING '.' INT_NUM
          '.' INT_NUM STRING_LITERAL
rule 81   identifier -> STRING '.' INT_NUM '.' INT_NUM
rule 82   identifier -> STRING '.' INT_NUM '.' INT_NUM
          STRING_LITERAL
rule 83   identifier -> STRING '.' STRING '.' INT_NUM '.' INT_NUM
rule 84   identifier -> STRING '.' STRING '.' INT_NUM '.' INT_NUM
          STRING_LITERAL
rule 85   wild_var -> STRING
rule 86   wild_var -> WILDCARD
rule 87   string_set -> WILDCARD
rule 88   string_set -> STRING
rule 89   string_set -> STRING_LITERAL
rule 90   string_set -> '[' string_list ']'
rule 91   string_list -> STRING
rule 92   string_list -> STRING_LITERAL
rule 93   string_list -> string_list ',' STRING
rule 94   string_list -> string_list ',' STRING_LITERAL
```

```
rule 95   int_set -> INT_NUM
rule 96   int_set -> '[' int_list ']'
rule 97   int_list -> INT_NUM
rule 98   int_list -> int_list ',' INT_NUM
rule 99   binding_list -> binding
rule 100  binding_list -> binding_list ',' binding
rule 101  binding -> STRING COMPARISON STRING_LITERAL
rule 102  binding -> STRING COMPARISON STRING
rule 103  binding -> STRING COMPARISON INT_NUM
```

APPENDIX D – DC-TRAIN 4.0 ECL MESSAGES

The following is the master list of ECL messages in the DC-Train 4.0 system. This details all the messages that can be sent between the GLINT system and/or student DCA, and the simulated agents. The first column, *ID*, is the code of the message, and the second column is the message's name.

The third column in the table is the method for translating an instance of the message with its various fields into an English sentence. Each bracketed word is a key that must appear in that message's fields. To translate to English, replace each bracketed key with its corresponding value.

ID	Name	Description
5101	Flood magazine	[to], [target], [source], [from], Flood magazine compartment, [compartment]
5105	Investigate space	[to], [target], [source], [from], Investigate compartment, [compartment]
5107	Investigate Firemain or Chillwater	[to], [target], [source], [from], Investigate, [loop], on level, [deck]
5108	Investigate Firemain or Chillwater in space	[to], [target], [source], [from], Investigate, [loop], in compartment, [compartment]
5110	Set boundaries fire/flood/smoke	[to], [target], [source], [from], Set, [casualty], boundaries as follows, [saft], [paft], [pfore], [sfore], [above], [below]
5114	Set Boundaries fire/flood/smoke	[to], [target], [source], [from], Set, [casualty], boundaries on compartment [compartment]
5120	Fight fire in space	[to], [target], [source], [from], Enter space and fight fire in compartment, [compartment]
5130	Dewater space	[to], [target], [source], [from], Dewater compartment, [compartment]
5140	Desmoke space remove smoke after fire is out	[to], [target], [source], [from], Desmoke compartment, [compartment]
5142	Active desmoke space remove smoke to prepare for firefighting – adjacent space to one on fire	[to], [target], [source], [from], Active desmoke compartment, [compartment]
5144	Desmoke space using overpressurization	[to],[target],[source],[from],Desmoke compartment using overpressurization, [compartment]
5150	Open/close Firemain valve	[to], [target], [source], [from], [valve_action], valve, [valve]
5156	Start/stop fire pump	[to], [target], [source], [from], [pump_action], fire pump, [pump]
5162	Isolate and patch Firemain or Chillwater rupture	[to], [target], [source], [from], Isolate and patch, [loop], rupture in compartment, [compartment], using valve, [valve], and valve, [valve]
5166	Patch Firemain or Chillwater rupture in place	[to], [target], [source], [from], Patch firemain, [loop], rupture in compartment, [compartment], in place
5170	Mechanically isolate space	[to], [target], [source], [from], Mechanically isolate compartment, [compartment]
5171	Electrically isolate space	[to],[target],[source],[from],Electrically isolate

ID	Name	Description
		compartment,[compartment]
5176	Patch hull rupture	[to], [target], [source], [from], Patch hull rupture, deck, [deck], frame, [frame]
5180	Set reflash watch	[to], [target], [source], [from], Set reflash watch in compartment, [compartment]
5182	Secure reflash watch	[to], [target], [source], [from], Secure reflash watch in compartment, [compartment]
5186	Route casualty power	[to], [target], [source], [from], Route casualty power from, [power_point], to, [power_point]
5190	Conduct post-fire gas-free test	[to], [target], [source], [from], Conduct post-fire gas-free test in compartment, [compartment]
5195	Assist another repair locker	[to], [target], [source], [from], Assist repair, [repair_locker], with, [casualty], in compartment, [compartment]
5198	Investigators contact repair locker sent via 1MC	[to], [target], [source], [from], Contact Repair, [repair_locker]
5201	Firemain Zebra status query to DCCO	[to], [target], [source], [from], Report status of Zebra on Firemain
5204	Valve status query to DCCO	[to], [target], [source], [from], Report status of firemain valve, [valve]
5206	Fire pump status query to DCCO	[to], [target], [source], [from], Report status of fire pump, [pump]
5208	Alarm status query to DCCO	[to], [target], [source], [from], Report status of alarm in compartment, [compartment]
5212	Firemain pressure query to DCCO	[to], [target], [source], [from], Report Firemain pressure [side], side
5501	Report status of MR/Z can also be used by CO	[to], [target], [source], [from], Report status of manned and ready, and Zebra
5503	Report status of MR/Z	[to], [target], [source], [from], Report status of manned and ready, and Zebra
5505	Report status of Zebra	[to], [target], [source], [from], Report status of Zebra
5510	Request permission to flood magazine	[to], [target], [source], [from], Request permission to flood magazine compartment [compartment]
5520	Request permission to start fire pump	[to], [target], [source], [from], Request permission to start number [pump], fire pump
5530	Report status of damage	[to], [target], [source], [from], report status of, [casualty], in compartment, [compartment]
5535	Request report of engineering plant status	[to],[target],[source],[from], Report the status of the engineering plant
5537	Request report of status of a compartment's E&M isolation	[to],[target],[source],[from], Report status of E&M isolation in compartment, [compartment]
5540	Recommend got to GQ to CO	[to], [target], [source], [from], Recommend go to General Quarters
5550	Damage report to CO	[to], [target], [source], [from], [casualty], in compartment, [compartment]
5601	MR/Z set report ship-wide	[to], [target], [source], [from], all stations are manned and ready, Zebra is set
5602	MR/Z in progress report ship-wide	[to], [target], [source], [from], all stations are manned and ready, Zebra is in progress
5603	MR in progress report station-specific	[to], [target], [source], [from], reports manned and ready in progress
5604	MR/Z in progress report station-specific	[to], [target], [source], [from], reports manned and ready, Zebra is in progress
5605	MR/Z set report station-specific	[to], [target], [source], [from], reports manned and ready, Zebra is set
5901	Belay my last cancels last order	[to], [target], [source], [from], Belay my last!

ID	Name	Description
5910	DCA acknowledge	[to], [target], [source], [from], DCA, Aye!
6101	Acknowledge order to flood space	[to], [target], [source], [from], Flood compartment, [compartment], Aye!
6105	Acknowledge order to investigate space	[to], [target], [source], [from], Investigate compartment, [compartment], Aye!
6107	Acknowledge order to investigate Firemain or Chillwater	[to], [target], [source], [from], Investigate, [loop], on level, [deck], Aye!
6108	Acknowledge order to Investigate Firemain or Chillwater in space	[to], [target], [source], [from], Investigate, [loop], in compartment, [compartment], Aye!
6110	Acknowledge order to Set boundaries fire/flood/smoke	[to], [target], [source], [from], Set, [casualty], boundaries as follows, [saft], [paft], [pfore], [sfore], [above], [below], Aye!
6114	Acknowledge order to set boundaries on compartment fire/flood/smoke	[to], [target], [source], [from], Set, [casualty], boundaries on compartment [compartment] Aye!
6120	Acknowledge order to fight fire in space	[to], [target], [source], [from], Fight fire in compartment, [compartment], Aye!
6130	Acknowledge order to dewater space	[to], [target], [source], [from], Dewater compartment, [compartment], Aye!
6140	Acknowledge order to desmoke space	[to], [target], [source], [from], Desmoke compartment, [compartment], Aye!
6142	Acknowledge order to active desmoke space	[to], [target], [source], [from], Active desmoke compartment, [compartment], Aye!
6144	Acknowledge order to Desmoke space using overpressurization	[to],[target],[source],[from],Desmoke compartment using overpressurization, [compartment], Aye!
6150	Acknowledge order to open/close Firemain valve	[to], [target], [source], [from], [valve_action], valve, [valve], Aye!
6156	Acknowledge order to start/stop fire pump	[to], [target], [source], [from], [pump_action], fire pump, [pump], Aye!
6162	Acknowledge order to Isolate and patch Firemain or Chillwater rupture	[to], [target], [source], [from], Isolate and patch, [loop], rupture in compartment, [compartment], using valve, [valve], and valve, [valve], Aye!
6166	Acknowledge order to patch Firemain or Chillwater rupture in place	[to], [target], [source], [from], Patch, [loop], rupture in compartment, [compartment], Aye!
6170	Acknowledge order to mechanically isolate space	[to], [target], [source], [from], Mechanically isolate compartment, [compartment], Aye!
6171	Acknowledge order to Electrically isolate space	[to],[target],[source],[from],Electrically isolate compartment,[compartment], Aye!
6176	Acknowledge order to Patch hull rupture	[to], [target], [source], [from], Patch hull rupture, deck, [deck], frame, [frame], Aye!
6180	Acknowledge order to set reflash watch	[to], [target], [source], [from], Set reflash watch in space, [compartment], Aye!
6182	Acknowledge order to secure reflash watch	[to], [target], [source], [from], Secure reflash watch in space, [compartment], Aye!
6186	Acknowledge order to Route casualty power	[to], [target], [source], [from], Route casualty power from, [power_point], to, [power_point], Aye!
6190	Acknowledge order to Conduct post-fire gas-free test	[to], [target], [source], [from], Conduct post-fire gas-free test in compartment, [compartment], Aye!
6195	Acknowledge order to Assist another repair locker	[to], [target], [source], [from], Assist repair, [repair_locker], with, [casualty], in compartment, [compartment], Aye!
6201	Flooding magazine	[to], [target], [source], [from], Reports magazine flooding activated

ID	Name	Description
		compartment, [compartment]
6210	Setting boundaries fire/flood/smoke in response to query	[to], [target], [source], [from], Reports, [casualty], boundaries on compartment, [compartment], in progress
6214	Setting boundaries fire/flood/smoke in response to query	[to], [target], [source], [from], Reports, [casualty], boundaries on bulkheads in progress
6220	Fighting fire in space in response to request of status of firefighting	[to], [target], [source], [from], Reports eight fire team members have activated SCBAs, time, [time], and fire team has entered compartment, [compartment]
6230	Dewatering space	[to], [target], [source], [from], Reports dewatering in progress, compartment, [compartment]
6240	Desmoking space	[to], [target], [source], [from], Reports desmoking in progress, compartment, [compartment]
6242	Active desmoking space	[to], [target], [source], [from], Reports active desmoking in progress, compartment, [compartment]
6262	Firemain or Chillwater rupture isolated, and patching in progress	[to], [target], [source], [from], Reports, [loop], rupture in compartment, [compartment], isolated, patching in progress
6266	Patching Firemain or Chillwater rupture in place	[to], [target], [source], [from], Reports patching in progress on, [loop], rupture in compartment, [compartment]
6270	Electrical and mechanical isolation in progress	[to], [target], [source], [from], Reports electrical and mechanical isolation in progress on compartment, [compartment]
6276	Patching hull rupture	[to], [target], [source], [from], Reports hull rupture patching in progress in compartment, [compartment]
6301	Magazine flooded	[to], [target], [source], [from], Reports magazine flooding system secured, compartment, [compartment]
6310	Boundaries set fire/flood/smoke	[to], [target], [source], [from], Reports, [casualty], boundaries set on compartment, [compartment]
6314	Boundaries set fire/flood/smoke	[to], [target], [source], [from], Reports, [casualty], boundaries set on bulkheads
6320	Fire contained	[to], [target], [source], [from], Reports fire in compartment, [compartment], contained
6324	Fire extinguished	[to], [target], [source], [from], Reports fire in compartment, [compartment], out, reflash watch set
6330	Space dewatered	[to], [target], [source], [from], Reports dewatering complete in compartment, [compartment]
6340	Space desmoked	[to], [target], [source], [from], Reports desmoking complete in compartment, [compartment]
6342	Active desmoking rigged	[to], [target], [source], [from], Reports active desmoking rigged in compartment, [compartment]
6350	Firemain valve opened/closed	[to], [target], [source], [from], Reports valve, [valve], is, [valve_action]
6356	Fire pump started/stopped	[to], [target], [source], [from], Reports fire pump, [pump], [pump_state]
6362	Firemain or Chillwater rupture has been patched	[to], [target], [source], [from], Reports, [loop], rupture in compartment, [compartment], has been patched
6370	Space electrically and mechanically isolated	[to], [target], [source], [from], Reports electrical and mechanical isolation in compartment, [compartment], complete
6376	Hull rupture patched	[to], [target], [source], [from], Reports hull rupture patched in compartment, [compartment]
6380	Zebra set on Firemain	[to], [target], [source], [from], Zebra set on Firemain
6401	Valve damaged	[to], [target], [source], [from], Reports valve, [valve], damaged
6404	Fire pump inoperable	[to], [target], [source], [from], Fire pump, [pump] inoperable

ID	Name	Description
6410	Cannot patch Firemain or Chillwater rupture in place	[to], [target], [source], [from], Reports, [loop], rupture in compartment, [compartment], cannot be patched in place
6414	Casualty out of control fire/flood/smoke	[to], [target], [source], [from], Reports, [casualty], in compartment, [compartment], out of control, evacuating space
6420	Halon ineffective	[to], [target], [source], [from], Reports Halon ineffective in compartment, [compartment]
6501	Request for assistance	[to], [target], [source], [from], Requests assistance to combat, [casualty], in compartment, [compartment]
6602	Permission to flood magazine from CO	[to], [target], [source], [from], permission to flood magazine compartment [compartment]
6603	Denied permission to flood magazine from CO	[to], [target], [source], [from], permission denied to flood magazine compartment [compartment]
6610	Permission to start fire pump from EOOW	[to], [target], [source], [from], permission to start fire pump number,[pump]
6611	Denied permission to start fire pump from EOOW	[to], [target], [source], [from], permission denied to start fire pump number, [pump]
6701	Review last order	[to], [target], [source], [from], Request review your last order
6705	Investigate space to investigators	[to], [target], [source], [from], Investigate compartment, [compartment]
6730	Reprimand for flooding magazine w/o permission from CO	[to], [target], [source], [from], DCA! I needed that magazine to fight the ship!
6735	Reprimand for starting fire pump w/o permission from EOOW	[to], [target], [source], [from], DCA! How dare you start a fire pump without my permission?!
6740	Insufficient personnel	[to], [target], [source], [from], Reports not enough personnel to combat, [casualty], in compartment, [compartment]
6745	Insufficient firemain pressure to fight fire	[to], [target], [source], [from], Insufficient firemain pressure to fight fire in compartment,[compartment]
6750	DCCO Recommends starting fire pump before Zebra	[to], [target], [source], [from], DCA, DCCO recommends starting number, [pump], fire pump before going to Zebra on the firemain.
6801	Damage/investigation report fire/flood/smoke note: report class Alpha fire	[to], [target], [source], [from], Reports, [casualty], in compartment, [compartment]
6804	Firemain or Chillwater rupture	[to], [target], [source], [from], Reports, [loop], rupture in compartment, [compartment]
6808	Hull rupture	[to], [target], [source], [from], Reports hull rupture, compartment, [compartment], [side], side, at frame, [frame]
6820	Alarm in compartment from DCCO	[to], [target], [source], [from], [alarm], in compartment, [compartment]
6822	Alarm reset in compartment from DCCO	[to], [target], [source], [from], [alarm], clear in compartment, [compartment]
6824	Firemain pressure normal from DCCO	[to], [target], [source], [from], [loop], pressure normal
6826	Firemain pressure low from DCCO	[to], [target], [source], [from], [side], side firemain pressure low
6830	Valve status report from DCCO or Repair Lockers	[to], [target], [source], [from], Valve, [valve], is, [valve_status]
6832	Fire pump status report from DCCO or Repair Lockers	[to], [target], [source], [from], Fire pump number, [pump], is, [pump_status]
6840	Halon activated	[to], [target], [source], [from], Reports compartment, [compartment], has been evacuated, and Halon activated
6903	Manned and ready, Zebra set throughout the ship announcement to ship	[to], [target], [source], [from], Manned and ready, Zebra set throughout the ship

ID	Name	Description
6920	Inform repair locker that ready to assist	[to], [target], [source], [from]
6922	Transfer tasks to repair locker	[to], [target], [source], [from]
6924	Accept tasks	[to], [target], [source], [from]
6926	Reject tasks	[to], [target], [source], [from]
6980	Missile or mine sighted from Bridge to DCA and CO through 1MC	[to], [target], [source], [from], [weapon], sighted, [side], side
6990	General Quarters	[to], [target], [source], [from], General Quarters! General Quarters! All hands man your battle stations. Set material condition Zebra throughout the ship. Make Zebra reports to DC Central.
6930	Request allocate compartment	[to], [target], [source], [from], [compartment]
6931	Grant compartment allocation	[to], [target], [source], [from], [compartment]
6932	Release compartment allocation	[to], [target], [source], [from], [compartment]
5555	Conduct COMM's	[to], [target], [source], [from], conduct COMM’s
6655	All stations report COMM's set	[to], [target], [source], [from], all stations report COMM’s set
6390	Compartment clear of noxious gases	[to], [target], [source], [from], Reports that compartment, [compartment], is clear of noxious gases
6490	Compartment IS NOT clear of noxious gases	[to], [target], [source], [from], Reports that compartment, [compartment], IS NOT clear of noxious gases
5191	Order to clear noxious gases	[to], [target], [source], [from], Clear compartment, [compartment], of noxious gases

APPENDIX E – GERONA GRAPH MODIFICATION OPERATORS

E.1. Subroutines

```
SUBROUTINE get-new-firemain-orders

LET gmo-name -> Rule,
    loop -> Loop,
    goal -> Goal

  RULE firemain-orders.suggest.1
  IF
    world-state(find, _, 4110, "Orders to Restore Firemain
       Pressure",
              [loop = Loop, pumps = Pumps, open-valves =
       OpenValves, close-valves = CloseValves], _, _)
  THEN
    FOR EACH Pump IN Pumps
      action(create, pending, 5156, "Start/Stop Firemain Pump",
             [pump_action = "start", target = DCCO, pump = Pump],
       Goal, _)
    END FOR

    FOR EACH OpenValve IN OpenValves
      RULE firemain-orders.suggest.2 "cancel pending close valve
       order"
      IF
        action(find, pending, 5150, "Open/Close Firemain Valve",
              [valve_action = "close", valve = OpenValve], _, A)
      THEN
        action(modify, expired, 5150, "Open/Close Firemain Valve",
              [valve_action = "close", valve = OpenValve], _, A)
      END RULE

      RULE 0.firemain-orders.suggest.3 "suggest new open valve
       order"
      IF
        NOT action(find, pending, 5150, "Open/Close Firemain Valve",
              [valve_action = "close", valve = OpenValve], _, _)
        AND world-state(find, _, 4112, "Best Station for Valve",
      [valve = OpenValve, station = Station], _, _)
      THEN
        action(create, pending, 5150, "Open/Close Firemain Valve",
              [valve_action = "open", valve = OpenValve, target =
       Station], Goal, A)
      END RULE
    END FOR

    FOR EACH CloseValve IN CloseValves
```

```
      RULE firemain-orders.suggest.4 "cancel pending open valve
       order"
      IF
        action(find, pending, 5150, "Open/Close Firemain Valve",
               [valve_action = "open", valve = CloseValve], _, A)
      THEN
        action(modify, expired, 5150, "Open/Close Firemain Valve",
               [valve_action = "open", valve = CloseValve], _, A)
      END RULE

      RULE firemain-orders.suggest.5 "suggest new close valve order"
      IF
        NOT action(find, pending, 5150, "Open/Close Firemain Valve",
               [valve_action = "open", valve = CloseValve], _, _)
        AND world-state(find, _, 4112, "Best Station for Valve",
        [valve = CloseValve, station = Station], _, _)
      THEN
        action(create, pending, 5150, "Open/Close Firemain Valve",
               [valve_action = "close", valve = CloseValve, target =
       Station], Goal, A)
      END RULE
    END FOR
  END RULE
END SUBROUTINE

SUBROUTINE reassign-personnel
  LET gmo-name -> Rule,
      station -> Station

  RULE reassign-personnel.interpret.1 "Locate invalid fight fire
       orders"
  IF
    action(find, pending, 5120, "Fight fire in space", [target =
        Station], _, ActionSet)
  THEN
    FOR EACH A IN ActionSet
      RULE reassign-personnel.suggest.1 "Make alternate personnel
       suggestion for fight fire order"
      IF
        action(find, pending, 5120, "Fight fire in space",
        [compartment = Compartment], _, A)
        AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                    [compartment = Compartment, station =
        NewStation], _, _)
      THEN
        action(modify, pending, 5120, "Fight fire in space",
        [station = NewStation], _, A)
      END RULE
    END FOR
  END RULE

  RULE reassign-personnel.interpret.2 "Locate invalid dewater
       orders"
  IF
```

```
    action(find, pending, 5130, "Dewater space", [target = Station],
        _, ActionSet)
  THEN
    FOR EACH A IN ActionSet
      RULE reassign-personnel.suggest.2 "Make alternate personnel
       suggestion for dewater order"
      IF
        action(find, pending, 5130, "Dewater Space", [compartment =
       Compartment], _, A)
        AND world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
                    [compartment = Compartment, station =
       NewStation], _, _)
      THEN
        action(modify, pending, 5130, "Dewater Space", [station =
       NewStation], _, A)
      END RULE
    END FOR
  END RULE

  RULE reassign-personnel.interpret.3 "Locate invalid desmoke
       orders"
  IF
    action(find, pending, 5140, "Desmoke space", [target = Station],
        _, ActionSet)
  THEN
    FOR EACH A IN ActionSet
      RULE reassign-personnel.suggest.3 "Make alternate personnel
       suggestion for desmoke order"
      IF
        action(find, pending, 5140, "Desmoke space", [compartment =
       Compartment], _, A)
        AND world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
                    [compartment = Compartment, station =
       NewStation], _, _)
      THEN
        action(modify, pending, 5140, "Desmoke space", [station =
       NewStation], _, A)
      END RULE
    END FOR
  END RULE
END SUBROUTINE
```

E.2. GMOs for 5000 Level ECL Commands (Student DCA Actions)

```
;*************************************************************
; NOTES
; * implement wrong compartment stuff
;
;*************************************************************

GMO 5101
FOR ECL 5101 "Flood Magazines"
LET compartment -> Compartment

   RULE 5101.flood-magazine.critique.1 "flood magazines order is
        correct"
   IF
      goal(find, unaddressed, 7160, "Prevent Magazine Explosion",
        [compartment = Compartment], _, G)
      AND action(find, pending, 5101, "Flood Magazines",
        [compartment = Compartment], _, A)
      AND report(find, _, 6602, "Permission to Flood Magazine from
        CO", [compartment = Compartment], _, _)
   THEN
      action(modify, correct, 5101, "Flood Magazines", [compartment
        = Compartment], _, A)
      goal(modify, addressed, 7160, "Prevent Magazine Explosion",
        [compartment = Compartment], _, G)
   END RULE


   RULE 5101.flood-magazine.critique.2 "flood magazines incorrect -
        no permission"
   IF
      goal(find, unaddressed, 7160, "Prevent Magazine Explosion",
        [compartment = Compartment], _, G)
      AND NOT report(find, _, 6602, "Permission to Flood Magazine
        from CO", [compartment = Compartment], _, _)
   THEN
      action(create, error-of-commission, 5101, "Flood Magazines",
        [compartment = Compartment], G, _)
      goal(modify, addressed, 7160, "Prevent Magazine Explosion",
        [compartment = Compartment], _, G)
   END RULE


   RULE 5101.flood-magazine.critique.3 "flood magazines incorrect -
        unnecessary"
   IF
      NOT goal(find, unaddressed, 7160, "Prevent Magazine
        Explosion", [compartment = Compartment], _, _)
      AND goal(find, [addressed, satisfed], 7160,"Prevent Magazine
        Explosion", [compartment = Compartment], _, G)
   THEN
      action(create, error-of-commission, 5101, "Flood Magazines",
        [compartment = Compartment], G, _)
```

```
   END RULE


   RULE 5101.flood-magazine.critique.4 "flood magazines order is
        untraceable"
   IF
      NOT goal(find, _, 7160, "Prevent Magazine Explosion",
       [compartment = Compartment], _, _)
   THEN
      action(create, error-of-commission, 5101, "Flood Magazines",
       [compartment = Compartment], scenario-node, _)
   END RULE

END GMO

;*****************************************************************

GMO 5105
FOR ECL 5105 "Investigate Compartment"
  LET compartment -> Compartment,
      target -> Station

  RULE 5105.investigate.critique.1 "investigate order is correct"
  IF
    goal(find, unaddressed, [7105, 7205, 7305], ["Identify Fire",
        "Identify Flooding", "Identify Smoke"],
         [compartment = Compartment], _, G)
    AND action(find, pending, 5105, "Investigate Compartment",
        [compartment = Compartment], _, A)
    AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                [compartment = Compartment, station = Station], _, _)
  THEN
    goal(modify, addressed, [7105, 7205, 7305], ["Identify Fire",
        "Identify Flooding", "Identify Smoke"],
         [compartment = Compartment], _, G)
    action(modify, correct, 5105, "Investigate Compartment",
           [compartment = Compartment, target = Station], _, A)
  END RULE

  RULE 5105.investigate.critique.2 "investigate order sub-optimal -
       wrong repair locker"
  IF
    goal(find, unaddressed, [7105, 7205, 7305], ["Identify Fire",
        "Identify Flooding", "Identify Smoke"],
         [compartment = Compartment], _, G)
    AND action(find, pending, 5105, "Investigate Compartment",
        [compartment = Compartment], P, A)
    AND NOT world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                    [compartment = Compartment, station = Station],
        _, _)
    AND world-state(find, _, 3401, "Has Personnel", [station =
        Station], _, _)
  THEN
```

```
    goal(modify, addressed, [7105, 7205, 7305], ["Identify Fire",
        "Identify Flooding", "Identify Smoke"],
          [compartment = Compartment], _, G)
    action(modify, expired, 5105, "Investigate Compartment", [], _,
        A)
    action(create, sub-optimal, 5105, "Investigate Compartment",
            [compartment = Compartment, target = Station], P, _)
  END RULE

  RULE 5105.investigate.critique.3 "investigate order incorrect - no
       personnel"
  IF
    goal(find, unaddressed, [7105, 7205, 7305], ["Identify Fire",
        "Identify Flooding", "Identify Smoke"],
          [compartment = Compartment], _, G)
    AND action(find, pending, 5105, "Investigate Compartment",
        [compartment = Compartment], P, A)
    AND NOT world-state(find, _, 3401, "Has Personnel", [station =
        Station], _, _)
  THEN
    action(create, error-of-commission, 5105, "Investigate
        Compartment",
            [compartment = Compartment, target = Station], P, _)
  END RULE

