

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

DECISION MODEL IMPLEMENTATION IN THE GLOBAL INFORMATION NETWORK ARCHITECTURE

by

Matthew K. Sweeney

September 2022

Thesis Advisor: Armon C. Barton
Co-Advisors: Gurminder Singh
Thomas S. Anderson

Approved for public release. Distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington, DC, 20503.

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2022	**3. REPORT TYPE AND DATES COVERED** Master's thesis

4. TITLE AND SUBTITLE DECISION MODEL IMPLEMENTATION IN THE GLOBAL INFORMATION NETWORK ARCHITECTURE	**5. FUNDING NUMBERS**
6. AUTHOR(S) Matthew K. Sweeney	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING / MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release. Distribution is unlimited.	**12b. DISTRIBUTION CODE** A

13. ABSTRACT (maximum 200 words)

The future battlefield is one that will be heavily influenced by rapidly changing technological capabilities of near-peer adversaries. Success in this environment will require simple to use systems that are adaptable to situations and capable of integration with other forces and systems. The Multi-Domain Operations Command, Control, Computers, Communications, Combat Systems, and Intelligence (MDOC5i) aims to prepare the Marine Corps for the future battlefield. Due to certain drawback of traditional machine learning techniques, MDOC5i uses vector relational data modeling (VRDM) to provide the Marine Corps with systems suitable for dynamic employment. MDOC5i uses the Global Information Network Architecture (GINA) as its VRDM platform. This research uses GINA to create a ubiquitous decision model that can be configured to USMC scenarios. The research implements the ubiquitous model and proves functionality through a network analysis use case. This decision model will serve as the base model for all GINA implementations. The ability to quickly construct and adapt GINA models based on scenarios and integrate those models into a common framework will provide the Marine Corps with information overmatch against future adversaries.

14. SUBJECT TERMS Global Information Network Architecture, GINA, tactical networks, decision models, vector-relational data modeling, VRDM, Multi-Domain Operations Command Control Computers Communications Combat Systems and Intelligence, MDOC5i	**15. NUMBER OF PAGES** 149
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release. Distribution is unlimited.

DECISION MODEL IMPLEMENTATION IN THE GLOBAL INFORMATION NETWORK ARCHITECTURE

Matthew K. Sweeney
Captain, United States Marine Corps
BS, United States Naval Academy, 2015

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
September 2022

Approved by: Armon C. Barton
Advisor

Gurminder Singh
Co-Advisor

Thomas S. Anderson
Co-Advisor

Gurminder Singh
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The future battlefield is one that will be heavily influenced by rapidly changing technological capabilities of near-peer adversaries. Success in this environment will require simple to use systems that are adaptable to situations and capable of integration with other forces and systems. The Multi-Domain Operations Command, Control, Computers, Communications, Combat Systems, and Intelligence (MDOC5i) aims to prepare the Marine Corps for the future battlefield. Due to certain drawback of traditional machine learning techniques, MDOC5i uses vector relational data modeling (VRDM) to provide the Marine Corps with systems suitable for dynamic employment. MDOC5i uses the Global Information Network Architecture (GINA) as its VRDM platform. This research uses GINA to create a ubiquitous decision model that can be configured to USMC scenarios. The research implements the ubiquitous model and proves functionality through a network analysis use case. This decision model will serve as the base model for all GINA implementations. The ability to quickly construct and adapt GINA models based on scenarios and integrate those models into a common framework will provide the Marine Corps with information overmatch against future adversaries.