  ; RULE 5105.investigate.critique.4 "investigate order incorrect -
        unnecessary"
  ; IF
  ;   NOT goal(find, unaddressed, [7105, 7205, 7305], ["Identify
        Fire", "Identify Flooding", "Identify Smoke"],
  ;         [compartment = Compartment], _, _)
  ;   AND goal(find, [addressed, satisfied], [7105, 7205, 7305],
  ;         ["Identify Fire", "Identify Flooding", "Identify Smoke"],
        [compartment = Compartment], _, G)

  RULE 5105.investigate.critique.5 "investigate order is
       untraceable"
  IF
    NOT goal(find, _, [7105, 7205, 7305], ["Identify Fire",
        "Identify Flooding", "Identify Smoke"],
          [compartment = Compartment], _, _)
  THEN
    action(create, error-of-commission, 5105, "Investigate
        Compartment",
            [compartment = Compartment, target = Station], scenario-
        node, _)
  END RULE
END GMO

;*****************************************************************

GMO 5107
FOR ECL 5107 "Investigate Firemain or Chillwater"
  LET loop -> Loop,
      deck -> Deck,
      target -> Station
```

```
  RULE 5107.investigate.critique.1 "investigate firemain or
       chillwater order is correct"
  IF
    goal(find, unaddressed, 7555, "Investigate Firemain if
       Necessary", [loop = Loop], _, G)
    AND action(find, pending, 5107, "Investigate Firemain or
       Chillwater", [loop = Loop], _, A)
    AND world-state(find, _, 3401, "Has Personnel", [station =
       Station], _, _)
  THEN
    goal(modify, addressed, 7555, "Investigate Firemain if
       Necessary", [loop = Loop], _, G)
    action(modify, correct, 5107, "Investigate Firemain or
       Chillwater", [loop = Loop, target = Station], _, A)
  END RULE

  RULE 5107.investigate.critique.2 "investigate firemain or
       chillwater incorrect - no personnel"
  IF
    goal(find, unaddressed, 7555, "Investigate Firemain if
       Necessary", [loop = Loop], _, G)
    AND action(find, pending, 5107, "Investigate Firemain or
       Chillwater", [loop = Loop], P, A)
    AND NOT world-state(find, _, 3401, "Has Personnel", [station =
       Station], _, _)
  THEN
    action(create, error-of-commission, 5107, "Investigate Firemain
       or Chillwater",
           [loop = Loop, target = Station], P, _)
  END RULE

  RULE 5107.investigate.critique.3 "investigate firemain or
       chillwater incorrect - unnecessary"
  IF
    NOT goal(find, unaddressed, 7555, "Investigate Firemain if
       Necessary", [loop = Loop], _, _)
    AND goal(find, [addressed, satisfied], 7555, "Investigate
       Firemain if Necessary", [loop = Loop], _, G)
  THEN
    action(create, error-of-commission, 5107, "Investigate Firemain
       or Chillwater",
           [loop = Loop, target = Station], G, _)
  END RULE

  RULE 5107.investigate.critique.4 "investigate firemain or
       chillwater order is untraceable"
  IF
    NOT goal(find, _, 7555, "Investigate Firemain if Necessary",
       [loop = Loop], _, _)
  THEN
    action(create, error-of-commission, 5107, "Investigate Firemain
       or Chillwater",
           [loop = Loop, target = Station], scenario-node, _)
  END RULE
END GMO
```

```
;*****************************************************************

GMO 5110.fire
FOR ECL 5110 "Set Boundaries"
WHERE casualty = "fire"
  LET target -> Station,
      saft -> SAft,
      paft -> PAft,
      pfor -> PFor,
      sfor -> SFor,
      above -> Above,
      below -> Below

  RULE 5110.set-fire-boundaries.critique.1 "set fire boundaries
       order is correct"
  IF
    action(find, pending, 5110, "Set Boundaries",
      [casualty = "fire", saft = SAft, paft = PAft, pfor = PFor,
       sfor = SFor, above = Above, below = Below], _, A)
    AND world-state(find, _, 4304, "Best Repair Locker for
       Boundaries",
              [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
       above = Above, below = Below, station = Station], _, _)
  THEN
    action(modify, correct,  5110, "Set Boundaries",
           [target = Station, casualty = "fire", saft = SAft, paft =
       PAft, pfor = PFor, sfor = SFor, above = Above, below =
       Below], _, A)

    RULE 5110.set-fire-boundaries.critique.1.1 "mark contain fire
       goals addressed"
    IF
      world-state(find, _, 4351, "Boundaries to Compartments",
                 [saft = SAft, paft = PAft, pfor = PFor, sfor =
       SFor, above = Above, below = Below, compartments = CompSet],
       _, _)
      AND goal(find-all, unaddressed, 7110, "Contain Fire",
       [compartment <~ CompSet], _, GoalSet)
    THEN
      goal(modify, addressed, 7110, "Contain Fire", [], _, GoalSet)
    END RULE
  END RULE

  RULE 5110.set-fire-boundaries.critique.2 "should have set
       boundaries on compartment - sub-optimal"
  IF
    NOT action(find, pending, 5110, "Set Boundaries",
      [casualty = "fire", paft = PAft, pfor = PFor, sfor = SFor,
       saft = SAft, above = Above, below = Below], _, _)
    AND world-state(find, _, 4351, "Boundaries to Compartments",
                [paft = PAft, pfor = PFor, sfor = SFor, saft = SAft,
       above = Above, below = Below, compartments = CompSet], _, _)
    AND goal(find, unaddressed, 7110, "Contain Fire", [compartment
       <~ CompSet], _, G)
    AND action(find, pending, 5114, "Set Boundaries on Compartment",
```

```
        [casualty = "fire", compartment = Compartment], P, _)
  AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
THEN
  action(create, sub-optimal, 5110, "Set Boundaries",
        [casualty = "fire", paft = PAft, pfor = PFor, sfor =
      SFor, saft = SAft, above = Above, below = Below, target =
      Station], P, _)

  RULE 5110.set-fire-boundaries.critique.2.1 "mark contain fire
      goals addressed"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
              [paft = PAft, pfor = PFor, sfor = SFor, saft =
      SAft, above = Above, below = Below, compartments = CompSet],
      _, _)
    AND goal(find-all, unaddressed, 7110, "Contain Fire",
      [compartment <~ CompSet], _, GoalSet)
  THEN
    goal(modify, addressed, 7110, "Contain Fire", [], _, GoalSet)
  END RULE
END RULE

RULE 5110.set-fire-boundaries.critique.3 "different fire
      boundaries - sub-optimal"
IF
  NOT action(find, pending, 5110, "Set Boundaries",
    [casualty = "fire", paft = PAft, pfor = PFor, sfor = SFor,
      saft = SAft, above = Above, below = Below], _, _)
  AND world-state(find, _, 4351, "Boundaries to Compartments",
            [paft = PAft, pfor = PFor, sfor = SFor, saft = SAft,
      above = Above, below = Below, compartments = CompSet], _, _)
  AND goal(find, unaddressed, 7110, "Contain Fire", [compartment
      <~ CompSet], _, G)
  AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
THEN
  action(create, sub-optimal, 5110, "Set Boundaries",
        [casualty = "fire", paft = PAft, pfor = PFor, sfor =
      SFor, saft = SAft, above = Above, below = Below, target =
      Station], G, _)

  RULE 5110.set-fire-boundaries.critique.3.1 "mark contain fire
      goals addressed"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
              [paft = PAft, pfor = PFor, sfor = SFor, saft =
      SAft, above = Above, below = Below, compartments = CompSet],
      _, _)
    AND goal(find-all, unaddressed, 7110, "Contain Fire",
      [compartment <~ CompSet], _, GoalSet)
  THEN
    goal(modify, addressed, 7110, "Contain Fire", [], _, GoalSet)
  END RULE
END RULE
```

```
RULE 5110.set-fire-boundaries.critique.4 "wrong repair locker for
     set fire boundaries"
IF
  action(find, pending, 5110, "Set Boundaries",
      [casualty = "fire", paft = PAft, pfor = PFor, sfor = SFor,
      saft = SAft, above = Above, below = Below], P, A)
  AND NOT world-state(find, _, 4304, "Best Repair Locker for
      Boundaries",
                  [paft = PAft, pfor = PFor, sfor = SFor, saft =
      SAft, above = Above, below = Below, station = Station], _, _)
  AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
THEN
  action(modify, expired,  5110, "Set Boundaries",
         [casualty = "fire", target = Station, paft = PAft, pfor =
      PFor, sfor = SFor, saft = SAft, above = Above, below =
      Below], _, A)
  action(create, sub-optimal,  5110, "Set Boundaries",
         [casualty = "fire", target = Station, paft = PAft, pfor =
      PFor, sfor = SFor, saft = SAft, above = Above, below =
      Below], P, _)

  RULE 5110.set-fire-boundaries.critique.4.1 "mark contain fire
      goals addressed"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
                [paft = PAft, pfor = PFor, sfor = SFor, saft =
    SAft, above = Above, below = Below, compartments = CompSet],
    _, _)
    AND goal(find-all, unaddressed, 7110, "Contain Fire",
      [compartment <~ CompSet], _, GoalSet)
  THEN
    goal(modify, addressed, 7110, "Contain Fire", [], _, GoalSet)
  END RULE
END RULE


RULE 5110.set-fire-boundaries.critique.5 "set fire boundaries is
     unnecessary"
IF
  world-state(find, _, 4351, "Boundaries to Compartments",
              [paft = PAft, pfor = PFor, sfor = SFor, saft = SAft,
      above = Above, below = Below, compartments = CompSet], _, _)
  AND NOT goal(find, unaddressed, 7110, "Contain Fire",
      [compartment <~ CompSet], _, _)
  AND goal(find, [addressed, satisfied], 7110, "Contain Fire",
      [compartment <~ CompSet], _, _)
THEN
  action(create, error-of-commission,  5110, "Set Boundaries",
         [casualty = "fire", target = Station, paft = PAft, pfor =
      PFor, sfor = SFor, saft = SAft, above = Above, below =
      Below], G, _)
END RULE

RULE 5110.set-fire-boundaries.critique.6 "set fire boundaries
     untraceable"
```

```
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
                [paft = PAft, pfor = PFor, sfor = SFor, saft = SAft,
        above = Above, below = Below, compartments = CompSet], _, _)
    AND NOT goal(find, _, 7110, "Contain Fire", [compartment <~
        CompSet], _, _)
  THEN
    action(create, error-of-commission,  5110, "Set Boundaries",
           [casualty = "fire", target = Station, paft = PAft, pfor =
        PFor, sfor = SFor, saft = SAft, above = Above, below =
        Below], scenario-node, _)
  END RULE
END GMO

;*****************************************************************

GMO 5110.flood
FOR ECL 5110 "Set Boundaries"
WHERE casualty = "flood"
  LET target -> Station,
      saft -> SAft,
      paft -> PAft,
      pfor -> PFor,
      sfor -> SFor,
      above -> Above,
      below -> Below

  RULE 5110.set-flood-boundaries.critique.1 "set flood boundaries
       order is correct"
  IF
    action(find, pending, 5110, "Set Boundaries",
      [casualty = "flood", saft = SAft, paft = PAft, pfor = PFor,
       sfor = SFor, above = Above, below = Below], _, A)
    AND world-state(find, _, 4304, "Best Repair Locker for
        Boundaries",
                [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
        above = Above, below = Below, station = Station], _, _)
  THEN
    action(modify, correct,  5110, "Set Boundaries",
           [target = Station, casualty = "flood", saft = SAft, paft
        = PAft, pfor = PFor, sfor = SFor, above = Above, below =
        Below], _, A)

    RULE 5110.set-flood-boundaries.critique.1.1 "mark contain flood
        goals addressed"
    IF
      world-state(find, _, 4351, "Boundaries to Compartments",
                  [saft = SAft, paft = PAft, pfor = PFor, sfor =
      SFor, above = Above, below = Below, compartments = CompSet],
       _, _)
      AND goal(find-all, unaddressed, 7210, "Contain Flooding",
       [compartment <~ CompSet], _, GoalSet)
    THEN
      goal(modify, addressed, 7210, "Contain Flooding", [], _,
       GoalSet)
    END RULE
```

```
END RULE

RULE 5110.set-flood-boundaries.critique.2 "should have set
     boundaries on compartment - sub-optimal"
IF
  NOT action(find, pending, 5110, "Set Boundaries",
    [casualty = "flood", saft = SAft, paft = PAft, pfor = PFor,
     sfor = SFor, above = Above, below = Below], _, _)
  AND world-state(find, _, 4351, "Boundaries to Compartments",
            [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
     above = Above, below = Below, compartments = CompSet], _, _)
  AND goal(find, unaddressed, 7210, "Contain Flooding",
     [compartment <~ CompSet], _, G)
  AND action(find, pending, 5114, "Set Boundaries on Compartment",
        [casualty = "flood", saft = SAft, paft = PAft, pfor =
     PFor, sfor = SFor, above = Above, below = Below], P, _)
  AND world-state(find, _, 3401, "Has Personnel", [station =
     Station], _, _)
THEN
  action(create, sub-optimal, 5110, "Set Boundaries",
        [casualty = "flood", saft = SAft, paft = PAft, pfor =
     PFor, sfor = SFor, above = Above, below = Below, target =
     Station], P, _)

  RULE 5110.set-flood-boundaries.critique.2.1 "mark contain flood
     goals addressed"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
              [saft = SAft, paft = PAft, pfor = PFor, sfor =
     SFor, above = Above, below = Below, compartments = CompSet],
     _, _)
    AND goal(find-all, unaddressed, 7210, "Contain Flooding",
     [compartment <~ CompSet], _, GoalSet)
  THEN
    goal(modify, addressed, 7210, "Contain Flooding", [], _,
     GoalSet)
  END RULE
END RULE

RULE 5110.set-flood-boundaries.critique.3 "different flood
     boundaries - sub-optimal"
IF
  NOT action(find, pending, 5110, "Set Boundaries",
    [casualty = "flood", saft = SAft, paft = PAft, pfor = PFor,
     sfor = SFor, above = Above, below = Below], _, _)
  AND world-state(find, _, 4351, "Boundaries to Compartments",
              [saft = SAft, paft = PAft, pfor = PFor, sfor =
     SFor, above = Above, below = Below, compartments = CompSet],
     _, _)
  AND goal(find, unaddressed, 7210, "Contain Flooding",
     [compartment <~ CompSet], _, G)
  AND world-state(find, _, 3401, "Has Personnel", [station =
     Station], _, _)
THEN
  action(create, sub-optimal, 5110, "Set Boundaries",
```

```
          [casualty = "flood", saft = SAft, paft = PAft, pfor =
        PFor, sfor = SFor, above = Above, below = Below, target =
        Station], G, _)

    RULE 5110.set-flood-boundaries.critique.3.1 "mark contain flood
        goals addressed"
    IF
      world-state(find, _, 4351, "Boundaries to Compartments",
                  [saft = SAft, paft = PAft, pfor = PFor, sfor =
        SFor, above = Above, below = Below, compartments = CompSet],
        _, _)
      AND goal(find-all, unaddressed, 7210, "Contain Flooding",
        [compartment <~ CompSet], _, GoalSet)
    THEN
      goal(modify, addressed, 7210, "Contain Flooding", [], _,
        GoalSet)
    END RULE
  END RULE

  RULE 5110.set-flood-boundaries.critique.4 "wrong repair locker for
       set flood boundaries"
  IF
    action(find, pending, 5110, "Set Boundaries",
      [casualty = "flood", saft = SAft, paft = PAft, pfor = PFor,
        sfor = SFor, above = Above, below = Below], P, A)
    AND NOT world-state(find, _, 4304, "Best Repair Locker for
        Boundaries",
                [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
        above = Above, below = Below, station = Station], _, _)
    AND world-state(find, _, 3401, "Has Personnel", [station =
        Station], _, _)
  THEN
    action(modify, expired,  5110, "Set Boundaries",
          [casualty = "flood", target = Station, saft = SAft, paft
        = PAft, pfor = PFor, sfor = SFor, above = Above, below =
        Below], _, A)
    action(create, sub-optimal,  5110, "Set Boundaries",
          [casualty = "flood", target = Station, saft = SAft, paft
        = PAft, pfor = PFor, sfor = SFor, above = Above, below =
        Below], P, _)

    RULE 5110.set-flood-boundaries.critique.4.1 "mark contain flood
        goals addressed"
    IF
      world-state(find, _, 4351, "Boundaries to Compartments",
                  [saft = SAft, paft = PAft, pfor = PFor, sfor =
        SFor, above = Above, below = Below, compartments = CompSet],
        _, _)
      AND goal(find-all, unaddressed, 7210, "Contain Flooding",
        [compartment <~ CompSet], _, GoalSet)
    THEN
      goal(modify, addressed, 7210, "Contain Flooding", [], _,
        GoalSet)
    END RULE
  END RULE
```

```
  RULE 5110.set-flood-boundaries.critique.5 "set flood boundaries is
       unnecessary"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
                [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
       above = Above, below = Below, compartments = CompSet], _, _)
    AND NOT goal(find, unaddressed, 7210, "Contain Flooding",
       [compartment <~ CompSet], _, _)
    AND goal(find, [addressed, satisfied], 7210, "Contain Flooding",
       [compartment <~ CompSet], _, _)
  THEN
    action(create, error-of-commission,  5110, "Set Boundaries",
           [casualty = "flood", target = Station, saft = SAft, paft
       = PAft, pfor = PFor, sfor = SFor, above = Above, below =
       Below], G, _)
  END RULE

  RULE 5110.set-flood-boundaries.critique.6 "set flood boundaries
       untraceable"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
                [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
       above = Above, below = Below, compartments = CompSet], _, _)
    AND NOT goal(find, _, 7210, "Contain Flooding", [compartment <~
       CompSet], _, _)
  THEN
    action(create, error-of-commission,  5110, "Set Boundaries",
           [casualty = "flood", target = Station, saft = SAft, paft
       = PAft, pfor = PFor, sfor = SFor, above = Above, below =
       Below], scenario-node, _)
  END RULE
END GMO

;****************************************************************

GMO 5110.smoke
FOR ECL 5110 "Set Boundaries"
WHERE casualty = "smoke"
  LET target -> Station,
      saft -> SAft,
      paft -> PAft,
      pfor -> PFor,
      sfor -> SFor,
      above -> Above,
      below -> Below

  RULE 5110.set-smoke-boundaries.critique.1 "set smoke boundaries
       order is correct"
  IF
    action(find, pending, 5110, "Set Boundaries",
      [casualty = "smoke", saft = SAft, paft = PAft, pfor = PFor,
       sfor = SFor, above = Above, below = Below], _, A)
    AND world-state(find, _, 4304, "Best Repair Locker for
       Boundaries",
                [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
       above = Above, below = Below, station = Station], _, _)
```

```
THEN
  action(modify, correct,  5110, "Set Boundaries",
         [target = Station, casualty = "smoke", saft = SAft, paft
      = PAft, pfor = PFor, sfor = SFor, above = Above, below =
      Below], _, A)

  RULE 5110.set-smoke-boundaries.critique.1.1 "mark contain smoke
      goals addressed"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
                [saft = SAft, paft = PAft, pfor = PFor, sfor =
      SFor, above = Above, below = Below, compartments = CompSet],
      _, _)
    AND goal(find-all, unaddressed, 7310, "Contain Smoke",
      [compartment <~ CompSet], _, GoalSet)
  THEN
    goal(modify, addressed, 7310, "Contain Smoke", [], _, GoalSet)
  END RULE
END RULE

RULE 5110.set-smoke-boundaries.critique.2 "should have set
      boundaries on compartment - sub-optimal"
IF
  NOT action(find, pending, 5110, "Set Boundaries",
    [casualty = "smoke", saft = SAft, paft = PAft, pfor = PFor,
      sfor = SFor, above = Above, below = Below], _, _)
  AND world-state(find, _, 4351, "Boundaries to Compartments",
              [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
      above = Above, below = Below, compartments = CompSet], _, _)
  AND goal(find, unaddressed, 7310, "Contain Smoke", [compartment
      <~ CompSet], _, G)
  AND action(find, pending, 5114, "Set Boundaries on Compartment",
         [casualty = "smoke", saft = SAft, paft = PAft, pfor =
      PFor, sfor = SFor, above = Above, below = Below], P, _)
  AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
THEN
  action(create, sub-optimal, 5110, "Set Boundaries",
         [casualty = "smoke", saft = SAft, paft = PAft, pfor =
      PFor, sfor = SFor, above = Above, below = Below, target =
      Station], P, _)

  RULE 5110.set-smoke-boundaries.critique.2.1 "mark contain smoke
      goals addressed"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
                [saft = SAft, paft = PAft, pfor = PFor, sfor =
      SFor, above = Above, below = Below, compartments = CompSet],
      _, _)
    AND goal(find-all, unaddressed, 7310, "Contain Smoke",
      [compartment <~ CompSet], _, GoalSet)
  THEN
    goal(modify, addressed, 7310, "Contain Smoke", [], _, GoalSet)
  END RULE
END RULE
```

```
RULE 5110.set-smoke-boundaries.critique.3 "different smoke
     boundaries - sub-optimal"
IF
  NOT action(find, pending, 5110, "Set Boundaries",
    [casualty = "smoke", saft = SAft, paft = PAft, pfor = PFor,
     sfor = SFor, above = Above, below = Below], _, _)
  AND world-state(find, _, 4351, "Boundaries to Compartments",
            [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
     above = Above, below = Below, compartments = CompSet], _, _)
  AND goal(find, unaddressed, 7310, "Contain Smoke", [compartment
     <~ CompSet], _, G)
  AND world-state(find, _, 3401, "Has Personnel", [station =
     Station], _, _)
THEN
  action(create, sub-optimal, 5110, "Set Boundaries",
         [casualty = "smoke", saft = SAft, paft = PAft, pfor =
     PFor, sfor = SFor, above = Above, below = Below, target =
     Station], G, _)

  RULE 5110.set-smoke-boundaries.critique.3.1 "mark contain smoke
     goals addressed"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
              [saft = SAft, paft = PAft, pfor = PFor, sfor =
     SFor, above = Above, below = Below, compartments = CompSet],
     _, _)
    AND goal(find-all, unaddressed, 7310, "Contain Smoke",
     [compartment <~ CompSet], _, GoalSet)
  THEN
    goal(modify, addressed, 7310, "Contain Smoke", [], _, GoalSet)
  END RULE
END RULE

RULE 5110.set-smoke-boundaries.critique.4 "wrong repair locker for
     set smoke boundaries"
IF
  action(find, pending, 5110, "Set Boundaries",
    [casualty = "smoke", saft = SAft, paft = PAft, pfor = PFor,
     sfor = SFor, above = Above, below = Below], P, A)
  AND NOT world-state(find, _, 4304, "Best Repair Locker for
     Boundaries",
              [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
     above = Above, below = Below, station = Station], _, _)
  AND world-state(find, _, 3401, "Has Personnel", [station =
     Station], _, _)
THEN
  action(modify, expired,  5110, "Set Boundaries",
         [casualty = "smoke", target = Station, saft = SAft, paft
     = PAft, pfor = PFor, sfor = SFor, above = Above, below =
     Below], _, A)
  action(create, sub-optimal,  5110, "Set Boundaries",
         [casualty = "smoke", target = Station, saft = SAft, paft
     = PAft, pfor = PFor, sfor = SFor, above = Above, below =
     Below], P, _)
```

```
    RULE 5110.set-smoke-boundaries.critique.4.1 "mark contain smoke
        goals addressed"
    IF
      world-state(find, _, 4351, "Boundaries to Compartments",
                  [saft = SAft, paft = PAft, pfor = PFor, sfor =
      SFor, above = Above, below = Below, compartments = CompSet],
      _, _)
      AND goal(find-all, unaddressed, 7310, "Contain Smoke",
        [compartment <~ CompSet], _, GoalSet)
    THEN
      goal(modify, addressed, 7310, "Contain Smoke", [], _, GoalSet)
    END RULE
  END RULE

  RULE 5110.set-smoke-boundaries.critique.5 "set smoke boundaries is
       unnecessary"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
                [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
      above = Above, below = Below, compartments = CompSet], _, _)
    AND NOT goal(find, unaddressed, 7310, "Contain Smoke",
        [compartment <~ CompSet], _, _)
    AND goal(find, [addressed, satisfied], 7310, "Contain Smoke",
        [compartment <~ CompSet], _, _)
  THEN
    action(create, error-of-commission,  5110, "Set Boundaries",
           [casualty = "smoke", target = Station, saft = SAft, paft
      = PAft, pfor = PFor, sfor = SFor, above = Above, below =
      Below], G, _)
  END RULE

  RULE 5110.set-smoke-boundaries.critique.6 "set smoke boundaries
       untraceable"
  IF
    world-state(find, _, 4351, "Boundaries to Compartments",
                [saft = SAft, paft = PAft, pfor = PFor, sfor = SFor,
      above = Above, below = Below, compartments = CompSet], _, _)
    AND NOT goal(find, _, 7310, "Contain Smoke", [compartment <~
        CompSet], _, _)
  THEN
    action(create, error-of-commission,  5110, "Set Boundaries",
           [casualty = "smoke", target = Station, saft = SAft, paft
      = PAft, pfor = PFor, sfor = SFor, above = Above, below =
      Below], scenario-node, _)
  END RULE
END GMO

;*****************************************************************
; GRD - This wasn't in the Doc, so I wrote some reasonable code for
;       my own use that will probably need to be changed in the
       future
;*****************************************************************

GMO 5114.fire
FOR ECL 5114 "Set Boundaries on Compartment"
WHERE casualty = "fire"
```

```
LET compartment -> Compartment,
    target -> Station

  RULE 5114.set-fire-boundaries.critique.1 "correct fire boundaries"
  IF
    goal(find, unaddressed, 7110, "Contain Fire", [compartment =
        Compartment], _, G)
    AND action(find, pending, 5114, "Set Boundaries on Compartment",
           [casualty = "fire", compartment = Compartment], _, A)
    AND world-state(find, _, 3401, "Has Personnel", [station =
        Station], _, _)
  THEN
    action(modify, correct, 5114, "Set Boundaries on Compartment",
           [casualty = "fire", compartment = Compartment], _, A)
    goal(modify, addressed, 7110, "Contain Fire", [compartment =
        Compartment], _, G)
  END RULE

  ; Handle other cases

END GMO

;*****************************************************************

GMO 5120
FOR ECL 5120 "fight fire in space"
  LET compartment -> Compartment,
      target -> Station

  RULE 5120.fight-fire.critique.1 "preconditions for firefighting
       are met"
  IF
    goal(find, unaddressed, 7118, "Apply Fire Suppressant",
        [compartment = Compartment], _, G)
    AND action(find, pending, 5120, "Fight Fire in Space",
        [compartment = Compartment], P, A)
    AND goal(find, satisfied, 7116, "Isolate Compartment if
        Necessary", [compartment = Compartment], _, _)
    AND goal(find, satisfied, 7117, "Active Desmoke if Necessary",
        [compartment = Compartment], _, _)
  THEN

    RULE 5120.fight-fire.critique.1.1 "fight fire order is correct"
    IF
      world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = Station], _,
        _)
    THEN
      action(modify, correct, 5120, "Fight Fire in Space",
        [compartment = Compartment, station = Station], _, A)
      goal(modify, addressed, 7118, "Apply Fire Suppressant",
        [compartment = Compartment], _, G)
    END RULE
```

```
  RULE 5120.fight-fire.critique.1.2 "fight fire order sub-optimal
      - wrong repair locker"
  IF
    NOT world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                    [compartment = Compartment, station = Station],
      _, _)
    AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
    action(modify, expired, 5120, "Fight Fire in Space",
      [compartment = Compartment], _, A)
    action(create, sub-optimal, 5120, "Fight Fire in Space",
           [compartment = Compartment, station = Station], P, _)
    goal(modify, addressed, 7118, "Apply Fire Suppressant",
      [compartment = Compartment], _, G)
  END RULE

  RULE 5120.fight-fire.critique.1.3 "fight fire order incorrect -
      no personnel"
  IF
    NOT world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
    action(create, error-of-commission, 5120, "Fight Fire in
      Space",
           [compartment = Compartment, station = Station], P, _)
  END RULE
END RULE

RULE 5120.fight-fire.critique.2 "preconditions for firefighting
      not met"
IF
  goal(find, unaddressed, 7118, "Apply Fire Suppressant",
      [compartment = Compartment], _, G)
  AND EITHER
    NOT goal(find, satisfied, 7116, "Isolate Compartment if
      Necessary", [compartment = Compartment], _, _)
  OR
    NOT goal(find, satisfied, 7117, "Active Desmoke if Necessary",
      [compartment = Compartment], _, _)
  END EITHER
THEN
  action(create, error-of-commission, 5120, "Fight Fire in Space",
         [compartment = Compartment, station = Station], G, _)
END RULE

RULE 5120.fight-fire.critique.3 "fight fire order incorrect -
      unnecessary"
IF
  NOT goal(find, unaddressed, 7118, "Apply Fire Suppressant",
      [compartment = Compartment], _, _)
  AND goal(find, [addressed, satisfied], 7118, "Apply Fire
      Suppressant", [compartment = Compartment], _, G)
THEN
  action(create, error-of-commission, 5120, "Fight Fire in Space",
```

```
          [compartment = Compartment, station = Station], G, _)
  END RULE

  RULE 5120.fight-fire.critique.4 "fight fire order is untraceable"
  IF
    NOT goal(find, _, 7118, "Apply Fire Suppressant", [compartment =
        Compartment], _, _)
  THEN
    action(create, error-of-commission, 5120, "Fight Fire in Space",
           [compartment = Compartment, station = Station], scenario-
        node, _)
  END RULE
END GMO

;*************************************************************

GMO 5130
FOR ECL 5130 "Dewater Space"
  LET compartment -> Compartment,
      target -> Station

  RULE 5130.dewater.critique.1 "preconditions for dewatering (flood
       crisis) are met"
  IF
    goal(find, unaddressed, 7218, "Remove Water", [compartment =
        Compartment], _, G)
    AND action(find, pending, 5130, "Dewater Space", [compartment =
        Compartment], P, A)
    AND goal(find, satisfied, 7116, "Isolate Compartment if
        Necessary", [compartment = Compartment], _, _)
  THEN

    RULE 5130.dewater.critique.1.1 "dewater order is correct"
    IF
      world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = Station], _,
        _)
    THEN
      action(modify, correct, 5130, "Dewater Space", [compartment =
        Compartment, station = Station], _, A)
      goal(modify, addressed, 7218, "Remove Water", [compartment =
        Compartment], _, G)
    END RULE

    RULE 5130.dewater.critique.1.2 "dewater order sub-optimal -
        wrong repair locker"
    IF
      NOT world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                    [compartment = Compartment, station = Station],
        _, _)
      AND world-state(find, _, 3401, "Has Personnel", [station =
        Station], _, _)
    THEN
```

```
    action(modify, expired, 5130, "Dewater Space", [compartment =
      Compartment], _, A)
    action(create, sub-optimal, 5130, "Dewater Space",
      [compartment = Compartment, station = Station], P, _)
    goal(modify, addressed, 7218, "Remove Water", [compartment =
      Compartment], _, G)
  END RULE

  RULE 5130.dewater.critique.1.3 "dewater order incorrect - no
     personnel"
  IF
    NOT world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
    action(create, error-of-commission, 5130, "Dewater Space",
           [compartment = Compartment, station = Station], P, _)
  END RULE
END RULE

RULE 5130.dewater.critique.2 "preconditions for dewatering (after
     firefighting) are met"
IF
  goal(find, unaddressed, 7130, "Clear Water from Firefighting",
     [compartment = Compartment], _, G)
  AND action(find, pending, 5130, "Dewater Space", [compartment =
     Compartment], P, A)
  AND goal(find, satisfied, 7118, "Apply Fire Suppressant",
     [compartment = Compartment], _, _)
THEN

  RULE 5130.dewater.critique.3.0 "dewater order is correct"
  IF
    world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                [compartment = Compartment, station = Station], _,
     _)
  THEN
    action(modify, correct, 5130, "Dewater Space", [compartment =
      Compartment, station = Station], _, A)
    goal(modify, addressed, 7130, "Clear Water from Firefighting",
      [compartment = Compartment], _, G)
  END RULE

  RULE 5130.dewater.critique.2.2 "dewater order sub-optimal -
     wrong repair locker"
  IF
    NOT world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                    [compartment = Compartment, station = Station],
     _, _)
    AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
    action(modify, expired, 5130, "Dewater Space", [compartment =
      Compartment], _, A)
```

```
    action(create, sub-optimal, 5130, "Dewater Space",
      [compartment = Compartment, station = Station], P, _)
    goal(modify, addressed, 7130, "Clear Water from Firefighting",
      [compartment = Compartment], _, G)
  END RULE

  RULE 5130.dewater.critique.2.3 "dewater order incorrect - no
     personnel"
  IF
    NOT world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
    action(create, error-of-commission, 5130, "Dewater Space",
           [compartment = Compartment, station = Station], P, _)
  END RULE
END RULE

RULE 5130.dewater.critique.3 "preconditions for dewatering (flood
     crisis) not met"
IF
  goal(find, unaddressed, 7218, "Remove Water", [compartment =
     Compartment], _, G)
  AND NOT goal(find, satisfied, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, _)
THEN
  action(create, error-of-commission, 5130, "Dewater Space",
         [compartment = Compartment, station = Station], G, _)
END RULE

RULE 5130.dewater.critique.4 "preconditions for dewatering (after
     firefighting) not met"
IF
  goal(find, unaddressed, 7130, "Clear Water from Firefighting",
     [compartment = Compartment], _, G)
  AND NOT goal(find, satisfied, 7118, "Apply Fire Suppressant",
     [compartment = Compartment], _, _)
THEN
  action(create, error-of-commission, 5130, "Dewater Space",
         [compartment = Compartment, station = Station], G, _)
END RULE

RULE 5130.dewater.critique.5 "dewater order incorrect -
     unnecessary"
IF
  NOT goal(find, unaddressed, 7218, "Remove Water", [compartment =
     Compartment], _, _)
  AND NOT goal(find, unaddressed, 7130, "Clear Water from
     Firefighting", [compartment = Compartment], _, _)
  AND EITHER
    goal(find, [addressed, satisfied], 7218, "Remove Water",
      [compartment = Compartment], _, G)
  OR
    goal(find, [addressed, satisfied], 7130, "Clear Water from
      Firefighting",
         [compartment = Compartment], _, G)
  END EITHER
```

```
  THEN
    action(create, error-of-commission, 5130, "Dewater Space",
           [compartment = Compartment, station = Station], G, _)
  END RULE

  RULE 5130.dewater.critique.6 "dewater order is untraceable"
  IF
    NOT goal(find, _, 7218, "Remove Water", [compartment =
        Compartment], _, _)
    AND NOT goal(find, _, 7130, "Clear Water from Firefighting",
        [compartment = Compartment], _, _)
  THEN
    action(create, error-of-commission, 5130, "Dewater Space",
           [compartment = Compartment, station = Station], scenario-
        node, _)
  END RULE
END GMO

;*************************************************************

GMO 5140
FOR ECL 5140 "Desmoke Space"
  LET compartment -> Compartment,
      target -> Station

  RULE 5140.desmoke.critique.1 "preconditions for desmoking (flood
       crisis) are met"
  IF
    goal(find, unaddressed, 7318, "Remove Smoke", [compartment =
        Compartment], _, G)
    AND action(find, pending, 5140, "Desmoke Space", [compartment =
        Compartment], P, A)
    AND goal(find, satisfied, 7116, "Isolate Compartment if
        Necessary", [compartment = Compartment], _, _)
  THEN

    RULE 5140.desmoke.critique.1.1 "desmoke order is correct"
    IF
      world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = Station], _,
        _)
    THEN
      action(modify, correct, 5140, "Desmoke Space", [compartment =
        Compartment, station = Station], _, A)
      goal(modify, addressed, 7318, "Remove Smoke", [compartment =
        Compartment], _, G)
    END RULE

    RULE 5140.desmoke.critique.1.2 "desmoke order sub-optimal -
        wrong repair locker"
    IF
      NOT world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                      [compartment = Compartment, station = Station],
        _, _)
```

```
    AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
    action(modify, expired, 5140, "Desmoke Space", [compartment =
      Compartment], _, A)
    action(create, sub-optimal, 5140, "Desmoke Space",
      [compartment = Compartment, station = Station], P, _)
    goal(modify, addressed, 7318, "Remove Smoke", [compartment =
      Compartment], _, G)
  END RULE

  RULE 5140.desmoke.critique.1.3 "desmoke order incorrect - no
     personnel"
  IF
    NOT world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
    action(create, error-of-commission, 5140, "Desmoke Space",
           [compartment = Compartment, station = Station], P, _)
  END RULE
END RULE

RULE 5140.desmoke.critique.2 "preconditions for desmoking (after
     firefighting) are met"
IF
  goal(find, unaddressed, 7125, "Clear Smoke", [compartment =
     Compartment], _, G)
  AND action(find, pending, 5140, "Desmoke Space", [compartment =
     Compartment], P, A)
  AND goal(find, satisfied, 7118, "Apply Fire Suppressant",
     [compartment = Compartment], _, _)
THEN

  RULE 5140.desmoke.critique.3.0 "desmoke order is correct"
  IF
    world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                [compartment = Compartment, station = Station], _,
    _)
  THEN
    action(modify, correct, 5140, "Desmoke Space", [compartment =
      Compartment, station = Station], _, A)
    goal(modify, addressed, 7125, "Clear Smoke", [compartment =
      Compartment], _, G)
  END RULE

  RULE 5140.desmoke.critique.2.2 "desmoke order sub-optimal -
     wrong repair locker"
  IF
    NOT world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                    [compartment = Compartment, station = Station],
    _, _)
    AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
```

```
    action(modify, expired, 5140, "Desmoke Space", [compartment =
      Compartment], _, A)
    action(create, sub-optimal, 5140, "Desmoke Space",
      [compartment = Compartment, station = Station], P, _)
    goal(modify, addressed, 7125, "Clear Smoke", [compartment =
      Compartment], _, G)
  END RULE

  RULE 5140.desmoke.critique.2.3 "desmoke order incorrect - no
      personnel"
  IF
    NOT world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
  THEN
    action(create, error-of-commission, 5140, "Desmoke Space",
           [compartment = Compartment, station = Station], P, _)
  END RULE
END RULE

RULE 5140.desmoke.critique.3 "preconditions for desmoking (flood
     crisis) not met"
IF
  goal(find, unaddressed, 7318, "Remove Smoke", [compartment =
      Compartment], _, G)
  AND NOT goal(find, satisfied, 7116, "Isolate Compartment if
      Necessary", [compartment = Compartment], _, _)
THEN
  action(create, error-of-commission, 5140, "Desmoke Space",
         [compartment = Compartment, station = Station], G, _)
END RULE

RULE 5140.desmoke.critique.4 "preconditions for desmoking (after
     firefighting) not met"
IF
  goal(find, unaddressed, 7125, "Clear Smoke", [compartment =
      Compartment], _, G)
  AND NOT goal(find, satisfied, 7118, "Apply Fire Suppressant",
      [compartment = Compartment], _, _)
THEN
  action(create, error-of-commission, 5140, "Desmoke Space",
         [compartment = Compartment, station = Station], G, _)
END RULE

RULE 5140.desmoke.critique.5 "desmoke order incorrect -
     unnecessary"
IF
  NOT goal(find, unaddressed, 7318, "Remove Smoke", [compartment =
      Compartment], _, _)
  AND NOT goal(find, unaddressed, 7125, "Clear Smoke",
      [compartment = Compartment], _, _)
  AND EITHER
    goal(find, [addressed, satisfied], 7318, "Remove Smoke",
      [compartment = Compartment], _, G)
  OR
    goal(find, [addressed, satisfied], 7125, "Clear Smoke",
      [compartment = Compartment], _, G)
```

```
    END EITHER
  THEN
    action(create, error-of-commission, 5140, "Desmoke Space",
           [compartment = Compartment, station = Station], G, _)
  END RULE

  RULE 5140.desmoke.critique.6 "desmoke order is untraceable"
  IF
    NOT goal(find, _, 7318, "Remove Smoke", [compartment =
        Compartment], _, _)
    AND NOT goal(find, _, 7125, "Clear Smoke", [compartment =
        Compartment], _, _)
  THEN
    action(create, error-of-commission, 5140, "Desmoke Space",
           [compartment = Compartment, station = Station], scenario-
        node, _)
  END RULE
END GMO

;*****************************************************************

GMO 5150
FOR ECL 5150 "Open/Close Firemain Valve"
  LET valve -> Valve,
      valve_action -> ValveAction

  RULE 5150.valve.critique.1 "Open/Close firemain valve is correct"
  IF
    world-state(find, _, 4105, "Valve to Firemain Loop", [valve =
        Valve, loop = Loop], _, _)
    AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
        [loop = Loop], _, G)
    AND action(find, pending, 5150, "Open/Close Firemain Valve",
        [valve = Valve, valve_action = ValveAction], _, A)
  THEN
    action(modify, correct, 5150, "Open/Close Firemain Valve",
        [valve = Valve, valve_action = ValveAction], _,A)

    RULE 5150.valve.critique.1.1 "action satisfies restore firemain
        pressure goal"
    IF
      NOT world-state(find, _, 3901, "Goal Has Pending Actions",
        [goal = G], _, _)
    THEN
      goal(modify, addressed, 7560, "Restore Firemain Pressure",
        [loop = Loop], _, G)
   END RULE
  END RULE

  RULE 5150.valve.critique.2 "wrong valve action"
  IF
    world-state(find, _, 4105, "Valve to Firemain Loop", [valve =
        Valve, loop = Loop], _, _)
    AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
        [loop = Loop], _, G)
```

```
  AND NOT action(find, pending, 5150, "Open/Close Firemain
      Valve",[valve = Valve, valve_action = ValveAction],_,_)
  AND action(find, pending, 5150, "Open/Close Firemain Valve",
      [valve = Valve], P, A)
THEN
  action(create, error-of-commission, 5150, "Open/Close Firemain
      Valve", [valve = Valve], P, _)
END RULE

RULE 5150.valve.critique.3 "sub-optimal valve order"
IF
  world-state(find, _, 4105, "Valve to Firemain Loop", [valve =
      Valve, loop = Loop], _, _)
  AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
      [loop = Loop], _, G)
  AND NOT action(find, pending, 5150, "Open/Close Firemain
      Valve",[valve = Valve, valve_action = ValveAction],_,_)
  AND world-state(find, _, 3151, "Firemain Order is Beneficial",
      [action = ECLMessage], _, _)
THEN
  action(create, sub-optimal, 5150, "Open/Close Firemain Valve",
         [valve = Valve, valve_action = ValveAction], G, _)
END RULE

RULE 5150.valve.critique.4 "valve order unnecessary to solve goal"
IF
  world-state(find, _, 4105, "Valve to Firemain Loop", [valve =
      Valve, loop = Loop], _, _)
  AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
      [loop = Loop], _, G)
  AND NOT action(find, pending, 5150, "Open/Close Firemain Valve",
             [valve = Valve, valve_action = ValveAction], _, _)
  AND NOT world-state(find, _, 3151, "Firemain Order is
      Beneficial", [action = ECLMessage], _, _)
THEN
  action(create, error-of-commission, 5150, "Open/Close Firemain
      Valve",
         [valve = Valve, valve_action = ValveAction], G, _)
END RULE

RULE 5150.valve.critique.4 "valve order unnecessary"
IF
  world-state(find, _, 4105, "Valve to Firemain Loop", [valve =
      Valve, loop = Loop], _, _)
  AND NOT goal(find, unaddressed, 7560, "Restore Firemain
      Pressure", [loop = Loop], _, _)
  AND goal(find, [addressed, satisfied], 7560, "Restore Firemain
      Pressure", [loop = Loop], _, G)
THEN
  action(create, error-of-commission, 5150, "Open/Close Firemain
      Valve",
         [valve = Valve, valve_action = ValveAction], G, _)
END RULE

RULE 5150.valve.critique.5 "valve order is untraceable"
IF
```

```
    NOT goal(find, _, 7560, "Restore Firemain Pressure", [loop =
        Loop], _, _)
  THEN
    action(create, error-of-commission, 5150, "Open/Close Firemain
        Valve",
            [valve = Valve, valve_action = ValveAction], scenario-
        node, _)
  END RULE
END GMO

;*****************************************************************

GMO 5156.start
FOR ECL 5156 "Start/Stop Fire Pump"
WHERE pump_action = "start"
  LET pump -> Pump

  RULE 5156.start-pump.critique.1 "start pump addresses achieve
       minimal port pressure goal"
  IF
    world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "port"], _, _)
    AND world-state(find, _, 3105, "Fire Pump is Operational", [pump
        = Pump], _, _)
    AND goal(find, unaddressed, 7921, "Achieve minimal port-side
        firemain pressure", [], _, G)
  THEN
    goal(modify, addressed, 7921, "Achieve minimal port-side
        firemain pressure", [], _, G)

    RULE 5156.start-pump.critique.7 "permission has been granted;
        start pump order correct"
    IF
      world-state(find, _, 3109, "Permission to start a fire pump ",
        [], _, _)
      AND action(find, pending, 5156, "Start/Stop fire pump",
        [pump_action = "start"], G, A)
    THEN
      action(modify, correct, 5156, "Start/Stop Fire Pump",
        [pump_action = "start", pump = Pump], _, A)
    END RULE

    RULE 5156.start-pump.critique.8 "permission has not been
        granted; start pump order incorrect"
    IF
      NOT world-state(find, _, 3109, "Permission to start a fire
        pump ", [], _, _)
      AND action(find, pending, 5520, "Request permission to start
        fire pump", [], G, A)
    THEN
      action(modify, expired, 5520, "Request permission to start
        fire pump", [], G, A)
      action(create, error-of-commission, 5156, "Start/Stop Fire
        Pump",
             [pump_action = "start", pump = Pump], G, _)
    END RULE
```

```
END RULE

RULE 5156.start-pump.critique.2 "start pump addresses achieve
     minimal starboard pressure goal"
IF
  world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
     Pump, loop = "starboard"], _, _)
  AND world-state(find, _, 3105, "Fire Pump is Operational", [pump
     = Pump], _, _)
  AND goal(find, unaddressed, 7922, "Achieve minimal starboard
     firemain pressure", [], _, G)
THEN
  goal(modify, addressed, 7922, "Achieve minimal starboard
     firemain pressure", [], _, G)

  RULE 5156.start-pump.critique.9 "permission has been granted;
     start pump order correct"
  IF
    world-state(find, _, 3109, "Permission to start a fire pump ",
      [], _, _)
    AND action(find, pending, 5156, "Start/Stop fire pump",
      [pump_action = "start"], G, A)
  THEN
    action(modify, correct, 5156, "Start/Stop Fire Pump",
      [pump_action = "start", pump = Pump], _, A)
  END RULE

  RULE 5156.start-pump.critique.10 "permission has not been
     granted; start pump order incorrect"
  IF
    NOT world-state(find, _, 3109, "Permission to start a fire
      pump ", [], _, _)
    AND action(find, pending, 5520, "Request permission to start
      fire pump", [], G, A)
  THEN
    action(modify, expired, 5520, "Request permission to start
      fire pump", [], G, A)
    action(create, error-of-commission, 5156, "Start/Stop Fire
      Pump",
           [pump_action = "start", pump = Pump], G, _)
  END RULE
END RULE

RULE 5156.start-pump.critique.3 "start pump associated with
     restore firemain pressure goal"
IF
  world-state(find, _, 3105, "Fire Pump is Operational", [pump =
     Pump], _, _)
  AND world-state(find, _, 4108, "Get Equivalent Fire Pumps",
     [pump = Pump, pumps = Pumps], _, _)
  AND world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
     Pump, loop = Loop], _, _)
  AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
     [loop = Loop], _, G)
  AND action(find, pending, 5156, "Start/Stop fire pump",
     [pump_action = "start", pump <~ Pumps], G, A)
```

```
THEN
  goal(modify, addressed, 7560, "Restore Firemain Pressure", [loop
     = Loop], _, G)

  ; GRD - These rules seem redundant.  Rule 4 should handle them.
  ; RULE 5156.start-pump.interpret.1 "start pump addresses restore
     firemain pressure goal"
  ; RULE 5156.start-pump.critique.11 "permission has been granted
     or is not necessary; start pump order correct"
  ; RULE 5156.start-pump.critique.12

END RULE

RULE 5156.start-pump.critique.4 "start fire pump is sub-optimal
     for restoring firemain pressure"
IF
  world-state(find, _, 3105, "Fire Pump is Operational", [pump =
     Pump], _, _)
  AND world-state(find, _, 3103, "Fire Pump is Off", [pump =
     Pump], _, _)
  AND world-state(find, _, 4108, "Get Equivalent Fire Pumps",
     [pump = Pump, pumps = Pumps], _, _)
  AND world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
     Pump, loop = Loop], _, _)
  AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
     [loop = Loop], _, G)
  AND NOT action(find, pending, 5156, "Start/Stop fire pump",
     [pump_action = "start", pump <~ Pumps], G, A)
  AND world-state(find, _, 3151, "Firemain order is beneficial",
     [action = ECLMessage], _, _)
THEN
  CALL get-new-firemain-orders(gmo-name = "5520.request-pump",
     goal = G, loop = Loop)

  RULE 5156.start-pump.critique.11 "permission has been granted or
     is not necessary; start pump order correct"
  IF
    EITHER
      NOT world-state(find, _, 3302, "General Quarters", [], _, _)
    OR
      world-state(find, _, 3109, "Permission to start a fire pump
      ", [], _, _)
    END EITHER
  THEN
    action(create, sub-optimal, 5156, "Start/Stop Fire Pump",
      [pump_action = "start", pump = Pump], G, _)
  END RULE

  RULE 5156.start-pump.critique.12
       "permission has not been granted and is necessary; start
     pump order incorrect"
  IF
    world-state(find, _, 3302, "General Quarters", [], _, _)
    AND NOT world-state(find, _, 3109, "Permission to start a fire
      pump ", [], _, _)
  THEN
```

```
      action(create, error-of-commission, 5156, "Start/Stop Fire
        Pump",
            [pump_action = "start", pump = Pump], G, _)
    END RULE
  END RULE

  RULE 5156.start-pump.critique.5 "Start pump order inappropriate -
       pump is on or damaged"
  IF
    EITHER
      NOT world-state(find, _, 3105, "Fire Pump is Operational",
        [pump = Pump], _, _)
    OR
      world-state(find, _, 3104, "Fire Pump is On", [pump = Pump],
        _, _)
    END EITHER
  THEN
    action(create, error-of-commission, 5156, "Start/Stop Fire
       Pump", [pump = Pump, pump_action = "start"],
          miscellaneous-actions, _)
  END RULE

  RULE 5520.request-pump.critique.6 "Start pump order does not
       address existing goal"
  IF
    NOT world-state(find, _, 3151, "Firemain order is beneficial",
       [action = ECLMessage], _, _)
    AND EITHER
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "port"], _, _)
      AND NOT goal(find, unaddressed, 7921, "Achieve minimal port-
        side firemain pressure", [], _, G)
    OR
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "starboard"], _, _)
      AND NOT goal(find, unaddressed, 7922, "Achieve minimal
        starboard firemain pressure", [], _, G)
    END EITHER
  THEN
    action(create, error-of-commission, 5156, "Start/Stop Fire
       Pump", [pump = Pump, pump_action = "start"],
          miscellaneous-actions, _)
  END RULE
END GMO

;*****************************************************************

GMO 5156.stop
FOR ECL 5156 "Start/Stop Fire Pump"
WHERE pump_action = "stop"
  LET pump -> Pump

  RULE 5156.stop-pump.critique.1 "stop pump order benefits firemain"
  IF
    world-state(find, _, 3151, "Firemain order is beneficial",
       [action = ECLMessage], _, _)
```

```
  THEN
    action(create, correct, 5156, "Start/Stop Fire Pump", [pump =
        Pump, pump_action = "stop"],
            miscellaneous-actions, _)
  END RULE

  RULE 5156.stop-pump.critique.2 "stop pump order does not benefit
        firemain"
  IF
    NOT world-state(find, _, 3151, "Firemain order is beneficial",
        [action = ECLMessage], _, _)
  THEN
    action(create, error-of-commission, 5156, "Start/Stop Fire
        Pump", [pump = Pump, pump_action = "stop"],
            miscellaneous-actions, _)
  END RULE
END GMO

;*************************************************************

GMO 5162
FOR ECL 5162 "Isolate and Patch Firemain or Chillwater Rupture"
  LET target -> Station,
      compartment -> Compartment,
      valve1 -> Valve1,
     valve2 -> Valve2,
     [valve1, valve2] -> Valves

  RULE 5162.patch-pipe.critique.1 "isolate and patch is correct"
  IF
    goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
        = Compartment], _, G)
   AND action(find, pending, 5162, "Isolate and Patch Firemain or
        Chillwater Rupture",
           [compartment = Compartment, valve1 <~ Valves, valve2 <~
        Valves], _, A)
    AND world-state(find, _, 3121, "Pipe Should Be Isolated",
        [compartment = Compartment], _, _)
    AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                [compartment = Compartment, station = Station], _, _)
  THEN
   action(modify, correct, 5162, "Isolate and Patch Firemain or
        Chillwater Rupture",
          [target = Station, compartment = Compartment, valve1 <~
        Valves, valve2 <~ Valves], _, A)
    goal(modify, addressed, 7500, "Patch Pipe Rupture", [compartment
        = Compartment], _, G)
  END RULE

  RULE 5162.patch-pipe.critique.2 "wrong repair locker for isolate
        and patch - sub-optimal"
  IF
    goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
        = Compartment], _, G)
```

```
 AND action(find, pending, 5162, "Isolate and Patch Firemain or
     Chillwater Rupture",
        [compartment = Compartment, valve1 <~ Valves, valve2 <~
     Valves], P, A)
  AND world-state(find, _, 3121, "Pipe Should Be Isolated",
     [compartment = Compartment], _, _)
  AND NOT world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
                 [compartment = Compartment, station = Station],
     _, _)
  AND world-state(find, _, 3401, "Has Personnel", [station =
     Station], _, _)
THEN
 action(modify, expired, 5162, "Isolate and Patch Firemain or
     Chillwater Rupture",
       [compartment = Compartment, valve1 <~ Valves, valve2 <~
     Valves], _, A)
 action(create, sub-optimal, 5162, "Isolate and Patch Firemain or
     Chillwater Rupture",
        [target = Station, compartment = Compartment, valve1 <~
     Valves, valve2 <~ Valves], P, _)
  goal(modify, addressed, 7500, "Patch Pipe Rupture", [compartment
     = Compartment], _, G)
END RULE

RULE 5162.patch-pipe.critique.3 "wrong valves for isolate and
     patch - error of commission"
IF
  goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
     = Compartment], _, G)
  AND action(find, pending, 5162, "Isolate and Patch Firemain or
     Chillwater Rupture",
         [compartment = Compartment], P, A)
 AND action(find, pending, 5162, "Isolate and Patch Firemain or
     Chillwater Rupture",
        [compartment = Compartment, valve1 <~ Valves, valve2 <~
     Valves], _, _)
  AND world-state(find, _, 3121, "Pipe Should Be Isolated",
     [compartment = Compartment], _, _)
  AND world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
            [compartment = Compartment, station = Station], _, _)
THEN
  action(create, error-of-commission, 5162, "Isolate and Patch
     Firemain or Chillwater Rupture",
         [compartment = Compartment, valve1 = Valve1, valve2 =
     Valve2], _, _)
END RULE