THIS PAGE INTENTIONALLY LEFT BLANK

Table of Contents

1 Introduction **1**

1.1 MDOC5i 2

1.2 MDOC5i Involvement in the Marine Corps 3

1.3 Thesis Focus and MDOC5i Tie-in. 4

1.4 Hypothesis and Research Questions 5

1.5 Tools Used 6

2 Background Information **9**

2.1 Limitations of Machine Learning and Deep Learning 9

2.2 Vector Relational Data Modeling 11

2.3 Global Information Network Architecture. 12

2.4 Recommender Model 21

3 Methodology **25**

3.1 Preliminary Work 25

3.2 Thesis Focus 33

3.3 Methodologies 33

4 Implementation **49**

4.1 Model Creation 49

4.2 Decision Implementation 80

4.3 Input Channel Implementation 100

5 Testing and Results **111**

5.1 Testing 111

5.2 Results 114

6 Conclusion and Future Work **121**

6.1 Conclusion 121
6.2 Extensive Military Use Cases 122
6.3 Future Work 126

List of References **129**

Initial Distribution List **131**

List of Figures

Figure 2.1 Global Information Network Architecture (GINA) Visual Construct 19

Figure 2.2 Recommender Model: Recommendation Domain 23

Figure 3.1 Dark Stax and GINA Integration 26

Figure 3.2 Overview of the Graph Model 28

Figure 3.3 Hypergraph of the Decider Information 31

Figure 3.4 Design Methodology 34

Figure 3.5 Decision Model Design 37

Figure 3.6 GINA Framework 39

Figure 3.7 GINA VRDM Application Model 40

Figure 3.8 Base GINA Construct Represented in Unified Modeling Language (UML) Format. 42

Figure 3.9 Full UML Depiction of the Decision Model 44

Figure 3.10 Proposed Test Scenario 46

Figure 4.1 Decision Model Design 50

Figure 4.2 Concept Implementation: Step-by-Step Instructions Photo 1 . . . 52

Figure 4.3 Concept Implementation: Step-by-Step Instructions Photo 2 . . . 53

Figure 4.4 Concept Implementation: Step-by-Step Instructions Photo 3 . . . 54

Figure 4.5 Concept Implementation: Step-by-Step Instructions Photo 4 . . . 55

Figure 4.6 Concept Implementation: Step-by-Step Instructions Photo 5 . . . 55

Figure 4.7 Concept Implementation: Step-by-Step Instructions Photo 6 . . . 56

Figure 4.8 Concept Implementation: Step-by-Step Instructions Photo 7 . . . 57

Figure 4.9 Concept Implementation: Step-by-Step Instructions Photo 8 . . . 58
Figure 4.10 Concept Implementation: Step-by-Step Instructions Photo 9 . . . 58
Figure 4.11 Concept Implementation: Step-by-Step Instructions Photo 10 . . 59
Figure 4.12 Concept Implementation: Step-by-Step Instructions Photo 11 . . 60
Figure 4.13 Concept Implementation: Step-by-Step Instructions Photo 12 . . 61
Figure 4.14 Concept Implementation: Step-by-Step Instructions Photo 13 . . 61
Figure 4.15 Concept Implementation: Step-by-Step Instructions Photo 14 . . 62
Figure 4.16 Concept Implementation: Step-by-Step Instructions Photo 15 . . 63
Figure 4.17 Concept Implementation: Step-by-Step Instructions Photo 16 . . 63
Figure 4.18 Concept Implementation: Step-by-Step Instructions Photo 17 . . 64
Figure 4.19 Concept Implementation: Step-by-Step Instructions Photo 18 . . 65
Figure 4.20 Concept Implementation: Step-by-Step Instructions Photo 19 . . 66
Figure 4.21 Concept Implementation: Step-by-Step Instructions Photo 20 . . 66
Figure 4.22 Concept Implementation: Step-by-Step Instructions Photo 21 . . 67
Figure 4.23 Concept Implementation: Step-by-Step Instructions Photo 22 . . 68
Figure 4.24 Concept Implementation: Step-by-Step Instructions Photo 33 . . 69
Figure 4.25 Concept Implementation: Step-by-Step Instructions Photo 34 . . 70
Figure 4.26 Concept Implementation: Step-by-Step Instructions Photo 35 . . 71
Figure 4.27 Concept Implementation: Step-by-Step Instructions Photo 23 . . 72
Figure 4.28 Concept Implementation: Step-by-Step Instructions Photo 24 . . 73
Figure 4.29 Concept Implementation: Step-by-Step Instructions Photo 25 . . 74
Figure 4.30 Concept Implementation: Step-by-Step Instructions Photo 26 . . 75
Figure 4.31 Concept Implementation: Step-by-Step Instructions Photo 27 . . 76

Figure 4.32 Concept Implementation: Step-by-Step Instructions Photo 28 . . 77
Figure 4.33 Concept Implementation: Step-by-Step Instructions Photo 29 . . 77
Figure 4.34 Concept Implementation: Step-by-Step Instructions Photo 30 . . 78
Figure 4.35 Concept Implementation: Step-by-Step Instructions Photo 31 . . 79
Figure 4.36 Concept Implementation: Step-by-Step Instructions Photo 32 . . 80
Figure 4.37 Decision Configuration - Payload Elements 82
Figure 4.38 Decision Configuration - Payload Element Fields 83
Figure 4.39 Decision Configuration - Concept Page Functionality 84
Figure 4.40 Dark Stax XMPP Concept Related to Design Methodology 85
Figure 4.41 Decision Configuration - Payload Elements in Mission Objective State 87
Figure 4.42 Decision Configuration - Payload Element Fields in Mission Objective State 88
Figure 4.43 Decision Configuration - Mission Objective State Concept Page Functionality 89
Figure 4.44 Mission Objective State in Terms of Design 90
Figure 4.45 Mission Objective Status Concept Page 91
Figure 4.46 Decision Configuration - Objective Payload State Vector 92
Figure 4.47 Decision Configuration - Objective Payload State Vector References 92
Figure 4.48 Decision Configuration - Objective State Element 94
Figure 4.49 Objective Payload State in Terms of Design 95
Figure 4.50 Decision Configuration - mObjectiveStatusNone Vector 96
Figure 4.51 Decision Configuration - mObjectiveStatusNone Vector References 97
Figure 4.52 Decision Configuration - Linked Action Access 98
Figure 4.53 Decision Configuration - Creation of the Linked Action 99

Figure 4.54 Action and Linked Action in Terms of Design 100
Figure 4.55 Input Channel Implementation - Components and Channels Relation 102
Figure 4.56 Input Channel Implementation - Content Format 103
Figure 4.57 Input Channel Implementation - Format Element 104
Figure 4.58 Input Channel Implementation - Channel Type 105
Figure 4.59 Input Channel Implementation - Channel 106
Figure 4.60 Input Channel Implementation - Connection 107
Figure 4.61 Input Channel Implementation - Channel Connection 107
Figure 4.62 Input Channel Implementation - Node Type Listener 108
Figure 4.63 COTS Configuration 109

Figure 5.1 COTS Transmission 112
Figure 5.2 DSX None 115
Figure 5.3 Status None 115
Figure 5.4 DSX Low 116
Figure 5.5 Status Low 117
Figure 5.6 DSX Medium 117
Figure 5.7 Status Medium 118
Figure 5.8 DSX High 118
Figure 5.9 Status High 119

List of Acronyms and Abbreviations

AI	Artificial Intelligence
ANIME	Army Network Information Management Environment
API	Application Programming Interface
BBNM	Behavior-Based Network Management
C2	Command and Control
CIP	Common Intelligence Picture
COA	Course of Action
COIN	Counterinsurgency
COP	Common Operational Picture
COT	Cursor On Target
COTS	Cursor On Target Simulator
CSV	Comma Separated Values
DL	Deep Learning
DOD	Department of Defense
GINA	Global Information Network Architecture
HCI	Human-Computer Interaction
IP	Internet Protocol
JSON	JavaScript Object Notation
LIDAR	Laser Imaging, Detection, and Ranging

MAGTF	Marine Air Ground Task Force
MCAGCC	Marine Corps Air Ground Combat Center
MDE	Model-Driven Engineering
MDO	Multi-Domain Operations
MDOC5i	Multi-Domain Operations Command, Control, Computers, Communications, Combat Systems, and Intelligence
ML	Machine Learning
MWX	Marine Air Ground Task Force (MAGTF) Warfighting Exercise
NETCOM	United States Army Network Enterprise Technology Command
NLP	Natural Language Processing
NPS	Naval Postgraduate School
OSINT	Open-Source Intelligence
PAI	Personal Activity Intelligence
PCAP	Packet Capture
SQL	Structured Query Language
TCP	Transmission Control Protocol
TECOM	Training and Education Command
UDP	User Datagram Protocol
UML	Unified Modeling Language
USMC	United States Marine Corps
VRDM	Vector Relational Data Modeling
XML	Extensible Markup Language
XMPP	Extensible Messaging and Presence Protocol

Acknowledgments

I would like to thank Dr. Thomas Anderson for his guidance and instruction throughout the course of this thesis.