RULE 5162.patch-pipe.critique.4 "Should have patched in place -
     error of commission"
IF
  goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
     = Compartment], _, G)
  AND action(find, pending, 5166, "Patch Firemain or Chillwater
     rupture in place",
```

```
            [compartment = Compartment], P, _)
      AND NOT world-state(find, _, 3121, "Pipe Should Be Isolated",
          [compartment = Compartment], _, _)
    THEN
      action(create, error-of-commission, 5162, "Isolate and Patch
          Firemain or Chillwater Rupture",
              [target = Station, compartment = Compartment, valve1 =
          Valve1, valve2 = Valve2], P, _)
    END RULE

    RULE 5162.patch-pipe.critique.5 "no personnel for isolate and
          patch - error of commission"
    IF
      goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
          = Compartment], _, G)
     AND action(find, pending, 5162, "Isolate and Patch Firemain or
          Chillwater Rupture",
             [compartment = Compartment, valve1 <~ Valves, valve2 <~
          Valves], P, _)
      AND NOT world-state(find, _, 3401, "Has Personnel", [station =
          Station], _, _)
    THEN
      action(create, error-of-commission, 5162, "Isolate and Patch
          Firemain or Chillwater Rupture",
              [target = Station, compartment = Compartment, valve1 =
          Valve1, valve2 = Valve2], P, _)
    END RULE

    RULE 5162.patch-pipe.critique.6 "isolate and patch unnecessary -
          error of commission"
    IF
      NOT goal(find, unaddressed, 7500, "Patch Pipe Rupture",
          [compartment = Compartment], _, _)
      AND goal(find, [addressed, satisfied], 7500, "Patch Pipe
          Rupture", [compartment = Compartment], _, G)
    THEN
      action(create, error-of-commission, 5162, "Isolate and Patch
          Firemain or Chillwater Rupture",
              [target = Station, compartment = Compartment, valve1 =
          Valve1, valve2 = Valve2], G, _)
    END RULE

    RULE 5162.patch-pipe.critique.7 "isolate and patch untraceable"
    IF
      NOT goal(find, _, 7500, "Patch Pipe Rupture", [compartment =
          Compartment], _, _)
    THEN
      action(create, error-of-commission, 5162, "Isolate and Patch
          Firemain or Chillwater Rupture",
              [target = Station, compartment = Compartment, valve1 =
          Valve1, valve2 = Valve2],
              scenario-node, _)
    END RULE
  END GMO

  ;*************************************************************
```

```
GMO 5166
FOR ECL 5166 "Patch Pipe in Place"
  LET target -> Station,
      compartment -> Compartment

  RULE 5166.patch-pipe.critique.1 "patch pipe in place is correct"
  IF
    goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
        = Compartment], _, G)
    AND action(find, pending, 5166, "Patch Pipe in Place ",
        [compartment = Compartment], _, A)
    AND NOT world-state(find, _, 3121, "Pipe Should Be Isolated",
        [compartment = Compartment], _, _)
    AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                [compartment = Compartment, station = Station], _, _)
  THEN
    action(modify, correct, 5166, "Patch Pipe in Place", [target =
        Station, compartment = Compartment], _, A)
    goal(modify, addressed, 7500, "Patch Pipe Rupture", [compartment
        = Compartment], _, G)
  END RULE

  RULE 5166.patch-pipe.critique.2 "should have isolated and patched
       - sub-optimal"
  IF
    goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
        = Compartment], _, G)
    AND action(find, pending, 5162, "Isolate and Patch Firemain or
        Chillwater Rupture",
            [compartment = Compartment], P, A)
    AND world-state(find, _, 3121, "Pipe Should Be Isolated",
        [compartment = Compartment], _, _)
    AND world-state(find, _, 3401, "Has Personnel", [station =
        Station], _, _)
  THEN
    action(modify, expired, 5162, "Isolate and Patch Firemain or
        Chillwater Rupture", [compartment = Compartment], _, A)
    action(create, sub-optimal, 5166, "Patch Pipe in Place",
            [target = Station, compartment = Compartment], P, _)
    goal(modify, addressed, 7500, "Patch Pipe Rupture", [compartment
        = Compartment], _, G)
  END RULE

  RULE 5166.patch-pipe.critique.3 "wrong repair locker for patch in
       place - sub-optimal"
  IF
    goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
        = Compartment], _, G)
    AND action(find, pending, 5166, "Patch Pipe in Place ",
        [compartment = Compartment], P, A)
    AND NOT world-state(find, _, 3121, "Pipe Should Be Isolated",
        [compartment = Compartment], _, _)
    AND NOT world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
```

```
                    [compartment = Compartment, station = Station],
        _, _)
      AND world-state(find, _, 3401, "Has Personnel", [station =
          Station], _, _)
    THEN
      action(modify, expired, 5166, "Patch Pipe in Place ",
          [compartment = Compartment], _, A)
      action(create, sub-optimal, 5166, "Patch Pipe in Place",
            [target = Station, compartment = Compartment], P, _)
      goal(modify, addressed, 7500, "Patch Pipe Rupture", [compartment
          = Compartment], _, G)
    END RULE

    RULE 5166.patch-pipe.critique.4 "no personnel for patch in place -
         error of commission"
    IF
      goal(find, unaddressed, 7500, "Patch Pipe Rupture", [compartment
          = Compartment], _, G)
      AND NOT world-state(find, _, 3401, "Has Personnel", [station =
          Station], _, _)
    THEN
      action(create, error-of-commission, 5166, "Patch Pipe in Place",
            [target = Station, compartment = Compartment], G, _)
    END RULE

    RULE 5166.patch-pipe.critique.5 "patch in place unnecessary -
         error of commission"
    IF
      NOT goal(find, unaddressed, 7500, "Patch Pipe Rupture",
          [compartment = Compartment], _, _)
      AND goal(find, [addressed, satisfied], 7500, "Patch Pipe
          Rupture", [compartment = Compartment], _, G)
    THEN
      action(create, error-of-commission, 5166, "Patch Pipe in Place",
            [target = Station, compartment = Compartment], G, _)
    END RULE

    RULE 5166.patch-pipe.critique.6 "patch in place untraceable"
    IF
      NOT goal(find, _, 7500, "Patch Pipe Rupture", [compartment =
          Compartment], _, _)
    THEN
      action(create, error-of-commission, 5166, "Patch Pipe in Place",
            [target = Station, compartment = Compartment], scenario-
          node, _)
    END RULE
  END GMO

  ;***************************************************************

  GMO 5170
  FOR ECL 5170 "Electrically and Mechanically Isolate Space"
    LET compartment -> Compartment,
        target -> Station
```

```
RULE 5170.isolate.critique.5 "isolate space is unnecessary - error
     of commission"
IF
  NOT goal(find, unaddressed, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, _)
  AND goal(find, [addressed, satisfied], 7116, "Isolate
     Compartment if Necessary",
       [compartment = Compartment], _, G)
THEN
  action(create, error-of-commission, 5170, "Electrically and
     Mechanically Isolate Space",
        [target = Station, compartment = Compartment], G, _)
END RULE

RULE 5170.isolate.critique.1 "isolate space order is correct"
IF
  goal(find, unaddressed, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, G)
  AND action(find, pending, 5170, "Electrically and Mechanically
     Isolate Space",
        [compartment = Compartment], _, A)
  AND world-state(find, _, 3360, "Can Isolate Compartment",
     [compartment = Compartment], _, _)
  AND world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
            [compartment = Compartment, station = Station], _, _)
THEN
  goal(modify, addressed, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, G)
  action(modify, correct, 5170, "Electrically and Mechanically
     Isolate Space",
        [target = Station, compartment = Compartment], _, A)
END RULE

RULE 5170.isolate.critique.2 "wrong repair locker for isolate
     space - sub-optimal action"
IF
  goal(find, unaddressed, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, G)
  AND action(find, pending, 5170, "Electrically and Mechanically
     Isolate Space",
        [compartment = Compartment], P, A)
  AND world-state(find, _, 3360, "Can Isolate Compartment",
     [compartment = Compartment], _, _)
  AND NOT world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
               [compartment = Compartment, station = Station],
     _, _)
  AND world-state(find, _, 3401, "Has Personnel", [station =
     Station], _, _)
THEN
  goal(modify, addressed, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, G)
  action(modify, expired, 5170, "Electrically and Mechanically
     Isolate Space",
        [compartment = Compartment], _, A)
```

```
    action(create, sub-optimal, 5170, "Electrically and Mechanically
        Isolate Space",
            [target = Station, compartment = Compartment], P, _)
  END RULE

  RULE 5170.isolate.critique.3 "no personnel for isolate space -
        error of commission"
  IF
    goal(find, unaddressed, 7116, "Isolate Compartment if
        Necessary", [compartment = Compartment], _, G)
    AND action(find, pending, 5170, "Electrically and Mechanically
        Isolate Space",
            [compartment = Compartment], P, A)
    AND world-state(find, _, 3360, "Can Isolate Compartment",
        [compartment = Compartment], _, _)
    AND NOT world-state(find, _, 3401, "Has Personnel", [station =
        Station], _, _)
  THEN
    action(create, error-of-commission, 5170, "Electrically and
        Mechanically Isolate Space",
            [target = Station, compartment = Compartment], P, _)
  END RULE

  RULE 5170.isolate.critique.4 "no need to isolate - error of
        commission"
  IF
    goal(find, _, 7116, "Isolate Compartment if Necessary",
        [compartment = Compartment], _, G)
    AND NOT world-state(find, _, 3360, "Can Isolate Compartment",
        [compartment = Compartment], _, _)
  THEN
    action(create, error-of-commission, 5170, "Electrically and
        Mechanically Isolate Space",
            [target = Station, compartment = Compartment], G, _)
  END RULE

  RULE 5170.isolate.critique.6 "isolate space order is untraceable"
  IF
    NOT goal(find, _, 7116, "Isolate Compartment if Necessary",
        [compartment = Compartment], _, _)
  THEN
    action(create, error-of-commission, 5170, "Electrically and
        Mechanically Isolate Space",
            [target = Station, compartment = Compartment], scenario-
        node, _)
  END RULE
END GMO

;*****************************************************************

GMO 5176
FOR ECL 5176 "Patch Hull Rupture"
  LET target -> Station,
      compartment -> Compartment

  RULE 5176.patch-hull.critique.1 "Patch Hull is correct"
```

```
IF
  goal(find, unaddressed, 7400, "Patch Hull Rupture", [compartment
      = Compartment], _, G)
  AND action(find, pending, 5176, "Patch Hull ", [compartment =
      Compartment], _, A)
  AND world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
              [compartment = Compartment, station = Station], _, _)
THEN
  action(modify, correct, 5176, "Patch Hull", [target = Station,
      compartment = Compartment], _, A)
  goal(modify, addressed, 7400, "Patch Hull Rupture", [compartment
      = Compartment], _, G)
END RULE

RULE 5176.patch-hull.critique.2 "wrong repair locker for patch
     hull - sub-optimal"
IF
  goal(find, unaddressed, 7400, "Patch Hull Rupture", [compartment
      = Compartment], _, G)
  AND action(find, pending, 5176, "Patch Hull ", [compartment =
      Compartment], P, A)
  AND NOT world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                  [compartment = Compartment, station = Station],
      _, _)
  AND world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
THEN
  action(modify, expired, 5176, "Patch Hull ", [compartment =
      Compartment], _, A)
  action(create, sub-optimal, 5176, "Patch Hull", [target =
      Station, compartment = Compartment], P, _)
  goal(modify, addressed, 7400, "Patch Hull Rupture", [compartment
      = Compartment], _, G)
END RULE

RULE 5176.patch-hull.critique.3 "no personnel for patch in place -
     error of commission"
IF
  goal(find, unaddressed, 7400, "Patch Hull Rupture", [compartment
      = Compartment], _, G)
  AND NOT world-state(find, _, 3401, "Has Personnel", [station =
      Station], _, _)
THEN
  action(create, error-of-commission, 5176, "Patch Hull", [target
      = Station, compartment = Compartment], G, _)
END RULE

RULE 5176.patch-hull.critique.5 "patch in place unnecessary -
     error of commission"
IF
  NOT goal(find, unaddressed, 7400, "Patch Hull Rupture",
      [compartment = Compartment], _, _)
  AND goal(find, [addressed, satisfied], 7400, "Patch Hull
      Rupture", [compartment = Compartment], _, G)
```

```
  THEN
    action(create, error-of-commission, 5176, "Patch Hull", [target
        = Station, compartment = Compartment], G, _)
  END RULE

  RULE 5176.patch-hull.critique.6 "patch in place untraceable"
  IF
    NOT goal(find, _, 7400, "Patch Hull Rupture", [compartment =
        Compartment], _, _)
  THEN
    action(create, error-of-commission, 5176, "Patch Hull",
           [target = Station, compartment = Compartment], scenario-
        node, _)
  END RULE
END GMO

;*************************************************************

GMO 5510
FOR ECL 5510 "Request Permission to Flood Magazine"
  LET compartment -> Compartment

  RULE 5510.request-flood.critique.1 "request to flood is correct"
  IF
    goal(find, unaddressed, 7155, "Try to Get Permission to Flood
        Magazines", [compartment = Compartment], _, G)
    AND action(find, pending, 5510, "Request Permission to Flood
        Magazine", [compartment = Compartment], _, A)
  THEN
    goal(modify, addressed, 7155, "Try to Get Permission to Flood
        Magazines", [compartment = Compartment], _, G)
    action(modify, correct, 5510, "Request Permission to Flood
        Magazine", [compartment = Compartment], _, A)
  END RULE

  RULE 5510.request-flood.critique.2 "request to flood is
       unnecessary - error of commission"
  IF
    NOT goal(find, unaddressed, 7155, "Try to Get Permission to
        Flood Magazines",
             [compartment = Compartment], _, _)
    AND goal(find, [addressed, satisfied], 7155, "Try to Get
        Permission to Flood Magazines",
          [compartment = Compartment], _, G)
  THEN
    action(create, error-of-commission, 5510, "Request Permission to
        Flood Magazine",
           [compartment = Compartment], G, _)
  END RULE

  RULE 5510.request-flood.critique.3 "request to flood is
       untraceable"
  IF
    NOT goal(find, _, 7155, "Try to Get Permission to Flood
        Magazines", [compartment = Compartment], _, _)
  THEN
```

```
    action(create, error-of-commission, 5510, "Request Permission to
        Flood Magazine",
            [compartment = Compartment], scenario-node, _)
  END RULE
END GMO

;*************************************************************

GMO 5520
FOR ECL 5520 "Request Permission to Start Fire Pump"
  LET pump -> Pump

  RULE 5520.request-pump.critique.1
       "request permission to start pump addresses achieve minimal
       port pressure goal"
  IF
    world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "port"], _, _)
    AND world-state(find, _, 3105, "Fire Pump is Operational", [pump
        = Pump], _, _)
    AND goal(find, unaddressed, 7921, "Achieve minimal port-side
        firemain pressure", [], _, G)
    AND action(find, pending, 5520, "Request Permission to Start
        Fire Pump", [], G, A)
  THEN
    action(modify, correct, 5520, "Request Permission to Start Fire
        Pump", [pump = Pump], _, A)
    goal(modify, addressed, 7921, "Achieve minimal port-side
        firemain pressure", [], _, G)
  END RULE

  RULE 5520.request-pump.critique.2
       "request permission to start pump addresses achieve minimal
       starboard pressure goal"
  IF
    world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "starboard"], _, _)
    AND world-state(find, _, 3105, "Fire Pump is Operational", [pump
        = Pump], _, _)
    AND goal(find, unaddressed, 7922, "Achieve minimal starboard
        firemain pressure", [], _, G)
    AND action(find, pending, 5520, "Request Permission to Start
        Fire Pump", [], G, A)
  THEN
    action(modify, correct, 5520, "Request Permission to Start Fire
        Pump", [pump = Pump], _, A)
    goal(modify, addressed, 7922, "Achieve minimal starboard
        firemain pressure", [], _, G)
  END RULE

  RULE 5520.request-pump.critique.3
       "request permission to start pump associated with restore
       firemain pressure goal"
  IF
    world-state(find, _, 3105, "Fire Pump is Operational", [pump =
        Pump], _, _)
```

```
  AND world-state(find, _, 4108, "Get Equivalent Fire Pumps",
      [pump = Pump, pumps = Pumps], _, _)
  AND world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
      Pump, loop = Loop], _, _)
  AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
      [loop = Loop], _, G)
  AND action(find, pending, 5520, "Request Permission to Start
      Fire Pump", [pump <~ Pumps], G, A)
THEN
  action(modify, correct, 5520, "Request Permission to Start Fire
      Pump", [pump = Pump], _, A)

  RULE 5520.request-pump.interpret.1
       "request permission to start pump addresses restore
      firemain pressure goal"
  IF
    NOT world-state(find, _, 3901, "Goal has pending actions",
      [goal = G], _, _)
  THEN
    goal(modify, addressed, 7560, "Restore Firemain Pressure",
      [loop = Loop], _, G)
  END RULE
END RULE

RULE 5520.request-pump.critique.4 "start fire pump request is sub-
      optimal for restoring firemain pressure"
IF
  world-state(find, _, 3105, "Fire Pump is Operational", [pump =
      Pump], _, _)
  AND world-state(find, _, 3103, "Fire Pump is Off", [pump =
      Pump], _, _)
  AND world-state(find, _, 4108, "Get Equivalent Fire Pumps",
      [pump = Pump, pumps = Pumps], _, _)
  AND world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
      Pump, loop = Loop], _, _)
  AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
      [loop = Loop], _, G)
  AND NOT action(find, pending, 5520, "Request Permission to Start
      Fire Pump", [pump <~ Pumps], G, _)
  AND world-state(find, _, 3151, "Firemain order is beneficial",
      [action = ECLMessage], _, _)
THEN
  action(create, sub-optimal, 5520, "Request Permission to Start
      Fire Pump", [pump = Pump], _, _)
  CALL get-new-firemain-orders(gmo-name = "5520.request-pump",
      goal = G, loop = Loop)
END RULE

RULE 5520.request-pump.critique.5 "Start pump order inappropriate
      - pump is on or damaged"
IF
  EITHER
    NOT world-state(find, _, 3105, "Fire Pump is Operational",
      [pump = Pump], _, _)
  OR
```

```
      world-state(find, _, 3104, "Fire Pump is On", [pump = Pump],
        _, _)
    END EITHER
  THEN
    action(create, error-of-commission, 5520, "Request Permission to
        Start Fire Pump", [pump = Pump],
           miscellaneous-actions, _)
  END RULE

  RULE 5520.request-pump.critique.6 "Start pump order does not
       address existing goal"
  IF
    NOT world-state(find, _, 3151, "Firemain order is beneficial",
        [action = ECLMessage], _, _)
    AND EITHER
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "port"], _, _)
      AND NOT goal(find, unaddressed, 7921, "Achieve minimal port-
        side firemain pressure", [], _, G)
    OR
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "starboard"], _, _)
      AND NOT goal(find, unaddressed, 7922, "Achieve minimal
        starboard firemain pressure", [], _, G)
    END EITHER
  THEN
    action(create, error-of-commission, 5520, "Request Permission to
        Start Fire Pump", [pump = Pump],
           miscellaneous-actions, _)
  END RULE
END GMO

;*****************************************************************

GMO 5601
FOR ECL 5601 "Report MRZ Set Shipwide"

  RULE 5601.mrz.critique.1 "MRZ set report is correct"
  IF
    goal(find, unaddressed, 7905, "Monitor Shipwide MRZ Status", [],
        _, G)
    AND action(find, pending, 5601, "Report MRZ Set Shipwide", [],
        _, A)
    AND world-state(find, _, 3201, "All Stations Manned Ready
        Zebra", [], _, _)
  THEN
    action(modify, correct, 5601, "Report MRZ Set Shipwide", [], _,
        A)
    goal(modify, satisfied, 7905, "Monitor Shipwide MRZ Status", [],
        _, G)
  END RULE

  RULE 5601.mrz.critique.2 "MRZ set report is correct but
       unnecessary"
  IF
```

```
    goal(find, [addressed, satisfied], 7905, "Monitor Shipwide MRZ
        Status", [], _, G)
    AND world-state(find, _, 3201, "All Stations Manned Ready
        Zebra", [], _, _)
  THEN
    action(create, correct, 5601, "Report MRZ Set Shipwide", [], G,
        _)
  END RULE

  RULE 5601.mrz.critique.3 "MRZ set report incorrect (no GQ)"
  IF
    NOT world-state(find, _, 3202, "General Quarters", [], _, _)
  THEN
    action(create, error-of-commission, 5601, "Report MRZ Set
        Shipwide", [], scenario-node, _)
  END RULE

  RULE 5601.mrz.critique.4 "MRZ set report incorrect (not all
       stations MRZ)"
  IF
    goal(find, unaddressed, 7905, "Monitor Shipwide MRZ Status", [],
        _, G)
    AND NOT world-state(find, _, 3201, "All Stations Manned Ready
        Zebra", [], _, _)
  THEN
    action(create, error-of-commission, 5601, "Report MRZ Set
        Shipwide", [], G, _)
  END RULE
END GMO

;*************************************************************

GMO 5602
FOR ECL 5602 "Report MRZ In-Progress Shipwide"

  RULE 5602.mrz.critique.1 "MRZ in-progress report is correct"
  IF
    goal(find, unaddressed, 7915, "Respond to CO Requests for
        Shipwide MRZ Status", [], _, G)
    AND action(find, pending, 5602, "Report MRZ In-Progress
        Shipwide", [], _, A)
    AND NOT world-state(find, _, 3201, "All Stations Manned Ready
        Zebra", [], _, _)
  THEN
    goal(modify, addressed, 7915, "Respond to CO Requests for
        Shipwide MRZ Status", [], _, G)
    action(modify, correct, 5602, "Report MRZ In-Progress Shipwide",
        [], _, A)
  END RULE

  RULE 5602.mrz.critique.2 "MRZ in-progress report is correct but
       unnecessary"
  IF
    goal(find, [addressed, satisfied], 7915, "Respond to CO Requests
        for Shipwide MRZ Status", [], _, G)
```

```
    AND NOT world-state(find, _, 3201, "All Stations Manned Ready
        Zebra", [], _, _)
  THEN
    action(create, correct, 5602, "Report MRZ In-Progress Shipwide",
        [], G, _)
  END RULE

  RULE 5602.mrz.critique.3 "MRZ in-progress report incorrect (no
       GQ)"
  IF
    NOT world-state(find, _, 3202, "General Quarters", [], _, _)
  THEN
    action(create, error-of-commission, 5602, "Report MRZ In-
        Progress Shipwide", [], scenario-node, _)
  END RULE

  RULE 5602.mrz.critique.4 "MRZ in-progress report incorrect (MRZ is
       set)"
  IF
    goal(find, _, 7915, "Respond to CO Requests for Shipwide MRZ
        Status", [], _, G)
    AND world-state(find, _, 3201, "All Stations Manned Ready
        Zebra", [], _, _)
  THEN
    action(create, error-of-commission, 5602, "Report MRZ In-
        Progress Shipwide", [], G, _)
  END RULE
END GMO
```

E.3. GMOs for 6000 Level ECL Reports (Agent to DCA Messages)

```
GMO 6105
FOR ECL 6105 "Acknowledge Order to Investigate Space"
LET compartment -> Compartment,
    source -> Station

  RULE 6105.investigate.interpret.1 "valid acknowledge investigate"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5105,
        "Investigate Space",
            [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6105, "Acknowledge Order to Investigate
        Compartment",
            [source = Station, compartment = Compartment], A, _)
  END RULE

  RULE 6105.investigate.interpret.2 "invalid acknowledge
        investigate"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
        5105, "Investigate Space",
            [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6105, "Acknowledge Order to Investigate
        Compartment",
            [source = Station, compartment = Compartment],
        miscellaneous-reports, _)
  END RULE
END GMO

GMO 6107
FOR ECL 6107 "Acknowledge Order to Investigate Firemain or
        Chillwater"
LET loop -> Loop,
    source -> Station

  RULE 6107.investigate.interpret.1 "valid acknowledge investigate"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5107,
        "Investigate Firemain or Chillwater",
            [loop = Loop, target = Station], _, A)
  THEN
    report(create, _, 6107, "Acknowledge Order to Investigate
        Firemain or Chillwater",
            [source = Station, loop = Loop], A, _)
  END RULE

  RULE 6107.investigate.interpret.2 "invalid acknowledge
        investigate"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
        5107, "Investigate Firemain or Chillwater",
```

```
            [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6107, "Acknowledge Order to Investigate
        Firemain or Chillwater",
            [source = Station, loop = Loop], miscellaneous-reports,
        _)
  END RULE
END GMO

GMO 6108
FOR ECL 6108 "Acknowledge order to Investigate Firemain or
       Chillwater in space"
LET source -> Station,
    compartment -> Compartment,
    loop -> Loop

  RULE 6108.Investigate-Firemain.interpret.1 "valid acknowledge
       Investigate-Firemain"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5108,
   "Investigate Firemain Or Chillwater In Space",
      [compartment = Compartment, loop = Loop, target = Station], _,
       A)
  THEN
    report(create, _, 6108, "Acknowledge order to Investigate
       Firemain or Chillwater in space",
      [compartment = Compartment, loop = Loop, source = Station], A,
       _)
  END RULE

  RULE 6108.Investigate-Firemain.interpret.2 "invalid acknowledge
       Investigate-Firemain"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5108,
   "Investigate Firemain Or Chillwater In Space",
      [compartment = Compartment, loop = Loop, target = Station], _,
       _)
  THEN
    report(create, _, 6108, "Acknowledge order to Investigate
       Firemain or Chillwater in space",
      [compartment = Compartment, loop = Loop, source = Station],
       untraceable-reports, _)
  END RULE
END GMO

GMO 6110
FOR ECL 6110 "Acknowledge Order to Set Boundaries Fire/Flood/Smoke"
LET casualty -> Casualty,
    source -> Station,
    saft -> SAft,
    paft -> PAft,
    pfor -> PFor,
    sfor -> SFor,
    above -> Above,
    below -> Below
```

```
  RULE 6110.set-boundaries.interpret.1 "valid acknowledge set-
       boundaries"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5110,
       "Set Boundaries Fire/Flood/Smoke",
          [casualty = Casualty, target = Station, saft = SAft, paft
       = PAft, pfor = PFor, sfor = SFor, above = Above, below =
       Below], _, A)
  THEN
    report(create, _, 6110, "Acknowledge Order to Set Boundaries
       Fire/Flood/Smoke",
          [casualty = Casualty, source = Station, saft = SAft, paft
       = PAft, pfor = PFor, sfor = SFor, above = Above, below =
       Below], A, _)
  END RULE

  RULE 6110.set-boundaries.interpret.2 "invalid acknowledge set-
       boundaries"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5110, "Set Boundaries Fire/Flood/Smoke",
          [casualty = Casualty, target = Station, saft = SAft, paft
       = PAft, pfor = PFor, sfor = SFor, above = Above, below =
       Below], _, _)
  THEN
    report(create, _, 6110, "Acknowledge Order to Set Boundaries
       Fire/Flood/Smoke",
          [casualty = Casualty, source = Station, saft = SAft, paft
       = PAft, pfor = PFor, sfor = SFor, above = Above, below =
       Below],
          miscellaneous-reports, _)
  END RULE
END GMO

GMO 6114
FOR ECL 6114 "Acknowledge Order to Set Boundaries on Compartment
       Fire/Flood/Smoke"
LET source -> Station,
  compartment -> Compartment,
  casualty -> Casualty