I would like to thank Dr. Armon Barton, and Dr. Gurminder Singh for their combined efforts and assistance throughout the course of this thesis.

I would also like to thank Dave Tinsley, LtCol Manuel Zepeda, and the rest of the Multi-Domain Operations Command, Control, Computers, Communications, Combat Systems, and Intelligence (MDOC5i) team for their assistance, motivation, and dedication to this thesis and to the advancement of USMC capabilities.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 1:
Introduction

In recent conflicts, the United States was able to assume technological superiority over its enemy [1]. However, since the U.S. has shifted focus from Counterinsurgency (COIN) operations to conflict with near peer adversaries, that is a luxury that can no longer be assumed. The United States and Department of Defense must continuously seek to gain and maintain a technological advantage over near peer adversaries. This has been emphatically laid out by all service commanders, including the Commandant's Planning Guidance [2]. The cyber battlefield is an increasingly complex and rapidly evolving domain that has never been seen in warfare to the extent that it is now capable. Current adversaries have both the desire and the capabilities of mastering that space [1]. Human-Computer Interaction (HCI) will be crucial to achieving information dominance in future conflicts. HCI incorporates computer science, cognitive science, and human factors engineering in order to "focus on the design of technology and, in particular, the interaction between the users and computers" [3]. We must master the human-computer interaction to aid commanders and maintain an advantage over the enemy.

The United States Marine Corps (USMC) is not well-equipped to succeed in the cyber domain. USMC Training and Education Command (TECOM) has identified this capability gap as a major problem statement, "The Marine Corps is not trained to meet peer threats, where we no longer enjoy historic advantages in numerical or technological superiority. To win on future battlefields, we must provide a learning framework that develops adaptive and decisive Marines and provide training environments that produce interoperable units capable of decisive effect" [4].

Advances in information technology have yielded a network centric application framework [5] that can help close the capability gap and allow USMC to maintain cyber dominance against our adversaries.

1.1 MDOC5i

Amongst initiatives taken to meet the Commandant's guidance and prepare the USMC for the cyber battlefield, the Marine Corps has established the Multi-Domain Operations Command, Control, Computers, Communications, Combat Systems, and Intelligence (MDOC5i). MDOC5i is a system of systems based on the Army Network Information Management Environment (ANIME) that provides a network centric, causal dynamic digital twin environment. Utilizing entity-based simulation, MDOC5i provides net centric interoperability and decision modeling that can enhance Multi-Domain Operations (MDO) [6]. MDOC5i plans to "provide grassroots developed technologies to enable operators to 'infer and adapt' to the changing needs of the evolving battle space" [7]. MDOC5i identifies three problem areas that need to be improved upon: interoperability, information processing and exploitation, and cultural shift [7].

As the battlefield continues to evolve, joint solutions will be crucial to gaining the advantage. These interoperative solutions will be reliant on the network and ability to communicate. Interoperability refers to the ability to communicate between all devices associated with various communication systems throughout the services. Thus, the current gap in interoperability needs to be closed to conduct joint operations. Interoperability between systems has not been achieved by a standardized and universal method [7]. MDOC5i identifies the root of this problem to be that all networks used by current systems are considered separate domains from each other, rather than one unified combat Command and Control (C2) system [7].

The next problem MDOC5i addresses is information processing and exploitation. This problem refers to the current inability throughout the Marine Corps to process massive amounts of information. Data is often abundant, and will become increasingly more abundant as sensor capabilities grow, but it is difficult to analyze all of it and sort useful data from noise. The overwhelming amount of data is useless, if not harmful, to decision making processes if not properly analyzed. The problem is depicted specifically as "the speed of current operations and data collection outpace our ability to process, identify, and derive actionable intelligence to quickly assess, adapt, and modify plans and real-time COAs to optimize force projection, lethality, and enable persistent overmatch" [7].

To improve the ability to process increasing amounts of data and keep up with the rapidly

evolving battlefield, the warfighter needs to focus on human-computer interaction. This relationship is crucial for being able to turn massive amounts of data into useful information, in an actionable timeframe, to make better decisions. Better HCI can help ensure that the "speed of data processing and decision-making is commensurate with the speed of operations" [7].

The final problem addressed, cultural shift, addresses the DOD's need to adjust its focus regarding data integration and joint operations. While the DOD is committed to providing actionable intelligence to the warfighters, its approach is ineffective and inefficient [7]. Further, individual services develop their own method of employment and approach to intelligence, which often results in ineffective combined operations. To operate in the dynamic battlefield that currently exists, the services must work together to "align capabilities with the mission, standard operating procedures, training tactics and protocols, acquisition and deployment policies, and overall culture of the operational force" [7].

1.2 MDOC5i Involvement in the Marine Corps

MDOC5i conducted a demonstration with 7th Marines at Marine Corps Air Ground Combat Center (MCAGCC) Twenty-Nine Palms from May 9 to May 13. The purpose of this initial test was to showcase enhanced fires capabilities that are enabled by MDOC5i and to determine the viability of MDOC5i to enhance operations by providing Common Intelligence Picture (CIP)-Common Operational Picture (COP) and decision support across Marine Air Ground Task Force (MAGTF) for MDO.

The MDOC5i exercise at MCAGCC Twenty-Nine Palms successfully depicted the fires capabilities of the system. Using state of the art scoping mechanisms and targeting systems, the MDOC5i system transforms standard-issue, area-fire weapons into precision fire weapon platforms capable of effective target engagement with little to no zeroing. While this alone significantly increases lethality in the Marine Corps, the enhanced fires capabilities are just the start of the utility provided by the MDOC5i concept. The underlying system uses the Global Information Network Architecture (GINA), a Vector Relational Data Modeling (VRDM) platform, to enable an accurate COP and CIP across all units tied through the network. This provides an advantage on the battlefield as all units gain awareness and would be able to contribute inputs to the shared system, resulting in the most accurate CIP-COP.

These inputs can be used to aid decision making and influence activities conducive to the competition to conflict space.

One of the critical enabling components of this process is the decision model within GINA that enables action. In the demonstration at Twenty-Nine Palms, Marines were shown the capability of first-shot accuracy on selected targets using standard weapon systems. The selected targets appear on all information displays tied through a network. To enable the targeting, the GINA model takes the input of a target and passes the information to all users. The system first makes a decision if the target is a valid target or a duplicate. It does this through a specifically designed decision model, which compares identified targets to other plotted targets. If the new target is within a specified distance, the program would deem it a duplicate. This prevents information overload and allows the commander to have the most accurate depiction of existing threats to better decide on how to use weapon systems to engage the enemy targets. Thus, in this specific instance, the input was locations of identified targets, the decision was if that target is legitimate or a duplicate, the deciding criteria was a determined stand-off distance from other already plotted target, and the result was an accurate depiction of threats that allows the Marines to best engage the enemy.

In the demonstration, the decision was related to identification of targets, and the influenced action was related to firing. However, as previously mentioned, enhanced fires capabilities are only the start of the benefits that MDOC5i can provide through the VRDM based GINA platforms. Creating and employing models that provide commanders with the most up to date CIP-COP and aid decision making would be useful across all facets of the Marine Corps and Department of Defense (DOD). As it stands now, a new decision model would need to be created from scratch every time a new model is implemented.

1.3 Thesis Focus and MDOC5i Tie-in

The intent of the Naval Postgraduate School (NPS) thesis is to build an agnostic decision model using a VRDM in the GINA platform. The focus is on the ubiquity of the model so that it could be easily molded to future scenarios. The decision model extends the ubiquitous data table concept to encompass information attributes about the data, and allows realization of knowledge attributes derived from both the data attributes and information attributes (logic type) through attribute-based relationships with truth tables. Thus, the model transforms

data to information, then acquires knowledge of state and prescribed processes from known truth (established protocols), and then the model executes the appropriate processes. This demonstrates the ubiquity of the approach and enables the transformation of data to action for any data-mission. This thesis validates the methodology of using a configured model-based approach consisting of conceptual objects of data, truth tables, and status that may be utilized for man-in/on-the-loop automatic data-decide-act and may be managed within a knowledge management graph framework for any mission.