  RULE 6114.set-boundaries.interpret.1 "valid acknowledge set-
       boundaries"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5114,
          "Set Boundaries on Compartment Fire/Flood/Smoke",
          [casualty = Casualty, target = Station, compartment =
       Compartment], _, A)
  THEN
    report(create, _, 6114, "Acknowledge Order to Set Boundaries on
       Compartment Fire/Flood/Smoke",
          [source = Station, compartment = Compartment], A, _)
  END RULE
```

```
  RULE 6114.set-boundaries.interpret.2 "invalid acknowledge set-
       boundaries"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5114,
           "Set Boundaries on Compartment Fire/Flood/Smoke",
           [casualty = Casualty, target = Station, compartment =
       Compartment], _, A)
  THEN
    report(create, _, 6114, "Acknowledge Order to Set Boundaries on
       Compartment Fire/Flood/Smoke",
           [source = Station, compartment = Compartment],
       miscellaneous-reports, _)
  END RULE
END GMO

GMO 6120
FOR ECL 6120 "Acknowledge Order to Fight Fire in Space"
LET source -> Station,
    compartment -> Compartment

  RULE 6120.fight-fire.interpret.1 "valid acknowledge fight fire"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5120,
           "Fight Fire in Space", [target = Station, compartment =
       Compartment], _, A)
  THEN
    report(create, _, 6120, "Acknowledge Order to Fight Fire in
       Space",
           [source = Station, compartment = Compartment], A, _)
  END RULE

  RULE 6114.set-boundaries.interpret.2 "invalid acknowledge fight
       fire"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5120,
           "Fight Fire in Space", [target = Station, compartment =
       Compartment], _, _)
  THEN
    report(create, _, 6120, "Acknowledge Order to Fight Fire in
       Space",
           [source = Station, compartment = Compartment],
       miscellaneous-reports, _)
  END RULE
END GMO

GMO 6130
FOR ECL 6130 "Acknowledge order to dewater space"
LET source -> Station,
    compartment -> Compartment

  RULE 6130.dewater-space.interpret.1 "valid acknowledge dewater-
       space"
  IF
```

```
    action(find, [correct, sub-optimal, error-of-comission], 5130,
        "Dewater Space",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6130, "Acknowledge order to dewater space",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6130.dewater-space.interpret.2 "invalid acknowledge dewater-
       space"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5130, "Dewater Space",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6130, "Acknowledge order to dewater space",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

GMO 6140
FOR ECL 6140 "Acknowledge order to desmoke space"
LET source -> Station,
    compartment -> Compartment

  RULE 6140.desmoke-space.interpret.1 "valid acknowledge desmoke-
       space"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5140,
        "Desmoke Space",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6140, "Acknowledge order to desmoke space",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6140.desmoke-space.interpret.2 "invalid acknowledge desmoke-
       space"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5140, "Desmoke Space",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6140, "Acknowledge order to desmoke space",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

GMO 6142
FOR ECL 6142 "Acknowledge order to active desmoke space"
LET source -> Station,
    compartment -> Compartment
```

```
  RULE 6142.active-desmoke.interpret.1 "valid acknowledge active-
       desmoke"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5142,
        "Active Desmoke Space",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6142, "Acknowledge order to active desmoke
        space",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6142.active-desmoke.interpret.2 "invalid acknowledge active-
       desmoke"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5142, "Active Desmoke Space",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6142, "Acknowledge order to active desmoke
        space",
      [compartment = Compartment, source = Station], untraceable-
        reports, _)
  END RULE
END GMO

GMO 6144
FOR ECL 6144 "Acknowledge order to Desmoke space using
       overpressurization"
LET source -> Station,
    compartment -> Compartment

  RULE 6144.Desmoke-space.interpret.1 "valid acknowledge Desmoke-
       space"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5144,
        "Desmoke Space Using Overpressurization",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6144, "Acknowledge order to Desmoke space
        using overpressurization",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6144.Desmoke-space.interpret.2 "invalid acknowledge Desmoke-
       space"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5144, "Desmoke Space Using Overpressurization",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6144, "Acknowledge order to Desmoke space
        using overpressurization",
      [compartment = Compartment, source = Station], untraceable-
        reports, _)
  END RULE
```

```
END GMO

GMO 6150
FOR ECL 6150 "Acknowledge order to open/close Firemain valve"
LET source -> Station,
    valve -> Valve,
    valve_action -> ValveAction

  RULE 6150.valve.interpret.1 "valid acknowledge open/close
       firemain"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5150,
        "Open/close Firemain Valve",
      [valve = Valve, valve_action = ValveAction, target = Station],
       _, A)
  THEN
    report(create, _, 6150, "Acknowledge order to open/close
        Firemain valve",
      [valve = Valve, valve_action = ValveAction, source = Station],
       A, _)
  END RULE

  RULE 6150.valve.interpret.2 "invalid acknowledge open/close
       firemain"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5150, "Open/close Firemain Valve",
      [valve = Valve, valve_action = ValveAction, target = Station],
       _, _)
  THEN
    report(create, _, 6150, "Acknowledge order to open/close
        Firemain valve",
      [valve = Valve, valve_action = ValveAction, source = Station],
       untraceable-reports, _)
  END RULE
END GMO

GMO 6156
FOR ECL 6156 "Acknowledge order to start/stop fire pump"
LET source -> Station,
    pump -> Pump,
    pump_action -> PumpAction

  RULE 6156.toggle-firepump.interpret.1 "valid acknowledge
       start/stop fire pump"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5156,
        "Start/stop Fire Pump",
      [pump = Pump, pump_action = PumpAction, target = Station], _,
       A)
  THEN
    report(create, _, 6156, "Acknowledge order to start/stop fire
        pump",
      [pump = Pump, pump_action = PumpAction, source = Station], A,
       _)
  END RULE
```

```
  RULE 6156.toggle-firepump.interpret.2 "invalid acknowledge
       start/stop fire pump"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5156, "Start/stop Fire Pump",
      [pump = Pump, pump_action = PumpAction, target = Station], _,
       _)
  THEN
    report(create, _, 6156, "Acknowledge order to start/stop fire
       pump",
      [pump = Pump, pump_action = PumpAction, source = Station],
       untraceable-reports, _)
  END RULE
END GMO

GMO 6162
FOR ECL 6162 "Acknowledge order to Isolate and patch Firemain or
       Chillwater rupture"
LET source -> Station,
    compartment -> Compartment,
    loop -> Loop,
    valve -> Valves

  RULE 6162.Isolate-and-patch.interpret.1 "valid acknowledge
       Isolate-and-patch "
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5162,
   "Isolate And Patch Firemain Or Chillwater Rupture",
      [compartment = Compartment, loop = Loop, valve = Valves,
       target = Station], _, A)
  THEN
    report(create, _, 6162, "Acknowledge order to Isolate and patch
       Firemain or Chillwater rupture",
      [compartment = Compartment, loop = Loop, valve = Valves,
       source = Station], A, _)
  END RULE

  RULE 6162. Isolate-and-patch.interpret.2 "invalid acknowledge
       Isolate-and-patch "
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5162,
   "Isolate And Patch Firemain Or Chillwater Rupture",
      [compartment = Compartment, loop = Loop, valve = Valves,
       target = Station], _, _)
  THEN
    report(create, _, 6162, "Acknowledge order to Isolate and patch
       Firemain or Chillwater rupture",
      [compartment = Compartment, loop = Loop, valve = Valves,
       source = Station], untraceable-reports, _)
  END RULE
END GMO

GMO 6166
```

```
FOR ECL 6166 "Acknowledge order to patch Firemain or Chillwater
       rupture in place"
LET source -> Station,
    compartment -> Compartment,
    loop -> Loop

  RULE 6166.patch-Firemain.interpret.1 "valid acknowledge patch-
       Firemain"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5166,
   "Patch Firemain Or Chillwater Rupture In Place",
   [compartment = Compartment, loop = Loop, target = Station], _, A)
  THEN
    report(create, _, 6166, "Acknowledge order to patch Firemain or
       Chillwater rupture in place",
      [compartment = Compartment, loop = Loop, source = Station], A,
       _)
  END RULE

  RULE 6166.patch-Firemain.interpret.2 "invalid acknowledge patch-
       Firemain"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5166,
   "Patch Firemain Or Chillwater Rupture In Place",
      [compartment = Compartment, loop = Loop, target = Station], _,
       _)
  THEN
    report(create, _, 6166, "Acknowledge order to patch Firemain or
       Chillwater rupture in place",
      [compartment = Compartment, loop = Loop, source = Station],
       untraceable-reports, _)
  END RULE
END GMO

GMO 6170
FOR ECL 6170 "Acknowledge order to mechanically isolate space"
LET source -> Station,
    compartment -> Compartment

  RULE 6170.mechanically-isolate.interpret.1 "valid acknowledge
       mechanically-isolate"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5170,
       "Mechanically Isolate Space",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6170, "Acknowledge order to mechanically
       isolate space",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6170.mechanically-isolate.interpret.2 "invalid acknowledge
       mechanically-isolate"
  IF
```

```
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5170, "Mechanically Isolate Space",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6170, "Acknowledge order to mechanically
        isolate space",
      [compartment = Compartment, source = Station], untraceable-
        reports, _)
  END RULE
END GMO

GMO 6171
FOR ECL 6171 "Acknowledge order to Electrically isolate space"
LET source -> Station,
    compartment -> Compartment

  RULE 6171.Electrically-isolate.interpret.1 "valid acknowledge
       Electrically-isolate"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5171,
        "Electrically Isolate Space",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6171, "Acknowledge order to Electrically
        isolate space",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6171.Electrically-isolate.interpret.2 "invalid acknowledge
       Electrically-isolate"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5171, "Electrically Isolate Space",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6171, "Acknowledge order to Electrically
        isolate space",
      [compartment = Compartment, source = Station], untraceable-
        reports, _)
  END RULE
END GMO

GMO 6176
FOR ECL 6176 "Acknowledge order to Patch hull rupture"
LET source -> Station,
    deck -> Deck,
    frame -> Frame

  RULE 6176.Patch-hull.interpret.1 "valid acknowledge Patch-hull"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5176,
        "Patch Hull Rupture",
      [deck = Deck, frame = Frame, target = Station], _, A)
  THEN
    report(create, _, 6176, "Acknowledge order to Patch hull
        rupture",
```

```
      [deck = Deck, frame = Frame, source = Station], A, _)
  END RULE

  RULE 6176.Patch-hull.interpret.2 "invalid acknowledge Patch-hull"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5176, "Patch Hull Rupture",
      [deck = Deck, frame = Frame, target = Station], _, _)
  THEN
    report(create, _, 6176, "Acknowledge order to Patch hull
       rupture",
      [deck = Deck, frame = Frame, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6180
FOR ECL 6180 "Acknowledge order to set reflash watch"
LET source -> Station,
    compartment -> Compartment

  RULE 6180.set-reflash.interpret.1 "valid acknowledge set-reflash"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5180,
       "Set Reflash Watch",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6180, "Acknowledge order to set reflash
       watch",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6180.set-reflash.interpret.2 "invalid acknowledge set-
       reflash"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5180, "Set Reflash Watch",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6180, "Acknowledge order to set reflash
       watch",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6182
FOR ECL 6182 "Acknowledge order to secure reflash watch"
LET source -> Station,
    compartment -> Compartment
```

```
  RULE 6182.secure-reflash.interpret.1 "valid acknowledge secure-
       reflash"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5182,
       "Secure Reflash Watch",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6182, "Acknowledge order to secure reflash
       watch",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6182.secure-reflash.interpret.2 "invalid acknowledge secure-
       reflash"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5182, "Secure Reflash Watch",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6182, "Acknowledge order to secure reflash
       watch",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6186
FOR ECL 6186 "Acknowledge order to Route casualty power"
LET source -> Station,
    power_point -> PowerPoints

  RULE 6186.Route-casualty.interpret.1 "valid acknowledge Route-
       casualty"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5186,
       "Route Casualty Power",
      [power_point = PowerPoints, target = Station], _, A)
  THEN
    report(create, _, 6186, "Acknowledge order to Route casualty
       power",
      [power_point = PowerPoints, source = Station], A, _)
  END RULE

  RULE 6186.Route-casualty.interpret.2 "invalid acknowledge Route-
       casualty"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5186, "Route Casualty Power",
      [power_point = PowerPoints, target = Station], _, _)
  THEN
    report(create, _, 6186, "Acknowledge order to Route casualty
       power",
      [power_point = PowerPoints, source = Station], untraceable-
       reports, _)
```

```
  END RULE
END GMO

;*****************************************************************

GMO 6190
FOR ECL 6190 "Acknowledge order to Conduct post-fire gas-free test"
LET source -> Station,
    compartment -> Compartment

  RULE 6190.Conduct-test.interpret.1 "valid acknowledge Conduct-
       post-fire-test"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5190,
        "Conduct Post-fire Gas-free Test",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6190, "Acknowledge order to Conduct post-fire
        gas-free test",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6190.Conduct-testt.interpret.2 "invalid acknowledge Conduct-
       post-fire-test"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5190, "Conduct Post-fire Gas-free Test",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6190, "Acknowledge order to Conduct post-fire
        gas-free test",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6195
FOR ECL 6195 "Acknowledge order to Assist another repair locker"
LET source -> Station,
    casualty -> Casualty,
    compartment -> Compartment,
    repair_locker -> RepairLocker

  RULE 6195.Assist-another.interpret.1 "valid acknowledge Assist-
       another"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5195,
        "Assist Another Repair Locker",
      [casualty = Casualty, compartment = Compartment, repair_locker
       = RepairLocker, target = Station],
          _, A)
  THEN
    report(create, _, 6195, "Acknowledge order to Assist another
        repair locker",
```

```
      [casualty = Casualty, compartment = Compartment, repair_locker
       = RepairLocker, source = Station],
          A, _)
  END RULE

  RULE 6195.Assist-another.interpret.2 "invalid acknowledge Assist-
       another"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5195, "Assist Another Repair Locker",
      [casualty = Casualty, compartment = Compartment, repair_locker
       = RepairLocker, target = Station],
          _, _)
  THEN
    report(create, _, 6195, "Acknowledge order to Assist another
        repair locker",
      [casualty = Casualty, compartment = Compartment, repair_locker
       = RepairLocker, source = Station],
          untraceable-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6201
FOR ECL 6201 "Flooding magazine"
LET source -> Station,
    compartment -> Compartment

  RULE 6201.flood-magazine.interpret.1 "valid flood magazine"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5101,
        "Flood magazine",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6201, "Flooding magazine",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6201.flood-magazine.interpret.2 "invalid flood magazine"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5101, "Flood magazine",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6201, "Flooding magazine",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6210
FOR ECL 6210 "Setting boundaries fire/flood/smoke in response to
       query"
```

```
LET source -> Station,
    casualty -> Casualty,
    compartment -> Compartment

  RULE 6210.set-boundaries.interpret.1 "valid set-boundaries"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5110,
        "Set boundaries fire/flood/smoke",
      [casualty = Casualty, compartment = Compartment, target =
        Station], _, A)
  THEN
    report(create, _, 6210, "Setting boundaries fire/flood/smoke in
        response to query",
      [casualty = Casualty, compartment = Compartment, source =
        Station], A, _)
  END RULE

  RULE 6210.set-boundaries.interpret.2 "invalid set-boundaries"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5110, "Set boundaries fire/flood/smoke ",
      [casualty = Casualty, compartment = Compartment, target =
        Station], _, _)
  THEN
    report(create, _, 6210, "Setting boundaries fire/flood/smoke in
        response to query",
      [casualty = Casualty, compartment = Compartment, source =
        Station], untraceable-reports, _)
  END RULE
END GMO


;*****************************************************************

GMO 6214
FOR ECL 6214 "Setting boundaries fire/flood/smoke in response to
       query"
LET source -> Station,
    casualty -> Casualty

  RULE 6214.set-boundaries.interpret.1 "valid set-boundaries "
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5114,
        "Set boundaries",
      [casualty = Casualty, target = Station], _, A)
  THEN
    report(create, _, 6214, "Setting boundaries fire/flood/smoke in
        response to query",
      [casualty = Casualty, source = Station], A, _)
  END RULE

  RULE 6214.set-boundaries.interpret.2 "invalid set-boundaries "
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5114, "Set boundaries",
      [casualty = Casualty, target = Station], _, _)
```

```
  THEN
    report(create, _, 6214, "Setting boundaries fire/flood/smoke in
        response to query",
      [casualty = Casualty, source = Station], untraceable-reports,
       _)
  END RULE
END GMO

;*****************************************************************

GMO 6220
FOR ECL 6220 "Fighting fire in space in response to request of
       status of firefighting"
LET source -> Station,
    compartment -> Compartment,
    time -> Time

  RULE 6220.fight-fire.interpret.1 "valid fight-fire"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5120,
        "Fight fire in space",
      [compartment = Compartment, time = Time, target = Station], _,
       A)
  THEN
    report(create, _, 6220, "Fighting fire in space in response to
        request of status of firefighting",
      [compartment = Compartment, time = Time, source = Station], A,
       _)
  END RULE

  RULE 6220.fight-fire.interpret.2 "invalid fight-fire"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5120, "Fight fire in space",
      [compartment = Compartment, time = Time, target = Station], _,
       _)
  THEN
    report(create, _, 6220, "Fighting fire in space in response to
        request of status of firefighting",
      [compartment = Compartment, time = Time, source = Station],
       untraceable-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6230
FOR ECL 6230 "Dewatering space"
LET source -> Station,
    compartment -> Compartment

  RULE 6230.dewater-space.interpret.1 "valid dewater-space"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5130,
        "Dewater space",
      [compartment = Compartment, target = Station], _, A)
```

```
  THEN
    report(create, _, 6230, "Dewatering space",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6230.dewater-space.interpret.2 "invalid dewater-space"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5130, "Dewater space",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6230, "Dewatering space",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************
GMO 6240
FOR ECL 6240 "Desmoking space"
LET source -> Station,
    compartment -> Compartment

  RULE 6240.desmoke-space.interpret.1 "valid desmoke-space"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5140,
        "Desmoke space",
            [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6240, "Desmoking space",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6240.desmoke-space.interpret.2 "invalid desmoke-space"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5140, "Desmoke space",
                [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6240, "Desmoking space",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6242
FOR ECL 6242 "Active desmoking space"
LET source -> Station,
    compartment -> Compartment

  RULE 6242.active-desmoke.interpret.1 "valid active-desmoke"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5142,
        "Active desmoke space",
```

```
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6242, "Active desmoking space",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6242. active-desmoke.interpret.2 "invalid active-desmoke"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5142, "Active desmoke space",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6242, "Active desmoking space",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6262
FOR ECL 6262 "Firemain or Chillwater rupture isolated, and patching
       in progress"
LET source -> Station,
    compartment -> Compartment,
    loop -> Loop

  RULE 6262.isolate-patch.interpret.1 "valid isolate-patch"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5162,
           "Isolate and patch Firemain or Chillwater rupture",
      [compartment = Compartment, loop = Loop, target = Station], _,
       A)
  THEN
    report(create, _, 6262, "Firemain or Chillwater rupture
       isolated, and patching in progress",
      [compartment = Compartment, loop = Loop, source = Station], A,
       _)
  END RULE

  RULE 6262.isolate-patch.interpret.2 "invalid isolate-patch"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5162,
            "Isolate and patch Firemain or Chillwater rupture",
      [compartment = Compartment, loop = Loop, target = Station], _,
       _)
  THEN
    report(create, _, 6262, "Firemain or Chillwater rupture
       isolated, and patching in progress",
      [compartment = Compartment, loop = Loop, source = Station],
       untraceable-reports, _)
  END RULE
END GMO

;*****************************************************************
```

```
GMO 6266
FOR ECL 6266 "Patching Firemain or Chillwater rupture in place"
LET source -> Station,
    compartment -> Compartment,
    loop -> Loop

  RULE 6266.patch-rupture.interpret.1 "valid patch-rupture"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5166,
           "Patch Firemain or Chillwater rupture in place",
      [compartment = Compartment, loop = Loop, target = Station], _,
       A)
  THEN
    report(create, _, 6266, "Patching Firemain or Chillwater rupture
       in place",
      [compartment = Compartment, loop = Loop, source = Station], A,
       _)
  END RULE

  RULE 6266. patch-rupture.interpret.2 "invalid patch-rupture"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5166,
          "Patch Firemain or Chillwater rupture in place",
      [compartment = Compartment, loop = Loop, target = Station], _,
       _)
  THEN
    report(create, _, 6266, "Patching Firemain or Chillwater rupture
       in place",
      [compartment = Compartment, loop = Loop, source = Station],
       untraceable-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6270
FOR ECL 6270 "Electrical and mechanical isolation in progress"
LET source -> Station,
    compartment -> Compartment

  RULE 6270.isolation.interpret.1 "valid isolation"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5170,
   "Electrically and mechanically isolate space",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6270, "Electrical and mechanical isolation in
       progress",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6270.isolation.interpret.2 "invalid isolation"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
       5170,
```

```
   "Electrically and mechanically isolate space",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6270, "Electrical and mechanical isolation in
        progress",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6276
FOR ECL 6276 "Patching hull rupture"
LET source -> Station,
    compartment -> Compartment

  RULE 6276.patch-rupture.interpret.1 "valid patch-rupture"
  IF
    action(find, [correct, sub-optimal, error-of-comission], 5176,
        "Patch hull rupture",
      [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6276, "Patching hull rupture",
      [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6276.patch-rupture.interpret.2 "invalid patch-rupture"
  IF
    NOT action(find, [correct, sub-optimal, error-of-comission],
        5176, "Patch hull rupture",
      [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6276, "Patching hull rupture",
      [compartment = Compartment, source = Station], untraceable-
       reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6301
FOR ECL 6301 "Magazine Flooded"
LET compartment -> Compartment

  ; place report node

  RULE 6301.magazine-flooded.interpret.1 "associate magazine flooded
       report with flood magazine action"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5101,
        "Flood Magazine",
           [compartment = Compartment], _, A)
  THEN
    report(create, _, 6301, "Magazine Flooded", [compartment =
        Compartment], A, _)
```

```
  END RULE

  RULE 6301.magazine-flooded.interpret.2 "cannot find action that
       flooded magazine report is associated with"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5101, "Flood Magazine",
           [compartment = Compartment], _, A)
  THEN
    report(create, _, 6301, "Magazine Flooded", [compartment =
       Compartment], miscellaneous-reports, _)
  END RULE

  ; solve crises and satisfy goals

  RULE 6301.magazine-flooded.interpret.3 "magazine flooded report
       solves fire crisis"
  IF
    crisis(find, unsolved, 8150, "Hot Magazines", [compartment =
       Compartment], _, C)
    AND goal(find, [unaddressed, addressed], 7150, "Protect
       Magazines", [compartment = Compartment], C, G)
  THEN
    crisis(modify, solved, 8150, "Hot Magazines", [compartment =
       Compartment], _, C)
    goal(modify, satisfied, 7150, "Protect Magazines", [compartment
       = Compartment], _, G)
  END RULE
END GMO


;*****************************************************************

GMO 6314
FOR ECL 6314 "Boundaries set on compartment fire/flood/smoke"
LET compartment -> Compartment,
    source -> Station

  ; place report node

  RULE 6314.boundaries-set.interpret.1 "associate report with set
       boundaries on compartment order"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5114,
           "Set boundaries on compartment fire/flood/smoke",
           [target = Station, compartment = Compartment], _, A)
  THEN
    report(create, _, 6310, "Boundaries set on compartment
       fire/flood/smoke",
           [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6314.boundaries-set.interpret.2 "cannot find action to set
       boundaries"
  IF
```

```
    NOT action(find, [correct, sub-optimal, error-of-commission],
        5114,
            "Set boundaries on compartment fire/flood/smoke",
            [target = Station, compartment = Compartment], _, _)
  THEN
    report(create, _, 6310, "Boundaries set on compartment
        fire/flood/smoke",
            [compartment = Compartment, source = Station],
        miscellaneous-actions, _)
  END RULE

  ; satisfy goals

  RULE 6314.boundaries-set.interpret.3 "satisfy contain fire goal"
  IF
    goal(find, [unaddressed, addressed], 7110, "Contain Fire",
        [compartment = Compartment], _, G)
  THEN
    goal(modify, satisfied, 7110, "Contain Fire", [compartment =
        Compartment], _, G)
  END RULE
END GMO

;*****************************************************************

GMO 6324
FOR ECL 6324 "Fire extinguished"
LET compartment -> Compartment,
    source -> Station

  RULE 6324.fire-extinguished.interpret.1 "associate report with
       fight fire action"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5120,
        "Fight fire in space",
            [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6324, "Fire extinguished", [compartment =
        Compartment, source = Station], A, _)
  END RULE

  RULE 6324.fire-extinguished.interpret.2 "associate the report
       directly with apply fire suppressant goal"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
        5120, "Fight fire in space",
            [compartment = Compartment], _, _)
    AND goal(find, _, 7118, "Apply Fire Suppressant", [compartment =
        Compartment], _, G)
  THEN
    report(create, _, 6324, "Fire extinguished", [compartment =
        Compartment, source = Station], G, _)
  END RULE

  RULE 6324.fire-extinguished.interpret.3 "nothing to associate fire
       extinguished report with"
```

```
IF
  NOT action(find, [correct, sub-optimal, error-of-commission],
      5120, "Fight fire in space",
          [compartment = Compartment], _, _)
  AND NOT goal(find, _, 7118, "Apply Fire Suppressant",
      [compartment = Compartment], _, _)
THEN
  report(create, _, 6324, "Fire extinguished", [compartment =
      Compartment, source = Station],
          miscellaneous-reports, _)
END RULE

RULE 6324.fire-extinguished.interpret.4 "fire extinguished
      satisfies extinguish fire goal"
IF
  goal(find, [unaddressed, addressed], 7115, "Extinguish Fire",
      [compartment = Compartment], _, G)
THEN
  goal(modify, satisfied, 7115, "Extinguish Fire", [compartment =
      Compartment], _, G)

  RULE 6324.fire-extinguished.suggest.1 "if there is residual
      smoke, order desmoking"
  IF
    report(find, _, 6801, "Damage Report Fire/Flood/Smoke",
           [casualty = "smoke", compartment = Compartment], _, _)
    AND goal(find, unaddressed, 7125, "Clear Smoke", [compartment
      = Compartment], _, G)
    AND world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                [compartment = Compartment, station = RL], _, _)
  THEN
    action(create, pending, 5140, "Desmoke space", [compartment =
      Compartment, target = RL], G, _)
  END RULE

  RULE 6324.fire-extinguished.suggest.2 "if there is water, order
      dewatering"
  IF
    report(find, _, 6801, "Damage Report Fire/Flood/Smoke",
           [casualty = "flood", compartment = Compartment], _, _)
    AND goal(find, unaddressed, 7130, "Clear Water from
      Firefighting", [compartment = Compartment], _, G)
    AND world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                [compartment = Compartment, station = RL], _, _)
  THEN
    action(create, pending, 5130, "Dewater space", [compartment =
      Compartment, target = RL], G, _)
  END RULE

  RULE 6324.fire-extinguished.interpret.5 "If there is neither
      residual smoke nor water, the crisis is solved"
  IF
    NOT report(find, _, 6801, "Damage Report Fire/Flood/Smoke",
           [casualty = "smoke", compartment = Compartment], _, _)
```

```
      AND NOT report(find, _, 6801, "Damage Report
        Fire/Flood/Smoke",
              [casualty = "flood", compartment = Compartment], _, _)
      AND crisis(find, unsolved, 8100, "Fire", [compartment =
        Compartment], _, C)
      AND goal(find, _, 7100, "Control Fire", [compartment =
        Compartment], C, G)
    THEN
      crisis(modify, solved, 8100, "Fire", [compartment =
        Compartment], _, C)
      goal(modify, satisfied, 7100, "Control Fire", [compartment =
        Compartment], C, G)
    END RULE
  END RULE
END GMO