The proposed model is tested on a scenario of determining network health by analyzing Extensible Messaging and Presence Protocol (XMPP) messages. The input to the model is fabricated Extensible Markup Language (XML) messages, designed to replicate the XMPP messages captured in the Packet Capture (PCAP) of large scale tactical networks. While the network diagnostic classification is important in itself and proves functionality, the main utility will lie in the ubiquity of the decision model. Because the model is agnostic, it could easily be modified to fit a range of desired scenarios. Pragmatically, it could serve as the base model for all other GINA implementations that will enable the Marine Corps to achieve information overmatch.

1.4 Hypothesis and Research Questions

The hypothesis of this thesis is that GINA will prove to be an efficient and effective platform in which to implement a ubiquitous decision model that can be easily configured to address multiple situations. At the core of this hypothesis, the primary objective is to successfully design and implement a ubiquitous decision model using the GINA architecture. That task was accomplished, proving the main hypothesis correct.

Questions for this thesis include:

1. Can a ubiquitous decision model be implemented in GINA's interface?
2. Does GINA offer a viable and actionable alternative to Machine Learning (ML) and does the model achieve the same effects as traditional machine learning techniques?
3. Are there tangible aspects of the model that prove superiority to traditional machine learning techniques?
4. Can the model and the GINA platform be used for large-scale network traffic analysis?

In line with the hypothesis, the first question was the most significant and was proven correct. The decision model that was implemented should enable and motivate future work. The remaining questions addressed the scalability of the model and the performance of the model in comparison to traditional techniques. While neither of these concepts were addressed directly, the model provides the opportunity to definitely test these concepts.

1.5 Tools Used

To successfully understand the implementation of the decision model and the scale to which it could be applied, it is necessary to understand the tools involved. Some of these applications are used directly in this thesis. Others are used in MDOC5i and will be useful to understand how this model can be extrapolated to multiple situations. The tools also provide good context and would be beneficial for future work.

1.5.1 Global Information Network Architecture

GINA is a cloud based, VRDM platform that provides an executable modeling environment which produces models capable of inference and adaptation [7], [8]. The architecture provides advanced data, information and knowledge interoperability through its reflexive, executable, component-based, platform agnostic, and model-driven construct [9]. The platform uses a semantic construct that allows application domain users to understand the constituent model components and results in systems with semi-sentient behavior, which is critical to adaptability and configurable flexibility for dynamic mission needs. The innovative platform is loosely coupled, meaning it can create models through configuration that use various inputs from legacy, existing, or future systems [8] without breaking or recompiling. Due to the conceptual information object constructs that may be introduced ad hoc and may exist in any domain, GINA offers tantalizing possibilities that are being explored by DOD [2].

GINA technology is comprised of methodology, development tools, and a deployment platform for executable models that function as software programs. The models do not have to be compiled, rather they are defined in metadata and compiled in real time. The platform uses behavioral, contextual, and causal modeling concepts implemented through configuration to provide definition, manipulation, and interoperability [10]. GINA can

be further understood by the components of its name. “Global” refers to the platform’s inclusion of all digital representation through multiple layers of abstraction. “Information” refers to the static and dynamic data and interactions that can be modeled and managed. “Network” refers to the digital representation of all interconnections that can be displayed, referenced, and managed through models and graphs. “Architecture” implies that GINA is the system that is used, specifically for making executable models of behavior, context, and causality [10].

The benefits and characteristics of GINA are discussed in depth in chapter two.

1.5.2 ANIME Dark Stax

Dark Stax is a tool developed and used by ANIME that is capable of creating digital twins of complex systems at near real time speed. These digital twins can be used to manipulate the cloned system for data manipulation and decision analytics. This federation aids in the data-driven decision-making process. This tool is capable of creating clones of tactical networks and filtering PCAP data to create input for network diagnostic models [10]. The Dark Stax tool is maintained and operated by Ad Hoc. Their mastery of the tool creates for a combination of premier Artificial Intelligence (AI) techniques and VRDM techniques that offer vast utility.

1.5.3 StarUML

StarUML is an open-source software modeling platform that supports the Unified Modeling Language (UML) [11]. It is designed to support concise and agile modeling and provides visual depictions of system overlays [12]. This paper uses UML diagrams to depict the static and dynamic aspects of the implemented VRDM model. UML does not capture all details contained in the VRDM model, but it does capture the most important information and provides a clear overlay of connections in the model.

For the purposes of this project, it is only used for visualization and documentation of the GINA model. However, the intent is to make GINA capable of accepting a UML design as input. Thus, a system could be modeled in UML and input into GINA to forgo the configuration.

1.5.4 Cursor on Target Simulator

Cursor On Target (COT) "is an Internet Protocol and an XML based machine-to-machine schema that can be read and understood by any system, enabling proprietary and open source systems to communicate with each other" [13]. The simulator is used in the GINA model to simulate XMPP traffic. Sample XMPP messages are generated in a text document. The Cursor On Target Simulator (COTS) simulator then inputs the contents of the text documents into GINA as XML. This XML is the input of the decision model.

CHAPTER 2:
Background Information

To enable the Marine Corps to gain the technological advantage, MDOC5i aims to deploy the most efficient and effective modeling techniques. While modern computing techniques, such as ML and Deep Learning (DL), are incredibly powerful tools, they are likely incapable of providing the optimal solution for the future battlefield. This section discusses the limitations associated with traditional AI techniques that hinder its utility. The section also states the benefits associated with GINA and VRDM models and discusses why MDOC5i favors this approach.

2.1 Limitations of Machine Learning and Deep Learning

AI has made significant strides in the last two decades and continues to do so. Overall, it has shown that tasks which were previously thought to be difficult, can be quite simple with AI methods such as DL and ML, amongst others [14]. However, while they have proven to be incredible processors, current AI techniques are unable to achieve the goal of human-like intelligence and likely never will. AI techniques, to include ML, DL, and Random Forests, amongst others, operate mostly in associational mode. This means that models constructed with these methods can only respond to pre-programmed and known situations. They have no contextual awareness to react to new and unknown situations [15]. While some complex methods can be advanced enough to program multiple contingency paths to account for known possibilities, they always risk the potential of encountering a new situation which they have not been told how to handle. They do not have the ability to learn and grow on their own. This lack of flexibility and adaptability is an insurmountable problem in achieving human-like cognition and a hindrance to the human-machine integration.

The inability of AI technologies to adapt to changing circumstances is a result of the lack of ability to manage concepts and relations between concepts. Human cognition recognizes relations between different objects or events and can often use that relation to infer proper actions. Humans can use these relations to apply an internal causal dynamic model, in which contextual indicators are used to determine or influence a decision [15]. Human cognition

achieves a level of understanding far past calculations and statistics, it achieves a level that recognizes theory and reasons and is capable of imagining [15].

AI methods do not have the ability to process concepts across relations and are, therefore, unable to truly meet this level of causal dynamic reasoning [15]. To provide more meaningful context in the real world, models must be able to make concepts, relations, and causality executable in computational environments. Statistical models are unable to determine exactly what something is if it does not already know it. They cannot create new concepts or adapt to learned causal relationships [15]. Current methods are not intended to grow and adapt amongst different types of users, languages, domains, and cultures [16]. They are limited solutions that would require constant reevaluation and upgrades when employed in dynamic situations, lest they become quickly outdated.

An illustrative example of this difference can be understood by thinking about how a human driver and a driverless car would react to a specific situation. The circumstances are that of a pedestrian walking on the sidewalk with a bottle of whiskey in their hand. A driverless car will react to the scenario based on one of the many rules they were programmed to follow, in this case likely telling them to proceed as planned because a pedestrian is on the sidewalk. A human driver on the other hand would see the bottle of whiskey, would recognize the abnormality, and be able to apply causal effects. Those effects being the pedestrian is likely drunk and could fall into the street. The human would then increase attention or change driving patterns to maintain a safer situation. This is the type of insight that current artificial intelligence techniques will not be able to achieve [15].