;*************************************************************

GMO 6330
FOR ECL 6330 "Space Dewatered"
LET compartment -> Compartment,
    source -> Station

  RULE 6330.space-dewatered.interpret.1 "Space Dewatered associated
       with Remove Water goal"
  IF
    goal(find, [unaddressed, addressed, satisfied], 7218, "Remove
       Water", [compartment = Compartment], _, G)
  THEN
    report(create, _, 6330, "Space Dewatered", [compartment =
       Compartment, source = Station], G, _)
  END RULE

  RULE 6330.space-dewatered.interpret.2 "Space Dewatered associated
       with Clear Water from Firefighting"
  IF
    goal(find, [unaddressed, addressed, satisfied], 7130, "Clear
       Water from Firefighting",
         [compartment = Compartment], _, G)
  THEN
    report(create, _, 6330, "Space Dewatered", [compartment =
       Compartment, source = Station], G, _)
  END RULE

  RULE 6330.space-dewatered.interpret.3 "Space Dewatered report
       untraceable"
  IF
    NOT goal(find, [unaddressed, addressed, satisfied], 7218,
       "Remove Water", [compartment = Compartment], _, _)
    AND NOT goal(find, [unaddressed, addressed, satisfied], 7130,
       "Clear Water from Firefighting",
             [compartment = Compartment], _, _)
  THEN
    report(create, _, 6330, "Space Dewatered", [compartment =
       Compartment, source = Station],
          miscellaneous-reports, _)
```

```
  END RULE

  RULE 6330.space-dewatered.interpret.4 "Space Dewatered satisfies
       Flooding goals and solves Flood crisis"
  IF
    crisis(find, unsolved, 8200, "Flood", [compartment =
       Compartment], _, C)
    AND goal(find, [unaddressed, addressed], 7200, "Control
       Flooding", [compartment = Compartment], _, G)
  THEN
    crisis(modify, solved, 8200, "Flood", [compartment =
       Compartment], _, C)
    goal(modify, satisfied, 7200, "Control Flooding", [compartment =
       Compartment], _, G)
  END RULE

  RULE 6330.space-dewatered.interpret.5 "Space Dewatered satisfies
       Clear Water from Firefighting Goal"
  IF
    goal(find, [unaddressed, addressed], 7130, "Clear Water from
       Firefighting",
         [compartment = Compartment], _, G)
  THEN
    goal(modify, satisfied, 7130, "Clear Water from Firefighting",
       [compartment = Compartment], _, G)

    RULE 6330.space-dewatered.interpret.5 "Space Dewatered marks end
       of fire crisis"
    IF
      goal(find, satisfied, 7115, "Extinguish Fire", [compartment =
       Compartment], _, _)
      AND goal(find, satisfied, 7120, "Set Reflash Watch",
       [compartment = Compartment], _, _)
      AND goal(find, satisfied, 7125, "Clear Smoke", [compartment =
       Compartment], _, _)
      AND goal(find, [addressed, unaddressed], 7100, "Control Fire",
       [compartment = Compartment], _, GCF)
      AND crisis(find, unsolved, 8100, "Fire", [compartment =
       Compartment], _, C)
    THEN
      goal(modify, satisfied, 7100, "Control Fire", [compartment =
       Compartment], _, GCF)
      crisis(modify, solved, 8100, "Fire", [compartment =
       Compartment], _, C)
    END RULE
  END RULE
END GMO


;*****************************************************************

GMO 6340
FOR ECL 6340 "Space Desmoked"
LET compartment -> Compartment,
    source -> Station
```

```
RULE 6340.space-desmoked.interpret.1 "Space Desmoked associated
     with Remove Water goal"
IF
  goal(find, [unaddressed, addressed, satisfied], 7318, "Remove
     Smoke", [compartment = Compartment], _, G)
THEN
  report(create, _, 6340, "Space Desmoked", [compartment =
     Compartment, source = Station], G, _)
END RULE

RULE 6340.space-desmoked.interpret.2 "Space Desmoked associated
     with Clear Smoke"
IF
  goal(find, [unaddressed, addressed, satisfied], 7125, "Clear
     Smoke",
       [compartment = Compartment], _, G)
THEN
  report(create, _, 6340, "Space Desmoked", [compartment =
     Compartment, source = Station], G, _)
END RULE

RULE 6340.space-desmoked.interpret.3 "Space Desmoked report
     untraceable"
IF
  NOT goal(find, [unaddressed, addressed, satisfied], 7318,
     "Remove Smoke", [compartment = Compartment], _, _)
  AND NOT goal(find, [unaddressed, addressed, satisfied], 7125,
     "Clear Smoke", [compartment = Compartment], _, _)
THEN
  report(create, _, 6340, "Space Desmoked", [compartment =
     Compartment, source = Station],
        miscellaneous-reports, _)
END RULE

RULE 6340.space-desmoked.interpret.4 "Space Desmoked satisfies
     Smoke goals and solves Smoke crisis"
IF
  crisis(find, unsolved, 8300, "Smoke", [compartment =
     Compartment], _, C)
  AND goal(find, [unaddressed, addressed], 7300, "Control Smoke",
     [compartment = Compartment], _, G)
THEN
  crisis(modify, solved, 8300, "Smoke", [compartment =
     Compartment], _, C)
  goal(modify, satisfied, 7300, "Control Smoke", [compartment =
     Compartment], _, G)
END RULE

RULE 6340.space-desmoked.interpret.5 "Space Desmoked satisfies
     Clear Water from Firefighting Goal"
IF
  goal(find, [unaddressed, addressed], 7125, "Clear Smoke",
     [compartment = Compartment], _, G)
THEN
  goal(modify, satisfied, 7125, "Clear Smoke", [compartment =
     Compartment], _, G)
```

```
    RULE 6340.space-desmoked.interpret.5 "Space Desmoked marks end
       of fire crisis"
    IF
      goal(find, satisfied, 7115, "Extinguish Fire", [compartment =
       Compartment], _, _)
      AND goal(find, satisfied, 7120, "Set Reflash Watch",
       [compartment = Compartment], _, _)
      AND goal(find, satisfied, 7130, "Clear Water from
       Firefighting", [compartment = Compartment], _, _)
      AND goal(find, [addressed, unaddressed], 7100, "Control Fire",
       [compartment = Compartment], _, GCF)
      AND crisis(find, unsolved, 8100, "Fire", [compartment =
       Compartment], _, C)
    THEN
      goal(modify, satisfied, 7100, "Control Fire", [compartment =
       Compartment], _, GCF)
      crisis(modify, solved, 8100, "Fire", [compartment =
       Compartment], _, C)
    END RULE
  END RULE
END GMO


;*****************************************************************

GMO 6342
FOR ECL 6342 "Active desmoking rigged"
LET compartment -> Compartment,
    source -> Station

  RULE 6342.active-desmoke.interpret.1 "associate report with active
       desmoke action"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5142,
       "Active desmoke space",
           [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6342, "Active desmoking rigged", [compartment
       = Compartment, source = Station], A, _)
  END RULE

  RULE 6342.active-desmoke.interpret.2 "no action to associate
       active desmoking rigged report with"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5142, "Active desmoke space",
           [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6342, "Active desmoking rigged", [compartment
       = Compartment, source = Station],
           miscellaneous-reports, _)
  END RULE

  RULE 6342.active-desmoke.interpret.3 "active desmoking rigged
       satisfies active desmoke if necessary goal"
```

```
  IF
    goal(find, [addressed, unaddressed], 7117, "Active Desmoke",
        [compartment = Compartment], _, G)
  THEN
    goal(modify, satisfied, 7117, "Active Desmoke ", [compartment =
        Compartment], _, G)

    RULE 6342.active-desmoke.suggest.1 "active desmoking rigged
        completes prerequisites for firefighting"
    IF
      EITHER
        goal(find, satisfied, 7116, "Isolate Compartment",
        [compartment = Compartment], _, _)
      OR
        NOT world-state(find, _, 3360, "Can Isolate Compartment",
        [compartment = Compartment], _, _)
      END EITHER
      AND goal(find, unaddressed, 7118, "Apply Fire Suppressant",
        [compartment = Compartment], _, GAFS)
      AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = RL], _, _)
      AND EITHER
        NOT world-state(find, _, 3304, "Compartment Has Halon",
        [compartment = Compartment], _, _)
      OR
        report(find, _, 6420, "Halon Ineffective", [compartment =
        Compartment], _, _)
      END EITHER
    THEN
      action(create, pending, 5120, "Fight fire in space",
        [compartment = Compartment, target = RL], GAFS, _)
    END RULE
  END RULE
END GMO


;*************************************************************

GMO 6350
FOR ECL 6350 "Firemain valve opened/closed"
LET valve -> Valve,
    valve_action -> ValveAction,
    source -> Station

  RULE 6356.valve-status.interpret.1 "Associate valve open/closed
       report with open/close valve action"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5150,
        "Open/Close firemain valve",
            [valve = Valve, valve_action = ValveAction], _, A)
  THEN
    report(create, _, 6350, "Firemain valve opened/closed",
            [valve = Valve, valve_action = ValveAction, source =
        Station], A, _)
  END RULE
```

```
  RULE 6356.valve-status.interpret.2 "Nothing to associate valve
       status with"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5150, "Open/Close firemain valve",
           [valve = Valve, valve_action = ValveAction], _, _)
  THEN
    report(create, _, 6350, "Firemain valve opened/closed",
           [valve = Valve, valve_action = ValveAction, source =
       Station], miscellaneous-reports, _)
  END RULE

  ; *** TODO: add callback message to check firemain pressure status
       ***

END GMO

;*************************************************************

GMO 6356
FOR ECL 6356 "Fire pump started/stopped"
LET pump -> Pump,
    pump_stat -> PumpState,
    source -> Station

  RULE 6356.pump-status.interpret.1 "Associate pump started/stopped
       report with pump order"
  IF
    world-state(find, _, 4107, "Pump state to pump action",
                [pump_state = PumpState, pump_action = PumpAction],
       _, _)
    AND action(find, [correct, sub-optimal, error-of-commission],
       5156, "Start/Stop fire pump",
           [pump = Pump, pump_action = PumpAction], _, A)
  THEN
    report(create, _, 6356, "Fire pump started/stopped",
           [pump = Pump, pump_state = PumpState, source = Station],
       A, _)
  END RULE

  RULE 6356.pump-status.interpret.2 "Pump started/stopped report
       cannot be associated with pump order"
  IF
    world-state(find, _, 4107, "Pump state to pump action",
                [pump_state = PumpState, pump_action = PumpAction],
       _, _)
    AND NOT action(find, [correct, sub-optimal, error-of-
       commission], 5156, "Start/Stop fire pump",
           [pump = Pump, pump_action = PumpAction], _, A)
  THEN
    report(create, _, 6356, "Fire pump started/stopped",
           [pump = Pump, pump_state = PumpState, source = Station],
       miscellaneous-reports, _)
  END RULE
```

```
  ; *** TODO: add callback message to check firemain pressure status
      ***

END GMO



;*****************************************************************

GMO 6362
FOR ECL 6362 "Firemain or Chillwater rupture has been patched"
LET loop -> Loop,
    compartment -> Compartment,
    source -> Station

  RULE 6362.pipe-patched.interpret.1 "associate patch report with
      patch pipe in place action"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5166,
           "Patch firemain or chillwater rupture in place",
           [compartment = Compartment, target = Station, loop =
      Loop], _, A)
  THEN
    report(create, _, 6362, "Firemain or Chillwater rupture has been
      patched",
           [compartment = Compartment, source = Station, loop =
      Loop], A, _)
  END RULE

  RULE 6362.pipe-patched.interpret.2 "associate patch report with
      isolate and patch order"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5166,
           "Isolate and patch Firemain or Chillwater rupture",
           [compartment = Compartment, target = Station, loop =
      Loop], _, A)
  THEN
    report(create, _, 6362, "Firemain or Chillwater rupture has been
      patched",
           [compartment = Compartment, source = Station, loop =
      Loop], A, _)
  END RULE

  RULE 6362.pipe-patched.interpret.3 "patch report cannot be
      associated with an action"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
      5166,
           "Patch firemain or chillwater rupture in place",
           [compartment = Compartment, target = Station, loop =
      Loop], _, A)
    AND NOT action(find, [correct, sub-optimal, error-of-
      commission], 5166,
           "Isolate and patch Firemain or Chillwater rupture",
           [compartment = Compartment, target = Station, loop =
      Loop], _, A)
```

```
  THEN
    report(create, _, 6362, "Firemain or Chillwater rupture has been
        patched",
            [compartment = Compartment, source = Station, loop =
        Loop], miscellaneous-reports, _)
  END RULE

  RULE 6362.pipe-patched.interpret.4 "patch report satisfies goal
        and ends crisis"
  IF
    crisis(find, unsolved, 8500, "Pipe Rupture", [compartment =
        Compartment, system = Loop], _, C)
    AND goal(find, _, 7500, "Patch Pipe Rupture", [compartment =
        Compartment, system = Loop], C, G)
  THEN
    crisis(modify, solved, 8500, "Pipe Rupture", [compartment =
        Compartment, system = Loop], _, C)
    goal(modify, satusfied, 7500, "Patch Pipe Rupture", [compartment
        = Compartment, system = Loop], C, G)
  END RULE
END GMO


;*****************************************************************

GMO 6370
FOR ECL 6370 "Space electrically and mechanically isolated"
LET compartment -> Compartment,
    source -> Station

  RULE 6370.space-isolated.interpret.1 "associate space isolated
        report with action"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5170,
            "Electrically and Mechanically Isolate Space",
        [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6370, "Space electrically and mechanically
        isolated",
            [compartment = Compartment, source = Station], A, _)
  END RULE

  RULE 6370.space-isolated.interpret.2 "no action with which to
        associate space isolated report"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
        5170,
            "Electrically and Mechanically Isolate Space",
        [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6370, "Space electrically and mechanically
        isolated",
            [compartment = Compartment, source = Station],
        miscellaneous-reports, _)
  END RULE
```

```
RULE 6370.space-isolated.interpret.3 "space isolated satisfies
     isoalte space goal for dewatering"
IF
  goal(find, unaddressed, 7215, "Eliminate Flooding", [compartment
     = Compartment], _, GEF)
  AND goal(find, [unaddressed, addressed], 7116, "Isolate Space",
     [compartment = Compartment], GEF, G)
THEN
  goal(modify, satisfied, 7116, "Isolate Space", [compartment =
     Compartment], _, G)

  RULE 6370.space-isolated.suggest.1 "space isolated allows
     dewatering"
  IF
    goal(find, unaddressed, 7218, "Remove Water", [compartment =
     Compartment], GEF, GRW)
    AND world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
               [compartment = Compartment, station = RL], _, _)
  THEN
    action(create, pending, 5130, "Dewater space", [compartment =
     Compartment, target = RL], GRW, _)
  END RULE
END RULE

RULE 6370.space-isolated.interpret.4 "space isolated satisfies
     isoalte space goal for desmoking"
IF
  goal(find, unaddressed, 7315, "Eliminate Smoke", [compartment =
     Compartment], _, GES)
  AND goal(find, [unaddressed, addressed], 7116, "Isolate Space",
     [compartment = Compartment], GES, G)
THEN
  goal(modify, satisfied, 7116, "Isolate Space", [compartment =
     Compartment], _, G)

  RULE 6370.space-isolated.suggest.2 "space isolated allows
     desmoking"
  IF
    goal(find, unaddressed, 7318, "Remove Smoke", [compartment =
     Compartment], GES, GRS)
    AND world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
               [compartment = Compartment, station = RL], _, _)
  THEN
    action(create, pending, 5140, "Desmoke space", [compartment =
     Compartment, target = RL], GRS, _)
  END RULE
END RULE

RULE 6370.space-isolated.interpret.5 "space isolated satisfies
     isoalte space goal for firefighting"
IF
  goal(find, unaddressed, 7115, "Extinguish Fire", [compartment =
     Compartment], _, GEF)
```

```
    AND goal(find, [unaddressed, addressed], 7116, "Isolate Space",
        [compartment = Compartment], GEF, G)
  THEN
    goal(modify, satisfied, 7116, "Isolate Space", [compartment =
        Compartment], _, G)

    RULE 6370.space-isolated.suggest.3 "space isolated allows
        firefighting"
    IF
      EITHER
        goal(find, satisfied, 7117, "Active Desmoke", [compartment =
        Compartment], _, _)
      OR
        NOT world-state(find, _, 3361, "Should Active Desmoke
        Compartment", [compartment = Compartment], _, _)
      END EITHER
      AND goal(find, unaddressed, 7118, "Apply Fire Suppressant",
        [compartment = Compartment], _, GAFS)
      AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = RL], _, _)
      AND EITHER
        NOT world-state(find, _, 3304, "Compartment Has Halon",
        [compartment = Compartment], _, _)
      OR
        report(find, _, 6420, "Halon Ineffective", [compartment =
        Compartment], _, _)
      END EITHER
    THEN
      action(create, pending, 5120, "Fight fire in space",
        [compartment = Compartment, target = RL], GAFS, _)
    END RULE
  END RULE
END GMO


;*****************************************************************

GMO 6376
FOR ECL 6376 "Hull rupture patched"
LET compartment -> Compartment,
    source -> Station

  RULE 6376.hull-patched.interpret.1 "associate patch report with
       patch hull order"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5176,
        "Patch hull rupture",
            [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6376, "Hull rupture patched", [compartment =
        Compartment, source = Station], A, _)
  END RULE

  RULE 6376.hull-patched.interpret.2 "patch report cannot be
       associated with an action"
```

```
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
        5176, "Patch hull rupture",
            [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6376, "Hull rupture patched", [compartment =
        Compartment, source = Station],
            miscellaneous-reports, _)
  END RULE

  RULE 6376.hull-patched.interpret.3 "patch report satisfies goal
       and ends crisis"
  IF
    crisis(find, unsolved, 8400, "Hull Rupture", [compartment =
        Compartment], _, C)
    AND goal(find, _, 7400, "Patch Hull Rupture", [compartment =
        Compartment], C, G)
  THEN
    crisis(modify, solved, 8400, "Hull Rupture", [compartment =
        Compartment], _, C)
    goal(modify, satusfied, 7400, "Patch Hull Rupture", [compartment
        = Compartment], C, G)
  END RULE
END GMO


;***************************************************************

GMO 6380
FOR ECL 6380 "Zebra Set on Firemain"

  RULE 6380.zebra-firemain.interpret.1 "Zebra set on firemain
       associated with monitor MRZ goal"
  IF
    goal(find, _, 7905, "Monitor Shipwide MRZ Status", [], _, G)
  THEN
    report(create, _, 6380, "Zebra Set on Firemain", [], G, _)

    RULE 6380.zebra-firemain.suggest.1 "Zebra set on firemain
        triggers MRZ set report"
    IF
      world-state(find, _, 3201, "All Stations Manned Ready Zebra",
        [], _, _)
    THEN
      action(create, pending, 5601, "MR/Z set report ship-wide",
        [target = CO], G, _)
    END RULE
  END RULE

  RULE 6380.zebra-firemain.interpret.2 "Zebra set of firemain before
       GQ goal tree created"
  IF
    NOT goal(find, _, 7905, "Monitor Shipwide MRZ Status", [], _, _)
  THEN
    report(create, _, 6380, "Zebra Set on Firemain", [],
        miscellaneous-reports, _)
```

```
  END RULE
END GMO

;*****************************************************************

GMO 6401
FOR ECL 6401 "Valve Damaged"
LET source -> Station,
    valve -> Valve

  RULE 6401.valve-damaged.suggest.1 "Report represents inability to
       order isolate and patch - patch in place"
  IF
    action(find, pending, 5162, "Isolate and Patch firemain or
       chillwater rupture",
           [compartment = Compartment, loop = Loop, valve <~ Valve],
       G, A)
    AND goal(find, unaddressed, 7500, "Patch Pipe Rupture", [], _,
       G)
    AND world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
              [compartment = Compartment, station = RepairLocker],
       _, _)
  THEN
    report(create, _, 6401, "Valve Damaged", [valve = Valve, source
       = Station], A, _)
    action(modify, expired, 5162, "Isolate and Patch firemain or
       chillwater rupture",
           [compartment = Compartment, loop = Loop, valve <~ Valve],
       G, A)
    action(create, pending, 5166, "Patch firemain or chillwater
       rupture in place",
           [compartment = Compartment, target = RepairLocker, loop =
       Loop], G, _)
  END RULE

  RULE 6401.valve-damaged.suggest.2 "Report represents failure of
       isolate and patch - patch in place"
  IF
    action(find, [correct, sub-optimal], 5162, "Isolate and Patch
       firemain or chillwater rupture",
           [compartment = Compartment, loop = Loop, valve <~ Valve],
       G, A)
    AND goal(find, addressed, 7500, "Patch Pipe Rupture", [], _, G)
    AND world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
              [compartment = Compartment, station = RepairLocker],
       _, _)
  THEN
    goal(modify, unaddressed, 7500, "Patch Pipe Rupture", [], _, G)
    report(create, _, 6401, "Valve Damaged", [valve = Valve, source
       = Station], A, _)
    action(create, pending, 5166, "Patch firemain or chillwater
       rupture in place",
           [compartment = Compartment, target = RepairLocker, loop =
       Loop], G, _)
```

```
  END RULE

  RULE 6401.valve-damaged.suggest.3
       "Report represents inability to restore firemain pressure
       using current plan - find alternatives"
  IF
    action(find, pending, 5150, "Open/Close Firemain Valve",
        [valve_action = ValveAction, valve = Valve], G, A)
    AND goal(find, unaddressed, 7560, "Restore Firemain Pressure",
        [loop = Loop], _, G)
  THEN
    action(modify, expired, 5150, "Open/Close Firemain Valve",
           [valve_action = ValveAction, valve = Valve], G, A)
    report(create, _, 6401, "Valve Damaged", [valve = Valve, source
        = Station], A, _)
    CALL get-new-firemain-orders(gmo-name = "6401.valve-damaged",
        goal = G, loop = Loop)
  END RULE

  RULE 6401.valve-damaged.suggest.4
       "Report represents failure of open/close valve operation -
       suggest alternatives"
  IF
    action(find, [correct, sub-optimal], 5150, "Open/Close Firemain
        Valve",
           [valve_action = ValveAction, valve = Valve], G, A)
    AND goal(find, addressed, 7560, "Restore Firemain Pressure",
        [loop = Loop], _, G)
  THEN
    report(create, _, 6401, "Valve Damaged", [valve = Valve, source
        = Station], A, _)
    goal(modify, unaddressed, 7560, "Restore Firemain Pressure",
        [loop = Loop], _, G)
    CALL get-new-firemain-orders(gmo-name = "6401.valve-damaged",
        goal = G, loop = Loop)
  END RULE

  RULE 6401.valve-damaged.interpret.1 "valve damaged report cannot
       be traced to an action"
  IF
    NOT action(find, [pending, correct, sub-optimal], 5162, "Isolate
        and Patch firemain or chillwater rupture",
                [compartment = Compartment, loop = Loop, valve <~
        Valve], _, _)
    AND NOT action(find, pending, 5150, "Open/Close Firemain Valve",
                [valve_action = ValveAction, valve = Valve], _, _)
  THEN
    report(create, _, 6401, "Valve Damaged", [valve = Valve, source
        = Station], miscellaneous-reports, _)
  END RULE
END GMO


;***************************************************************

GMO 6404
```

```
FOR ECL 6404 "Fire Pump Inoperable"
LET source -> Station,
    pump -> Pump

  RULE 6404.pump-inoperable.interpret.1 "pump inoperable report
       associated with start pump order"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5156,
       "Start/Stop Fire Pump",
          [pump = Pump, pump_action = "start"], _, A)
  THEN
    report(create, _, 6404, "Fire Pump Inoperable", [source =
       Station, pump = Pump], A, _)

    RULE 6404.pump-inoperable.suggest.1 "attempt to address achieve
       minimal port pressure goal differently"
    IF
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
       Pump, loop = "port"], _, _)
      AND goal(find, [addressed, unaddressed], 7921, "Achieve
       Minimal Port-Side Firemain Pressure", [], _, G)
      AND world-state(find, _, 4101, "Get Functional Port-Side Fire
       Pump", [pump = NewPump], _, _)
    THEN
      goal(modify, unaddressed, 7921, "Achieve Minimal Port-Side
       Firemain Pressure", [], _, G)
      action(create, pending, 5156, "Start/Stop Fire Pump", [pump =
       NewPump, pump_action = "start"], G, _)
    END RULE

    RULE 6404.pump-inoperable.suggest.2 "attempt to address achieve
       minimal starboard pressure goal differently"
    IF
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
       Pump, loop = "starboard"], _, _)
      AND goal(find, [addressed, unaddressed], 7922, "Achieve
       Minimal Starboard Firemain Pressure", [], _, G)
      AND world-state(find, _, 4102, "Get Functional Starboard Fire
       Pump", [pump = NewPump], _, _)
    THEN
      goal(modify, unaddressed, 7922, "Achieve Minimal Starboard
       Firemain Pressure", [], _, G)
      action(create, pending, 5156, "Start/Stop Fire Pump", [pump =
       NewPump, pump_action = "start"], G, _)
    END RULE

    RULE 6404.pump-inoperable.suggest.3 "attempt to address restore
       pressure goal differently"
    IF
      goal(find, [addressed, unaddressed], 7560, "Restore Firemain
       Pressure", [], _, G)
    THEN
      goal(modify, unaddressed, 7560, "Restore Firemain Pressure",
       [], _, G)
      CALL get-new-firemain-orders(gmo-name = "6404.pump-
       inoperable", goal = G, loop = Loop)
```

```
  END RULE
END RULE

RULE 6404.pump-inoperable.interpret.2 "pump inoperable report
     indicates request permission action must change"
IF
  action(find, pending, 5520, "Request Permission to start fire
     pump", [pump = Pump], G, A)
THEN
  action(modify, expired, 5520, "Request Permission to start fire
     pump", [pump = Pump], G, A)

  RULE 6404.pump-inoperable.suggest.4 "attempt to address achieve
     minimal port pressure goal differently"
  IF
    world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
      Pump, loop = "port"], _, _)
    AND goal(find, unaddressed, 7921, "Achieve Minimal Port-Side
      Firemain Pressure", [], _, G)
    AND world-state(find, _, 4101, "Get Functional Port-Side Fire
      Pump", [pump = NewPump], _, _)
  THEN
    action(create, pending, 5520, "Request Permission to start
      fire pump", [pump = NewPump], G, _)
  END RULE

  RULE 6404.pump-inoperable.suggest.5 "attempt to address achieve
     minimal starboard pressure goal differently"
  IF
    world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
      Pump, loop = "starboard"], _, _)
    AND goal(find, unaddressed, 7922, "Achieve Minimal Starboard
      Firemain Pressure", [], _, G)
    AND world-state(find, _, 4102, "Get Functional Starboard Fire
      Pump", [pump = NewPump], _, _)
  THEN
    action(create, pending, 5520, "Request Permission to start
      fire pump", [pump = NewPump], G, _)
  END RULE

  RULE 6404.pump-inoperable.suggest.6 "attempt to address restore
     pressure goal differently"
  IF
    goal(find, unaddressed, 7560, "Restore Firemain Pressure", [],
      _, G)
  THEN
    CALL get-new-firemain-orders(gmo-name = "6404.pump-
      inoperable", goal = G, loop = Loop)
  END RULE
END RULE

RULE 6404.pump-inoperable.interpret.3 "pump inoperable report
     indicates start pump order must change"
IF
  action(find, pending, 5156, "Start/Stop Fire Pump", [pump =
     Pump, pump_action = "start"], G, A)
```

```
  THEN
    action(modify, expired, 5156, "Start/Stop Fire Pump", [pump =
        Pump, pump_action = "start"], G, A)

    RULE 6404.pump-inoperable.suggest.7 "attempt to address achieve
        minimal port pressure goal differently"
    IF
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "port"], _, _)
      AND goal(find, unaddressed, 7921, "Achieve Minimal Port-Side
        Firemain Pressure", [], _, G)
      AND world-state(find, _, 4101, "Get Functional Port-Side Fire
        Pump", [pump = NewPump], _, _)
    THEN
      action(create, pending, 5156, "Start/Stop Fire Pump", [pump =
        NewPump, pump_action = "start"], G, _)
    END RULE

    RULE 6404.pump-inoperable.suggest.8 "attempt to address achieve
        minimal starboard pressure goal differently"
    IF
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "starboard"], _, _)
      AND goal(find, unaddressed, 7922, "Achieve Minimal Starboard
        Firemain Pressure", [], _, G)
      AND world-state(find, _, 4102, "Get Functional Starboard Fire
        Pump", [pump = NewPump], _, _)
    THEN
      action(create, pending, 5156, "Start/Stop Fire Pump", [pump =
        NewPump, pump_action = "start"], G, _)
    END RULE

    RULE 6404.pump-inoperable.suggest.9 "attempt to address restore
        pressure goal differently"
    IF
      goal(find, unaddressed, 7560, "Restore Firemain Pressure", [],
        _, G)
    THEN
      CALL get-new-firemain-orders(gmo-name = "6404.pump-
        inoperable", goal = G, loop = Loop)
    END RULE
  END RULE

  RULE 6404.pump-inoperable.interpret.4 "pump inoperable untraceable
       to any specific action"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
        5156, "Start/Stop Fire Pump",
                [pump = Pump, pump_action = "start"], _, _)
  THEN
    report(create, _, 6404, "Fire Pump Inoperable", [source =
        Station, pump = Pump], miscellaneous-reports, _)
  END RULE
END GMO
```

```
;*****************************************************************

GMO 6410
FOR ECL 6410 "Cannot patch Firemain or Chillwater rupture in place"
LET compartment -> Compartment,
    loop -> Loop,
    source -> Station

  RULE 6410.cannot-patch.interpret.1 "associate cannot patch in
       place report with patch order"
  IF
    action(find, [correct, sub-optimal, error-of-commission], 5166,
       "Patch Firemain or Chillwater in Place",
          [compartment = Compartment, loop = Loop], G, A)
    AND goal(find, [unaddressed, addressed], 7500, "Patch Pipe
       Rupture",
         [compartment = Compartment, loop = Loop], _, G)
  THEN
    report(create, _, 6410, "Cannot patch Firemain or Chillwater
       rupture in place",
          [compartment = Compartment, loop = Loop, source =
       Station], A, _)
    goal(modify, unaddressed, 7500, "Patch Pipe Rupture",
       [compartment = Compartment, loop = Loop], _, G)

    RULE 6410.cannot-patch.suggest.1 "order isolate and patch
       instead of patch in place"
    IF
      world-state(find, _, 3122, "Pipe Can Be Isolated", [loop =
       Loop, compartment = Compartment], _, _)
      AND world-state(find, _, 4121, "Valves Needed to Isolate",
                 [loop = Loop, compartment = Compartment, valve =
       Valves], _, _)
    THEN
      action(create, pending, 5162, "Isolate and Patch Firemain or
       Chillwater Rupture",
            [compartment = Compartment, loop = Loop, valve =
       Valves, target = Station], G, _)
    END RULE
  END RULE

  RULE 6410.cannot-patch.interpret.2 "cannot trace source of cannot
       patch in place report"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5166, "Patch Firemain or Chillwater in Place",
          [compartment = Compartment, loop = Loop], _, _)
  THEN
    report(create, _, 6410, "Cannot patch Firemain or Chillwater
       rupture in place",
          [compartment = Compartment, loop = Loop, source =
       Station], miscellaneous-reports, _)
  END RULE
END GMO

;*****************************************************************
```

```
GMO 6420
FOR ECL 6420 "Halon Ineffective"
LET compartment -> Compartment,
    source -> Station

  RULE 6420.halon-ineffective.interpret.1
       "halon ineffective means goal of applying fire suppressant is
       not addressed"
  IF
    goal(find, [addressed, unaddressed], 7118, "Apply Fire
       Suppressant", [compartment = Compartment], _, G)
  THEN
    report(create, _, 6420, "Halon Ineffective", [compartment =
       Compartment, source = Station], G, _)
    goal(modify, unaddressed, 7118, "Apply Fire Suppressant",
       [compartment = Compartment], _, G)

    RULE 6420.halon-ineffective.suggest.2 "if firefighting is
       allowed, suggest it"
    IF
      EITHER
        goal(find, satisfied, 7116, "Isolate Compartment",
       [compartment = Compartment], _, _)
      OR
        NOT world-state(find, _, 3360, "Can Isolate Compartment",
       [compartment = Compartment], _, _)
      END EITHER
      AND EITHER
        goal(find, satisfied, 7117, "Active Desmoke", [compartment =
       Compartment], _, _)
      OR
        NOT world-state(find, _, 3361, "Should Active Desmoke
       Compartment", [compartment = Compartment], _, _)
      END EITHER
      AND world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
                 [compartment = Compartment, station = RL], _, _)
    THEN
      action(create, pending, 5120, "Fight fire in space",
       [compartment = Compartment, target = RL], G, _)
    END RULE
  END RULE
END GMO

;*************************************************************

GMO 6602
FOR ECL 6602 "Permission to Flood Magazine from CO"
LET compartment -> Compartment

  RULE 6602.flood-magazines.interpret.1 "permission to flood
       magazines satisfies goal"
  IF
    goal(find, _, 7155, "Try to Get Permission to Flood Magazines",
       [compartment = Compartment], _, G)
```

```
    AND action(find, [correct, sub-optimal, error-of-commission],
        5510, "Request Permission to Flood Magazine",
            [compartment = Compartment], _, A)
  THEN
    goal(modify, satisfied, 7155, "Try to Get Permission to Flood
        Magazines", [compartment = Compartment], _, G)
    report(create, _, 6602, "Permission to Flood Magazine from CO",
        [compartment = Compartment], A, _)
  END RULE

  RULE 6602.flood-magazines.suggest.1 "permission to flood magazines
       triggers flood order"
  IF
    goal(find, unaddressed, 7160, "Prevent Magazine Explosion",
        [compartment = Compartment], _, G)
    AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                [compartment = Compartment, station = Station], _, _)
  THEN
    action(create, pending, 5101, "Flood Magazine", [compartment =
        Compartment, target = Station], G, _)
  END RULE

  RULE 6602.flood-magazines.interpret.1 "denied permission to flood
       magazines refers to erroneous action"
  IF
    NOT goal(find, _, 7155, "Try to Get Permission to Flood
        Magazines", [compartment = Compartment], _, _)
    AND action(find, [correct, sub-optimal, error-of-commission],
        5510, "Request Permission to Flood Magazine",
            [compartment = Compartment], _, A)
  THEN
    report(create, _, 6602, "Permission to Flood Magazine from CO",
        [compartment = Compartment], A, _)
  END RULE

  RULE 6602.flood-magazines.interpret.2 "denied permission to flood
       magazines untraceable"
  IF
    NOT goal(find, _, 7155, "Try to Get Permission to Flood
        Magazines", [compartment = Compartment], _, _)
    AND NOT action(find, [correct, sub-optimal, error-of-
        commission], 5510, "Request Permission to Flood Magazine",
            [compartment = Compartment], _, _)
  THEN
    report(create, _, 6602, "Permission to Flood Magazine from CO",
        [compartment = Compartment],
            miscellaneous-reports, _)
  END RULE
END GMO

;*************************************************************

GMO 6603
FOR ECL 6603 "Denied Permission to Flood Magazine from CO"
LET compartment -> Compartment
```

```
RULE 6603.flood-magazines.interpret.1 "permission denied to flood
     magazines satisfies goal"
IF
  goal(find, _, 7155, "Try to Get Permission to Flood Magazines",
     [compartment = Compartment], _, G)
  AND action(find, [correct, sub-optimal, error-of-commission],
     5510, "Request Permission to Flood Magazine",
        [compartment = Compartment], _, A)
THEN
  goal(modify, satisfied, 7155, "Try to Get Permission to Flood
     Magazines", [compartment = Compartment], _, G)
  report(create, _, 6603, "Denied Permission to Flood Magazine
     from CO", [compartment = Compartment], A, _)
END RULE

RULE 6603.flood-magazines.suggest.1 "denied permission to flood
     magazines triggers set boundaries action"
IF
  goal(find, unaddressed, 7160, "Prevent Magazine Explosion",
     [compartment = Compartment], _, G)
  AND world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
            [compartment = Compartment, station = Station], _, _)
THEN
  action(create, pending, 5114, "Set Boundaries on Compartment
     Fire/Flood/Smoke",
        [casualty = "fire", compartment = Compartment, target =
     Station], G, _)
END RULE

RULE 6603.flood-magazines.interpret.1 "denied permission to flood
     magazines refers to erroneous action"
IF
  NOT goal(find, _, 7155, "Try to Get Permission to Flood
     Magazines", [compartment = Compartment], _, _)
  AND action(find, [correct, sub-optimal, error-of-commission],
     5510, "Request Permission to Flood Magazine",
        [compartment = Compartment], _, A)
THEN
  report(create, _, 6603, "Denied Permission to Flood Magazine
     from CO", [compartment = Compartment], A, _)
END RULE

RULE 6603.flood-magazines.interpret.2 "denied permission to flood
     magazines untraceable"
IF
  NOT goal(find, _, 7155, "Try to Get Permission to Flood
     Magazines", [compartment = Compartment], _, _)
  AND NOT action(find, [correct, sub-optimal, error-of-
     commission], 5510, "Request Permission to Flood Magazine",
        [compartment = Compartment], _, _)
THEN
  report(create, _, 6603, "Denied Permission to Flood Magazine
     from CO", [compartment = Compartment],
        miscellaneous-reports, _)
```

```
  END RULE
END GMO


;*************************************************************

GMO 6610
FOR ECL 6610 "Permission to Start Fire Pump from EOOW"
LET pump -> Pump

  RULE 6610.pump.suggest.1 "permission to start pump allows
       addressing restore firemain pressure"
  IF
    world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = Loop], _, _)
    AND goal(find, [unaddressed, addressed], 7560, "Restore Firemain
        Pressure", [loop = Loop], _, G)
    AND action(find, [correct, sub-optimal, error-of-commission],
        5520, "Request Permission to Start Fire Pump",
            [pump = Pump], G, A)
  THEN
    goal(modify, unaddressed, 7560, "Restore Firemain Pressure",
        [loop = Loop], _, G)
    report(create, _, 7560, "Permission to Start Fire Pump from
        EOOW", [pump = Pump], A, _)
    action(create, pending, 5156, "Start/Stop Fire Pump",
            [pump_action = "start", pump = Pump, target = DCCO], _,
        _)
  END RULE

  RULE 6610.pump.suggest.2 "permission to start pump allows
       addressing achieve minimal port firemain pressure"
  IF
    world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "port"], _, _)
    AND goal(find, [unaddressed, addressed], 7921, "Achieve Minimal
        Port-Side Firemain Pressure", [], _, G)
    AND action(find, [correct, sub-optimal, error-of-commission],
        5520, "Request Permission to Start Fire Pump",
            [pump = Pump], G, A)
  THEN
    goal(modify, unaddressed, 7921, "Achieve Minimal Port-Side
        Firemain Pressure", [], _, G)
    report(create, _, 6610, "Permission to Start Fire Pump from
        EOOW", [pump = Pump], A, _)
    action(create, pending, 5156, "Start/Stop Fire Pump",
            [pump_action = "start", pump = Pump, target = DCCO], _,
        _)
  END RULE

  RULE 6610.pump.suggest.3 "permission to start pump allows
       addressing achieve minimal starboard pressure"
  IF
    world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "starboard"], _, _)
```

```
    AND goal(find, [unaddressed, addressed], 7922, "Achieve Minimal
        Starboard Firemain Pressure", [], _, G)
    AND action(find, [correct, sub-optimal, error-of-commission],
        5520, "Request Permission to Start Fire Pump",
            [pump = Pump], G, A)
  THEN
    goal(modify, unaddressed, 7922, "Achieve Minimal Starboard
        Firemain Pressure", [], _, G)
    report(create, _, 6610, "Permission to Start Fire Pump from
        EOOW", [pump = Pump], A, _)
    action(create, pending, 5156, "Start/Stop Fire Pump",
            [pump_action = "start", pump = Pump, target = DCCO], _,
        _)
  END RULE

  RULE 6610.pump.interpret.1 "permission to start pump refers to
       erroneous action"
  IF
    EITHER
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "port"], _, _)
      AND NOT goal(find, [unaddressed, addressed], 7560, "Restore
        Firemain Pressure", [loop = "port"], _, _)
      AND NOT goal(find, [unaddressed, addressed], 7921, "Achieve
        Minimal Port-Side Firemain Pressure", [], _, _)
    OR
      world-state(find, _, 4106, "Pump to Firemain Loop", [pump =
        Pump, loop = "starboard"], _, _)
      AND NOT goal(find, [unaddressed, addressed], 7560, "Restore
        Firemain Pressure", [loop = "starboard"], _, _)
      AND NOT goal(find, [unaddressed, addressed], 7922, "Achieve
        Minimal Starboard Firemain Pressure", [], _, _)
    END EITHER
    AND action(find, [correct, sub-optimal, error-of-commission],
        5520, "Request Permission to Start Fire Pump",
            [pump = Pump], _, A)
  THEN
    report(create, _, 6610, "Permission to Start Fire Pump from
        EOOW", [pump = Pump], A, _)
  END RULE

  RULE 6610.pump.interpret.2 "permission to start pump untraceable"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
        5520, "Request Permission to Start Fire Pump",
            [pump = Pump], _, A)
  THEN
    report(create, _, 6610, "Permission to Start Fire Pump from
        EOOW", [pump = Pump], miscellaneous-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6735
FOR ECL 6735 "Reprimand for Starting Pump w/o Permission from EOOW"
```

```
  RULE 6735.reprimand.interpret.1 "link reprimand with erroneous
       action"
  IF
    action(find, error-of-commission, 5156, "Start/Stop Fire Pump",
       [pump_action = "start"], _, A)
  THEN
    report(create, _, 6735, "Reprimand for Starting Pump w/o
       Permission from EOOW", [], A, _)
  END RULE

  RULE 6735.reprimand.interpret.2 "just create reprimand report"
  IF
    NOT action(find, error-of-commission, 5156, "Start/Stop Fire
       Pump", [pump_action = "start"], _, _)
  THEN
    report(create, _, 6735, "Reprimand for Starting Pump w/o
       Permission from EOOW", [],
    miscellaneous-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6740.fire
FOR ECL 6740 "Insufficient Personnel"
WHERE casualty = "fire"
LET source -> Station,
    compartment -> Compartment

  RULE 6740.no-personnel-fire.interpret.1 "No personnel for
       erroneous action"
  IF
    action(find, error-of-commission, 5120, "Fight fire in
       compartment",
           [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "fire"], A, _)
  END RULE

  RULE 6740.no-personnel-fire.interpret.2 "No personnel for non-
       error action"
  IF
    action(find, [correct, sub-optimal], 5120, "Fight fire in
       compartment",
           [compartment = Compartment, target = Station], G, A)
  THEN
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "fire"], A, _)

    RULE 6740.no-personnel-fire.suggest.1 "No personnel requires
       alternate order"
    IF
```

```
      world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = NewStation],
        _, _)
      AND goal(find, [unaddressed, addressed], 7118, "Apply fire
        suppressant", [compartment = Compartment], _, G)
    THEN
      action(create, pending, 5120, "Fight Fire in Space",
             [compartment = Compartment, target = NewStation], G, _)
      goal(modify, unaddressed, 7118, "Apply fire suppressant",
        [compartment = Compartment], _, G)
    END RULE
  END RULE

  RULE 6740.no-personnel-fire.interpret.3 "No personnel report
       cannot be traced"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5120, "Fight fire in compartment",
           [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "fire"], miscellaneous-reports, _)

    CALL reassign-personnel(rule = "6740.no-personnel-fire", station
       = Station)
  END RULE
END GMO


;*****************************************************************

GMO 6740.flood
FOR ECL 6740 "Insufficient Personnel"
WHERE casualty = "flood"
LET source -> Station,
    compartment -> Compartment

  RULE 6740.no-personnel-flood.interpret.1 "No personnel for
       erroneous action"
  IF
    action(find, error-of-commission, 5140, "Dewater space",
           [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "flood"], A, _)
  END RULE

  RULE 6740.no-personnel-flood.interpret.2 "No personnel for non-
       error action"
  IF
    action(find, [correct, sub-optimal], 5130, "Dewater Space",
           [compartment = Compartment, target = Station], G, A)
  THEN
```

```
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "flood"], A, _)

    RULE 6740.no-personnel-flood.suggest.1 "No personnel requires
       alternate order"
    IF
      world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
                [compartment = Compartment, station = NewStation],
       _, _)
      AND goal(find, [unaddressed, addressed], 7218, "Remove Water",
       [compartment = Compartment], _, G)
    THEN
      action(create, pending, 5130, "Dewater Space", [compartment =
       Compartment, target = NewStation], G, _)
      goal(modify, unaddressed, 7218, "Remove Water", [compartment =
       Compartment], _, G)
    END RULE

    RULE 6740.no-personnel-flood.suggest.2 "No personnel requires
       alternate order"
    IF
      world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
                [compartment = Compartment, station = NewStation],
       _, _)
      AND goal(find, [unaddressed, addressed], 7130, "Clear Water
       from Firefighting",
           [compartment = Compartment], _, G)
    THEN
      action(create, pending, 5130, "Dewater Space", [compartment =
       Compartment, target = NewStation], G, _)
      goal(modify, unaddressed, 7130, "Clear Water from
       Firefighting", [compartment = Compartment], _, G)
    END RULE
  END RULE

  RULE 6740.no-personnel-flood.interpret.3 "No personnel report
       cannot be traced"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5130, "Dewater Space",
           [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "flood"], miscellaneous-reports, _)

    CALL reassign-personnel(rule = "6740.no-personnel-flood",
       station = Station)
  END RULE
END GMO


;*************************************************************
```

```
GMO 6740.smoke
FOR ECL 6740 "Insufficient Personnel"
WHERE casualty = "smoke"
LET source -> Station,
    compartment -> Compartment

  RULE 6740.no-personnel-smoke.interpret.1 "No personnel for
       erroneous action"
  IF
    action(find, error-of-commission, 5140, "Desmoke space",
           [compartment = Compartment, target = Station], _, A)
  THEN
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "smoke"], A, _)
  END RULE

  RULE 6740.no-personnel-smoke.interpret.2 "No personnel for non-
       error action"
  IF
    action(find, [correct, sub-optimal], 5140, "Desmoke Space",
           [compartment = Compartment, target = Station], G, A)
  THEN
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "smoke"], A, _)

    RULE 6740.no-personnel-smoke.suggest.1 "No personnel requires
       alternate order"
    IF
      world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = NewStation],
        _, _)
      AND goal(find, [unaddressed, addressed], 7318, "Remove Smoke",
        [compartment = Compartment], _, G)
    THEN
      action(create, pending, 5140, "Desmoke Space", [compartment =
        Compartment, target = NewStation], G, _)
      goal(modify, unaddressed, 7318, "Remove Smoke", [compartment =
        Compartment], _, G)
    END RULE

    RULE 6740.no-personnel-smoke.suggest.2 "No personnel requires
       alternate order"
    IF
      world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = NewStation],
        _, _)
      AND goal(find, [unaddressed, addressed], 7125, "Clear Smoke",
        [compartment = Compartment], _, G)
    THEN
      action(create, pending, 5140, "Desmoke Space", [compartment =
        Compartment, target = NewStation], G, _)
```

```
      goal(modify, unaddressed, 7125, "Clear Smoke", [compartment =
        Compartment], _, G)
    END RULE
  END RULE

  RULE 6740.no-personnel-smoke.interpret.3 "No personnel report
       cannot be traced"
  IF
    NOT action(find, [correct, sub-optimal, error-of-commission],
       5140, "Desmoke Space",
           [compartment = Compartment, target = Station], _, _)
  THEN
    report(create, _, 6740, "Insufficient Personnel",
           [compartment = Compartment, source = Station, casualty =
       "smoke"], miscellaneous-reports, _)

    CALL reassign-personnel(rule = "6740.no-personnel-smoke",
       station = Station)
  END RULE
END GMO

;*****************************************************************

GMO 6750
FOR ECL 6750 "DCCO Recommends Starting Fire Pump before Zebra"

  RULE 6750.recommend-start-pump.interpret.1 "DCCO recommendation
       suggests achieving GQ firemain config."
  IF
    goal(find, _, 7920, "Achieve GQ Firemain Configuration", [], _,
       G)
  THEN
    report(create, _, 6750, "DCCO Recommends Starting Fire Pump
       before Zebra", [], G, _)
  END RULE

  RULE 6750.recommend-start-pump.interpret.2 "DCCO recommendation
       untraceable"
  IF
    NOT goal(find, _, 7920, "Achieve GQ Firemain Configuration", [],
       _, _)
  THEN
    report(create, _, 6750, "DCCO Recommends Starting Fire Pump
       before Zebra", [], miscellaneous-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6801.fire
FOR ECL 6801 "Damage report fire/flood/smoke"
WHERE casualty = "fire"
LET source -> Station,
    compartment -> Compartment
```

```
RULE 6801.fire-report.interpret.1 "fire report overrides flood
     crisis"
IF
  crisis(find, unsolved, 8200, "Flood", [compartment =
     Compartment], _, C)
  AND goal(find, [unaddressed, addressed], 7200, "Control
     Flooding", [compartment = Compartment], _, G)
THEN
  crisis(modify, overridden, 8200, "Flood", [compartment =
     Compartment], _, C)
  goal(modify, overridden, 7200, "Control Flooding", [compartment
     = Compartment], _, G)
END RULE

RULE 6801.fire-report.interpret.2 "fire report overrides smoke
     crisis"
IF
  crisis(find, unsolved, 8300, "Smoke", [compartment =
     Compartment], _, C)
  AND goal(find, [unaddressed, addressed], 7300, "Control Smoke",
     [compartment = Compartment], _, G)
THEN
  crisis(modify, overridden, 8300, "Smoke", [compartment =
     Compartment], _, C)
  goal(modify, overridden, 7300, "Control Smoke", [compartment =
     Compartment], _, G)
END RULE

RULE 6801.fire-report.interpret.3 "fire report creates new crisis
     and goal tree"
IF
  NOT crisis(find, unsolved, 8100, "Fire", [compartment =
     Compartment], _, _)
THEN
  crisis(create, unsolved, 8100, "Fire", [compartment =
     Compartment], _, C)
  goal(create, unaddressed, 7100, "Control Fire", [compartment =
     Compartment], C, _)
END RULE

RULE 6801.fire-report.interpret.4 "associate new fire report with
     Identify Fire goal"
IF
  goal(find, _, 7105, "Identify Fire", [compartment =
     Compartment], _, G)
THEN
  report(create, _, 6801, "Damage report fire/flood/smoke",
         [compartment = Compartment, casualty = "fire", source =
     Station], G, _)
END RULE

RULE 6801.fire-report.interpret.5 "fire report satisfies identify
     fire goal"
IF
  goal(find, [unaddressed, addressed], 7105, "Identify Fire",
     [compartment = Compartment], _, GIF)
```

```
THEN
  goal(modify, satisfied, 7105, "Identify Fire", [compartment =
     Compartment], _, GIF)

  RULE 6801.fire-report.interpret.5.1 "check if isolation is
     already achieved - this satisfies isolate goal"
  IF
    goal(find, unaddressed, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, G)
    AND report(find, _, 6370, "Space Electrically and Mechanically
     Isolated", [compartment = Compartment], _, _)
  THEN
    goal(modify, satisfied, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, G)
  END RULE

  RULE 6801.fire-report.suggest.1 "if isolation has not already
     been achieved, propose it if it is possible"
  IF
    goal(find, unaddressed, 7116, "Isolate Compartment if
     Necessary", [compartment = Compartment], _, G)
    AND NOT report(find, _, 6370, "Space Electrically and
     Mechanically Isolated",
               [compartment = Compartment], _, _)
    AND world-state(find, _, 3360, "Can Isolate Compartment",
     [compartment = Compartment], _, _)
    AND world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
               [compartment = Compartment, station =
     RepairLocker], _, _)
  THEN
    action(create, pending, 5170, "Electrically and Mechanically
     Isolate Space",
          [compartment = Compartment, target = RepairLocker], G,
     _)
  END RULE

  RULE 6801.fire-report.suggest.2 "if active desmoking is
     necessary, propose it "
  IF
    goal(find, unaddressed, 7117, "Active Desmoke", [compartment =
     Compartment], _, G)
    AND world-state(find, _, 3361, "Should Active Desmoke
     Compartment", [compartment = Compartment], _, _)
    AND world-state(find, _, 4302, "Best Repair Locker for
     Compartment",
               [compartment = Compartment, station =
     RepairLocker], _, _)
  THEN
    action(create, pending, 5142, "Active Desmoke Space",
     [compartment = Compartment, target = RepairLocker], G, _)
  END RULE

  RULE 6801.fire-report.suggest.3 "set fire boundaries on
     compartment if necessary"
  IF
```

```
      goal(find, unaddressed, 7110, "Contain Fire", [compartment =
        Compartment], _, GB)
    THEN
      action(create, pending, 5114, "Set Boundaries on Compartment",
        [casualty = "fire", compartment = Compartment], GB, _)
    END RULE

    RULE 6801.fire-report.suggest.4
         "if there is no need to order active desmoking or
       isolation, then fight the fire"
    IF
      EITHER
        goal(find, satisfied, 7116,"Isolate Compartment if
       Necessary", [compartment = Compartment], _, _)
      OR
        NOT world-state(find, _, 3360, "Can Isolate Compartment",
       [compartment = Compartment], _, _)
      END EITHER
      AND EITHER
        goal(find, satisfied, 7117, "Active Desmoke", [compartment =
       Compartment], _, _)
      OR
        NOT world-state(find, _, 3361, "Should Active Desmoke
       Compartment", [compartment = Compartment], _, _)
      END EITHER
      AND goal(find, unaddressed, 7118, "Apply Fire Suppressant",
        [compartment = Compartment], _, GAFS)
      AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = RL], _, _)
      AND EITHER
        NOT world-state(find, _, 3304, "Compartment Has Halon",
       [compartment = Compartment], _, _)
      OR
        report(find, _, 6420, "Halon Ineffective", [compartment =
       Compartment], _, _)
      END EITHER
    THEN
      action(create, pending, 5120, "Fight fire in space",
        [compartment = Compartment, target = RL], GAFS, _)
    END RULE
  END RULE
END GMO


;*****************************************************************

GMO 6801.flood
FOR ECL 6801 "Damage report fire/flood/smoke"
WHERE casualty = "flood"
LET source -> Station,
    compartment -> Compartment

  RULE 6801.flood-report.interpret.1 "flood report creates new
       crisis and goal tree"
  IF
```

```
  NOT crisis(find, unsolved, 8100, "Fire", [compartment =
      Compartment], _, _)
  AND NOT crisis(find, unsolved, 8200, "Flood", [compartment =
      Compartment], _, _)
THEN
  crisis(create, unsolved, 8200, "Flood", [compartment =
      Compartment], _, C)
  goal(create, unaddressed, 7200, "Control Flooding", [compartment
      = Compartment], C, _)
END RULE

RULE 6801.flood-report.interpret.2 "associate new flood report
      with Identify Flooding goal"
IF
  goal(find, _, 7205, "Identify Flooding", [compartment =
      Compartment], _, G)
THEN
  report(create, _, 6801, "Damage report fire/flood/smoke",
          [compartment = Compartment, casualty = "flood", source =
      Station], G, _)
END RULE

RULE 6801.flood-report.interpret.3 "flood report satisfies
      identify flooding goal"
IF
  goal(find, [unaddressed, addressed], 7205, "Identify Flooding",
      [compartment = Compartment], _, GIF)
THEN
  goal(modify, satisfied, 7205, "Identify Flooding", [compartment
      = Compartment], _, GIF)

  RULE 6801.flood-report.interpret.4 "check if isolation is
      already achieved - this satisfies isoalte goal"
  IF
    goal(find, unaddressed, 7116, "Isolate Compartment",
      [compartment = Compartment], _, G)
    AND report(find, _, 6370, "Space Electrically and Mechanically
      Isolated", [compartment = Compartment], _, _)
  THEN
    goal(modify, satisfied, 7116, "Isolate Compartment",
      [compartment = Compartment], _, G)
  END RULE

  RULE 6801.flood-report.suggest.1 "if isolation has not already
      been achieved, propose it if it is possible"
  IF
    goal(find, unaddressed, 7116, "Isolate Compartment",
      [compartment = Compartment], _, G)
    AND NOT report(find, _, 6370, "Space Electrically and
      Mechanically Isolated",
                [compartment = Compartment], _, _)
    AND world-state(find, _, 3360, "Can Isolate Compartment",
      [compartment = Compartment], _, _)
    AND world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
```

```
                [compartment = Compartment, station = Station], _,
      _)
    THEN
      action(create, pending, 5170, "Electrically and Mechanically
       Isolate Space",
            [compartment = Compartment, target = Station], G, _)
    END RULE

    RULE 6801.flood-report.suggest.2 "if there is no need to order
       isolation, then dewater"
    IF
      EITHER
        goal(find, satisfied, 7116,"Isolate Compartment",
        [compartment = Compartment], _, _)
      OR
        NOT world-state(find, _, 3360, "Can Isolate Compartment",
        [compartment = Compartment], _, _)
      END EITHER
      AND goal(find, unaddressed, 7218, "Remove Water", [compartment
       = Compartment], _, GRW)
      AND world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
                [compartment = Compartment, station = RL], _, _)
    THEN
      action(create, pending, 5130, "Dewater Space", [compartment =
       Compartment, target = RL], GRW, _)
    END RULE
  END RULE
END GMO


;*****************************************************************

GMO 6801.smoke
FOR ECL 6801 "Damage report fire/flood/smoke"
WHERE casualty = "smoke"
LET source -> Station,
    compartment -> Compartment

  RULE 6801.smoke-report.interpret.1 "smoke report creates new
       crisis and goal tree"
  IF
    NOT crisis(find, unsolved, 8100, "Fire", [compartment =
       Compartment], _, _)
    AND NOT crisis(find, unsolved, 8300, "Smoke", [compartment =
       Compartment], _, _)
  THEN
    crisis(create, unsolved, 8300, "Smoke", [compartment =
       Compartment], _, C)
    goal(create, unaddressed, 7300, "Control Smoke", [compartment =
       Compartment], C, _)
  END RULE

  RULE 6801.smoke-report.interpret.2 "associate new flood report
       with Identify Flooding goal"
  IF
```

```
  goal(find, _, 7305, "Identify Smoke", [compartment =
      Compartment], _, G)
THEN
  report(create, _, 6801, "Damage report fire/flood/smoke",
         [compartment = Compartment, casualty = "smoke", source =
      Station], G, _)
END RULE

RULE 6801.smoke-report.interpret.3 "smoke report satisfies
      identify smoke goal"
IF
  goal(find, [unaddressed, addressed], 7305, "Identify Smoke",
      [compartment = Compartment], _, GIF)
THEN
  goal(modify, satisfied, 7305, "Identify Smoke", [compartment =
      Compartment], _, GIF)

  RULE 6801.smoke-report.interpret.4 "check if isolation is
      already achieved - this satisfies isoalte goal"
  IF
    goal(find, unaddressed, 7116, "Isolate Compartment",
      [compartment = Compartment], _, G)
    AND report(find, _, 6370, "Space Electrically and Mechanically
      Isolated", [compartment = Compartment], _, _)
  THEN
    goal(modify, satisfied, 7116, "Isolate Compartment",
      [compartment = Compartment], _, G)
  END RULE

  RULE 6801.smoke-report.suggest.1 "if isolation has not already
      been achieved, propose it if it is possible"
  IF
    goal(find, unaddressed, 7116, "Isolate Compartment",
      [compartment = Compartment], _, G)
    AND NOT report(find, _, 6370, "Space Electrically and
      Mechanically Isolated",
                [compartment = Compartment], _, _)
    AND world-state(find, _, 3360, "Can Isolate Compartment",
      [compartment = Compartment], _, _)
    AND world-state(find, _, 4302, "Best Repair Locker for
      Compartment",
                [compartment = Compartment, station = Station], _,
      _)
  THEN
    action(create, pending, 5170, "Electrically and Mechanically
      Isolate Space",
           [compartment = Compartment, target = Station], G, _)
  END RULE

  RULE 6801.smoke-report.suggest.2 "if there is no need to order
      isolation, then desmoke"
  IF
    EITHER
      goal(find, satisfied, 7116,"Isolate Compartment",
      [compartment = Compartment], _, _)
    OR
```

```
        NOT world-state(find, _, 3360, "Can Isolate Compartment",
        [compartment = Compartment], _, _)
      END EITHER
      AND goal(find, unaddressed, 7318, "Remove Smoke", [compartment
        = Compartment], _, GRW)
      AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station = RL], _, _)
    THEN
      action(create, pending, 5140, "Desmoke Space", [compartment =
        Compartment, target = RL], GRW, _)
    END RULE
  END RULE
END GMO


;*************************************************************

GMO 6804
FOR ECL 6804 "Firemain or Chillwater Rupture"
LET loop -> Loop,
    compartment -> Compartment,
    source -> Station

  RULE 6804.pipe-rupture.interpret.1 "create new crisis and goals if
       necessary"
  IF
    NOT crisis(find, unsolved, 8500, "Pipe Rupture", [system = Loop,
       compartment = Compartment], _, _)
  THEN
    crisis(create, unsolved, 8500, "Pipe Rupture",
           [system = Loop, compartment = Compartment, source =
       Station], _, C)
    goal(create, unaddressed, 7500, "Patch Pipe Rupture", [system =
       Loop, compartment = Compartment], C, G)

    RULE 6804.pipe-rupture.suggest.1 "if the rupture can be
       isolated, isolate and patch"
    IF
      world-state(find, _, 3121, "Pipe Should Be Isolated",
        [compartment = Compartment], _, _)
      AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station =
       RepairLocker], _, _)
      AND world-state(find, _, 4121, "Valves Needed to Isolate",
        [compartment = Compartment, valves = Valves], _, _)
    THEN
      action(create, pending, 5162, "Isolate and Patch Firemain or
        Chillwater Rupture",
             [loop = Loop, compartment = Compartment, target =
       RepairLocker, valve = Valves], G, _)
    END RULE

    RULE 6804.pipe-rupture.suggest.2 "if the rupture cannot be
       isolated, patch in place"
```

```
    IF
      NOT world-state(find, _, 3121, "Pipe Should Be Isolated",
        [compartment = Compartment], _, _)
      AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station =
        RepairLocker], _, _)
    THEN
      action(create, pending, 5166, " Patch Firemain or Chillwater
        Rupture in Place",
              [loop = Loop, compartment = Compartment, target =
        RepairLocker], G, _)
    END RULE
  END RULE
END GMO


;***************************************************************

GMO 6808
FOR ECL 6808 "Hull Rupture"
LET compartment -> Compartment,
    source -> Station

  RULE 6808.hull-rupture.interpret.1 "create new crisis and goals if
       necessary"
  IF
    NOT crisis(find, unsolved, 8400, "Hull Rupture", [compartment =
       Compartment], _, _)
  THEN
    crisis(create, unsolved, 8400, "Hull Rupture",
           [source = Station, compartment = Compartment], scenario-
       node, C)
    goal(create, unaddressed, 7400, "Patch Hull Rupture",
       [compartment = Compartment], C, G)

    RULE 6808.hull-rupture.suggest.1 "get personnel to patch the
       rupture"
    IF
      world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                  [compartment = Compartment, station =
        RepairLocker], _, _)
    THEN
      action(create, pending, 5176, "Patch Hull Rupture",
             [compartment = Compartment, target = RepairLocker], G,
        _)
    END RULE
  END RULE
END GMO


;***************************************************************

GMO 6820.fire
FOR ECL 6820 "Alarm in compartment from DCCO"
```

```
WHERE alarm = "fire"
LET compartment -> Compartment,
    source -> Station

  RULE 6820.fire-alarm.interpret.1 "fire alarm overrides flood
       crisis"
  IF
    crisis(find, unsolved, 8200, "Flood", [compartment =
       Compartment], _, C)
    AND goal(find, [unaddressed, addressed], 7200, "Control
       Flooding", [compartment = Compartment], _, G)
  THEN
    crisis(modify, overridden, 8200, "Flood", [compartment =
       Compartment], _, C)
    goal(modify, overridden, 7200, "Control Flooding", [compartment
       = Compartment], _, G)
  END RULE

  RULE 6820.fire-alarm.interpret.2 "fire alarm overrides smoke
       crisis"
  IF
    crisis(find, unsolved, 8300, "Smoke", [compartment =
       Compartment], _, C)
    AND goal(find, [unaddressed, addressed], 7300, "Control Smoke",
       [compartment = Compartment], _, G)
  THEN
    crisis(modify, overridden, 8300, "Smoke", [compartment =
       Compartment], _, C)
    goal(modify, overridden, 7300, "Control Smoke", [compartment =
       Compartment], _, G)
  END RULE

  RULE 6820.fire-alarm.interpret.3 "Fire alarm triggers new crisis"
  IF
    NOT crisis(find, unsolved, 8100, "Fire", [compartment =
       Compartment], _, _)
  THEN
    crisis(create, unsolved, 8100, "Fire", [compartment =
       Compartment], scenario-node, C)
    goal(create, unaddressed, 7100, "Control Fire", [compartment =
       Compartment], C, _)

    RULE 6820.fire-alarm.suggest.1 "fire alarm may require
       investigation"
    IF
      goal(find, unaddressed, 7105, "Identify Fire", [compartment =
       Compartment], _, G)
      AND NOT world-state(find, _, 3303, "Compartment is Manned",
       [compartment = Compartment], _, _)
      AND world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
                 [compartment = Compartment, station =
       RepairLocker], _, _)
    THEN
      action(create, pending, 5105, "Investigate Compartment",
```

```
            [compartment = Compartment, target = RepairLocker], G,
        _)
    END RULE
  END RULE

  RULE 6820.fire-alarm.interpret.4 "fire alarm report attached to
       identify fire goal"
  IF
    goal(find, _, 7105, "Identify Fire", [compartment =
        Compartment], _, G)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
        "fire"], G, _)
  END RULE
END GMO

;*****************************************************************

GMO 6820.flood
FOR ECL 6820 "Alarm in compartment from DCCO"
WHERE alarm = "flood"
LET compartment -> Compartment,
    source -> Station

  RULE 6820.flood-alarm.interpret.1 "flood alarm generates new flood
       crisis"
  IF
    NOT crisis(find, unsolved, 8100, "Fire", [compartment =
        Compartment], _, _)
    AND NOT crisis(find, unsolved, 8200, "Flood", [compartment =
        Compartment], _, _)
  THEN
    crisis(create, unsolved, 8200, "Flood", [compartment =
        Compartment], _, C)
    goal(create, unaddressed, 7200, "Control Flooding", [compartment
        = Compartment], C, _)

    RULE 6820.flood-alarm.suggest.1 "flood alarm may require
        investigation"
    IF
      goal(find, unaddressed, 7205, "Identify Flooding",
        [compartment = Compartment], _, G)
      AND NOT world-state(find, _, 3303, "Compartment is Manned",
        [compartment = Compartment], _, _)
      AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                 [compartment = Compartment, station =
        RepairLocker], _, _)
    THEN
      action(create, pending, 5105, "Investigate Compartment",
             [compartment = Compartment, target = RepairLocker], G,
        _)
    END RULE
  END RULE
```

```
  RULE 6820.flood-alarm.interpret.2 "flood alarm associated with
       identify flood goal"
  IF
    goal(find, [unaddressed, addressed, satisfied], 7205, "Identify
       Flooding",
         [compartment = Compartment], _, G)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
       "flood"], G, _)
  END RULE

  RULE 6820.flood-alarm.interpret.3 "flood alarm associated with
       clear water from firefighting goal"
  IF
    NOT goal(find, [unaddressed, addressed, satisfied], 7205,
       "Identify Flooding",
         [compartment = Compartment], _, _)
    AND goal(find, [unaddressed, addressed, satisfied], 7130, "Clear
       water from firefighting",
         [compartment = Compartment], _, G)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
       "flood"], G, _)
  END RULE

  RULE 6820.flood-alarm.interpret.4 "flood alarm cannot be
       associated with an existing goal"
  IF
    NOT goal(find, [unaddressed, addressed, satisfied], 7205,
       "Identify Flooding",
         [compartment = Compartment], _, _)
    AND NOT goal(find, [unaddressed, addressed, satisfied], 7130,
       "Clear water from firefighting",
         [compartment = Compartment], _, _)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
       "flood"], miscellaneous-reports, _)
  END RULE
END GMO

;*************************************************************

GMO 6820.halon
FOR ECL 6820 "Alarm in compartment from DCCO"
WHERE alarm = "halon"
LET compartment -> Compartment,
    source -> Station

  RULE 6820.halon-alarm.interpret.1 "halon alarm means goal of
       applying fire suppressant addressed"
  IF
    goal(find, [unaddressed, addressed], 7118, "Apply Fire
       Suppressant", [compartment = Compartment], _, G)
```

```
  THEN
    goal(modify, addressed, 7118, "Apply Fire Suppressant",
        [compartment = Compartment], _, G)
    report(create, _, 6820, "Alarm in Compartment from DCCO",
            [source = Station, compartment = Compartment, alarm =
        "halon"], G, _)
  END RULE

  RULE 6820.halon-alarm.interpret.2 "halon alarm not associated with
       any particular goal"
  IF
    NOT goal(find, [unaddressed, addressed], 7118, "Apply Fire
       Suppressant", [compartment = Compartment], _, _)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
            [source = Station, compartment = Compartment, alarm =
        "halon"], miscellaneous-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6820.magazine
FOR ECL 6820 "Alarm in compartment from DCCO"
WHERE alarm = "magazine"
LET compartment -> Compartment,
    source -> Station

  RULE 6820.magazine-alarm.interpret.1 "Magazine Alarm generates new
       crisis, goals"
  IF
    NOT crisis(find, unsolved, 8150, "Hot Magazines", [compartment =
       Compartment], _, _)
  THEN
    crisis(create, unsolved, 8150, "Hot Magazines", [compartment =
       Compartment], _, C)
    goal(create, unaddressed, 7150, "Protect Magazines",
       [compartment = Compartment], G, _)

    RULE 6820.magazine-alarm.suggest.1 "Try to get permission to
       flood"
    IF
      goal(find, unaddressed, 7155, "Try to get permission to flood
       magazines",
           [compartment = Compartment], _, GGPF)
      AND world-state(find, _, 4302, "Best Repair Locker for
       Compartment",
                [compartment = Compartment, station =
       RepairLocker], _, _)
    THEN
      action(create, pending, 5101, "Flood Magazines",
             [compartment = Compartment, target = RepairLocker],
       GGPF, _)
    END RULE
  END RULE
```

```
  RULE 6820.magazine-alarm.suggest.2 "Associate mag alarm report
       with protect magazines goal"
  IF
    goal(find, [addressed, unaddressed], 7150, "Protect Magazines",
        [compartment = Compartment], _, G)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
        "halon"], G, _)
  END RULE

  RULE 6820.magazine-alarm.suggest.3 "Cannot associate magazine
       alarm with existing goal"
  IF
    NOT goal(find, [addressed, unaddressed], 7150, "Protect
        Magazines", [compartment = Compartment], _, _)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
        "halon"], miscellaneous-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6820.smoke
FOR ECL 6820 "Alarm in compartment from DCCO"
WHERE alarm = "smoke"
LET compartment -> Compartment,
    source -> Station

  RULE 6820.smoke-alarm.interpret.1 "smoke alarm generates new smoke
       crisis"
  IF
    NOT crisis(find, unsolved, 8100, "Fire", [compartment =
        Compartment], _, _)
    AND NOT crisis(find, unsolved, 8300, "Smoke", [compartment =
        Compartment], _, _)
  THEN
    crisis(create, unsolved, 8300, "Smoke", [compartment =
        Compartment], _, C)
    goal(create, unaddressed, 7300, "Control Smoke", [compartment =
        Compartment], C, _)

    RULE 6820.smoke-alarm.suggest.1 "smoke alarm may require
        investigation"
    IF
      goal(find, unaddressed, 7305, "Identify Smoke", [compartment =
        Compartment], _, G)
      AND NOT world-state(find, _, 3303, "Compartment is Manned",
        [compartment = Compartment], _, _)
      AND world-state(find, _, 4302, "Best Repair Locker for
        Compartment",
                 [compartment = Compartment, station =
        RepairLocker], _, _)
    THEN
```

```
      action(create, pending, 5105, "Investigate Compartment",
             [compartment = Compartment, target = RepairLocker], G,
      _)
    END RULE
  END RULE

  RULE 6820.smoke-alarm.interpret.2 "flood alarm associated with
       identify flood goal"
  IF
    goal(find, [unaddressed, addressed, satisfied], 7305, "Identify
       Smoke",
         [compartment = Compartment], _, G)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
       "smoke"], G, _)
  END RULE

  RULE 6820.smoke-alarm.interpret.3 "flood alarm associated with
       clear water from firefighting goal"
  IF
    NOT goal(find, [unaddressed, addressed, satisfied], 7305,
       "Identify Smoke",
         [compartment = Compartment], _, _)
    AND goal(find, [unaddressed, addressed, satisfied], 7125, "Clear
       Smoke",
         [compartment = Compartment], _, G)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
       "smoke"], G, _)
  END RULE

  RULE 6820.smoke-alarm.interpret.4 "flood alarm cannot be
       associated with an existing goal"
  IF
    NOT goal(find, [unaddressed, addressed, satisfied], 7305,
       "Identify Smoke",
         [compartment = Compartment], _, _)
    AND NOT goal(find, [unaddressed, addressed, satisfied], 7125,
       "Clear Smoke",
         [compartment = Compartment], _, _)
  THEN
    report(create, _, 6820, "Alarm in Compartment from DCCO",
           [source = Station, compartment = Compartment, alarm =
       "smoke"], miscellaneous-reports, _)
  END RULE
END GMO


;*************************************************************

GMO 6824
FOR ECL 6824 "Firemain Pressure Normal from DCCO"
LET loop -> Loop
```

```
  RULE 6824.firemain-pressure.interpret.1 "normal pressure satisfies
       restore pressure goal"
  IF
    goal(find, _, 7550, "Handle Low Firemain Pressure", [loop =
       Loop], _, G)
  THEN
    goal(modify, satisfied, 7550, "Handle Low Firemain Pressure",
       [loop = Loop], _, G)
    report(create, _, 6824, "Firemain Pressure Normal from DCCO",
       [loop = Loop], G, _)
  END RULE

  RULE 6824.firemain-pressure.interpret.2 "normal pressure report
       untraceable"
  IF
    NOT goal(find, _, 7550, "Handle Low Firemain Pressure", [loop =
       Loop], _, _)
  THEN
    report(create, _, 6824, "Firemain Pressure Normal from DCCO",
       [loop = Loop], miscellaneous-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6826
FOR ECL 6826 "Firemain Pressure Low from DCCO"
LET loop -> Loop

  RULE 6826.firemain-pressure.interpret.1 "If there is no existing
       crisis, create one"
  IF
    NOT crisis(find, unsolved, 8550, "Low Firemain Pressure", [loop
       = Loop], _, _)
  THEN
    crisis(create, unsolved, 8550, "Low Firemain Pressure", [loop =
       Loop], _, C)
    goal(create, unaddressed, 7550, "Handle Low Firemain Pressure",
       [loop = Loop], C, _)
  END RULE

  RULE 6826.firemain-pressure.suggest.1
    "If the low pressure is not explained by the current set of
       reports, suggest investigating the firemain"
  IF
    NOT world-state(find, _, 3150, "Current Reports Explain Low
       Firemain Pressure", [loop = Loop], _, _)
    AND goal(find, [unaddressed, addressed], 7555, "Investigate
       Firemain", [loop = Loop], _, G)
    AND NOT action(find, pending, 5107, "Investigate Firemain or
       Chillwater", [loop = Loop], _, _)
    AND NOT action(find, pending, 5108, "Investigate Firemain or
       Chillwater in Space", [loop = Loop], _, _)
    AND world-state(find, _, 4111, "Best Repair Locker for Firemain
       Loop",
              [loop = Loop, station = RepairLocker], _, _)
```

```
  THEN
    goal(modify, unaddressed, 7555, "Investigate Firemain", [loop =
        Loop], _, G)
    action(create, pending, 5107, "Investigate Firemain or
        Chillwater",
            [loop = Loop, target = RepairLocker], G, _)
  END RULE

  RULE 6826.firemain-pressure.suggest.2 "create any new firemain
       orders that are necessary to restore pressure"
  IF
    goal(find, [unaddressed, addressed], 7560, "Restore Firemain
        Pressure", [loop = Loop], _, G)
  THEN
    goal(modify, unaddressed, 7560, "Restore Firemain Pressure",
        [loop = Loop], _, G)
    CALL get-new-firemain-orders(gmo-name = "6826.firemain-
        pressure", goal = G, loop = Loop)
  END RULE
END GMO

;*****************************************************************

GMO 6830
FOR ECL 6830 "Valve Status Report from DCCO or Repair Lockers"
LET valve -> Valve,
    valve_status -> ValveStatus,
    source -> Source

  RULE 6830.valve-status.interpret.1 "note valve status"
  IF TRUE
  THEN
    report(create, _, 6830, "Valve Status Report from DCCO or Repair
        Lockers",
            [source = Source, valve = Valve, valve_status =
        ValveStatus], untraceable_reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6832
FOR ECL 6832 "Pump Status Report from DCCO or Repair Lockers"
LET pump -> Pump,
    pump_status -> PumpStatus,
    source -> Source

  RULE 6832.pump-status.interpret.1 "note valve status"
  IF TRUE
  THEN
    report(create, _, 6832, "Pump Status Report from DCCO or Repair
        Lockers",
            [source = Source, pump = Pump, pump_status = PumpStatus],
        untraceable_reports, _)
  END RULE
END GMO
```

```
;*****************************************************************

GMO 6840
FOR ECL 6840 "Halon Activated"
LET compartment -> Compartment

  RULE 6840.halon.interpret.1 "Halon activated addressed goal of
       applying fire suppressant"
  IF
    goal(find, [unaddressed, addressed], 7118, "Apply Fire
       Suppressant", [compartment = Compartment], _, G)
  THEN
    goal(modify, addressed, 7118, "Apply Fire Suppressant",
       [compartment = Compartment], _, G)
    report(create, _, 6840, "Halon activated addressed goal of
       applying fire suppressant",
          [compartment = Compartment], G, _)
  END RULE

  RULE 6840.halon.interpret.2 "Halon activated report untraceable"
  IF
    NOT goal(find, [unaddressed, addressed], 7118, "Apply Fire
       Suppressant", [compartment = Compartment], _, _)
  THEN
    report(create, _, 6840, "Halon activated addressed goal of
       applying fire suppressant",
          [compartment = Compartment], miscellaneous-reports, _)
  END RULE
END GMO

;*****************************************************************

GMO 6903
FOR ECL 6903 "Manned and ready, Zebra set throughout the ship
       announcement to ship"

  RULE 6903.mrz-set.interpret.1 "associate with get ship to MRZ
       goal"
  IF
    goal(find, _, 7900, "Get ship to Manned, Ready, Zebra Set", [],
       _, G)
  THEN
    report(create, _, 6903, "Manned and ready, Zebra set throughout
       the ship announcement to ship", [], G, _)
  END RULE

  RULE 6903.mrz-set.interpret.2 "no goal to associate MRZ shipwide
       report with"
  IF
    NOT goal(find, _, 7900, "Get ship to Manned, Ready, Zebra Set",
       [], _, _)
  THEN
    report(create, _, 6903, "Manned and ready, Zebra set throughout
       the ship announcement to ship", [],
          miscellaneous-reports, _)
```

```
  END RULE
END GMO

;*****************************************************************

GMO 6980
FOR ECL 6980 "Missile or Mine Sighted from Bridge to DCA and CO
       through 1MC"
LET weapon -> Weapon,
    side -> Side

  RULE 6980.missile-or-mine.interpret.1 "register top-level scenario
       event"
  IF TRUE
  THEN
    report(create, _, 6980, "Missile or Mine Sighted from Bridge to
       DCA and CO through 1MC",
           [weapon = Weapon, side = Side], scenario-node, _)
  END RULE
END GMO

;*****************************************************************

GMO 6990
FOR ECL 6990 "General Quarters"

  RULE 6990.gq.interpret.1 "create GQ report and goals"
  IF TRUE
  THEN
    report(create, _, 6990, "General Quarters", [], scenario-node,
       GQReport)
    goal(create, unaddressed, 7900, "Get Ship to Manned, Ready,
       Zebra Set", [], GQReport, _)
  END RULE

  RULE 6990.gq.interpret.2 "if starboard pump is on, mark
       corresponding goal satisfied"
  IF
    world-state(find, _, 3102, "Starboard Fire Pump is On", [], _,
       _)
    AND goal(find, [unaddressed, addressed], 7922, "Achieve Minimal
       Starboard Firemain Pressure", [], _, G)
  THEN
    goal(modify, satisfied, 7922, "Achieve Minimal Starboard
       Firemain Pressure", [], _, G)
  END RULE

  RULE 6990.gq.suggest.1 "if starboard pump is not on, suggest
       requesting to start one"
  IF
    NOT world-state(find, _, 3102, "Starboard Fire Pump is On", [],
       _, _)
    AND goal(find, [unaddressed, addressed], 7922, "Achieve Minimal
       Starboard Firemain Pressure", [], _, G)
    AND world-state(find, _, 4102, "Get Functional Starboard Fire
       Pump", [pump = Pump], _, _)
```

```
  THEN
    action(create, pending, 5520, "Request Permission to Start Fire
        Pump",
            [target = "EOOW", pump = Pump], G, _)
  END RULE

  RULE 6990.gq.interpret.3 "if port pump is on, mark corresponding
       goal satisfied"
  IF
    world-state(find, _, 3101, "Port Fire Pump is On", [], _, _)
    AND goal(find, [unaddressed, addressed], 7921, "Achieve Minimal
        Port-Side Firemain Pressure", [], _, G)
  THEN
    goal(modify, satisfied, 7921, "Achieve Minimal Port-Side
        Firemain Pressure", [], _, G)
  END RULE

  RULE 6990.gq.suggest.2 "if port pump is not on, suggest requesting
       to start one"
  IF
    NOT world-state(find, _, 3101, "Port-Side Fire Pump is On", [],
        _, _)
    AND goal(find, [unaddressed, addressed], 7921, "Achieve Minimal
        Port-Side Firemain Pressure", [], _, G)
    AND world-state(find, _, 4101, "Get Functional Port-Side Fire
        Pump", [pump = Pump], _, _)
  THEN
    action(create, pending, 5520, "Request Permission to Start Fire
        Pump",
            [target = "EOOW", pump = Pump], G, _)
  END RULE
END GMO
```