Without the ability to process causal dynamics, or conceptual and contextual relations, AI methods will never grow past the role of crunching large quantities of data for optimized pattern matching. While pattern matching can be useful in identifying what something is in relation to a group of pre-existing objects, it is not able to determine why something exists, what something could have been, or what it will be in the future [15].

Though controversial, one could say ML and DL do not contain any intelligence, rather only the appearance of intelligence. Despite the recent success in AI, the underlying methods and logic are not a drastic improvement to artificial intelligence, but rather a reapplication of old methods using more data and increased computational power [15]. The foundation

of these methods remains statistical math such as data crunching, pattern-matching, and object-oriented association, which are all decades old. While AI can capitalize on increased power and large databases to calculate statistically relevant associations more effectively and efficiently, this does result in any true improvements to the intelligence of the machines. A radical interpretation of this can be expressed by "Today's slick AI applications are powered by a massive amount of custom engineering and data preparation in order to accomplish a single task or collection of stove-piped related tasks. These statistical algorithmic machines are less of an 'artificial intelligence' than they are a powerful tool. . . called 'spreadsheets on steroids"' [15].

To truly introduce intelligence into computing and achieve the desired effects of human-machine interaction, there must be a connection between objects and relations. Cognitive computing must go beyond representation of symbols like ML, and correlations and attributes like DL, but must represent the "causal dynamics existing between the concepts and systems of concepts that are the models of our interactive world" [15]. While there is benefit to these AI applications, the dynamic battlefield will require human-machine interaction to best aid the warfighters' decision making.

2.2 Vector Relational Data Modeling

To achieve superiority in the information domain and gain a significant advantage in computing and machine-aided decision making, it is necessary to explore model building capabilities beyond common AI techniques. MDOC5i recognizes this critical capability and created behavior-based and integrated models for network management [7]. The intention is stated as "The MDOC5i system will model dynamics between dimensions of causality and enable the provision of actionable intelligence derived from multi-sourced, multi-dimensional data" [7] which will provide "cognitive amplification to influence decision-making and associated behavior in a manner consistent with mission objective and desired end state" [7]. MDOC5i uses entity-based simulation to model causal dynamics, aiding near-real time decision support and enhancing the COP. To accomplish this, models need to understand underlying causality and reasons for behavior.

The MDOC5i concept focuses on VRDM. The basis is that vector executable models are more "accurate, flexible, obvious, and robust than the traditional code required by object-

oriented programming or relational database models" [7]. The use of VRDM provides the Marine Corps with a "competitive technological advantage that facilitates our ability to respond quickly and cost-effectively to rapidly evolving adversary challenges" [7].

VRDM optimizes the effectiveness of existing information systems by creating a conceptual modeling layer capable of inference and adaptation [15]. Within VRDM applications, data, services, sources, and relations can be represented as discrete objects within an executable model, which enables the manipulation and processing of all concepts. The crucial aspect is the "ability to wield relationships and concepts as malleable objects" which is critical for configuring systems that are capable of inference and adaptive behavior [15]. A VRDM application's ability to learn by processing concepts and relations produce results closer to emulating human cognition than that of any common AI techniques.

2.3 Global Information Network Architecture

MDOC5i uses GINA to bridge the gap between human cognitions and machines. GINA can be described as "an executable modeling environment, with models capable of inference and adaptation. The system is used for information management for heuristics predictive, and prescriptive analytics; sensor fusion, systems integration, and interoperability" [7]. GINA enables the creation of interrelating concepts for control, adaptation, inference, and causal dynamics over connected systems. In line with the goal of MDOC5i, GINA provides a capability to perform decision support by modeling causal dynamics. MDOC5i defines the need for this capability as "Today's MDO environment, has many non-tactical aspects, such as cultural, political, cyber-social factors that evolve rapidly and can greatly change which Course of Action (COA)s should be considered to achieve mission objectives. The critical relationships between these factors have with the kinetic world are called causal dynamics and are usually highly complex" [7].

The benefits offered by GINA and VRDM can vary based on the intended purpose of the system. The most pertinent benefits of GINA models are flexibility, adaptability, reflexivity, interoperability, efficiency, and affordability. Flexibility is the ease in which a VRDM model can be modified and increased in size after initial employment, based on new requirements such as data and events. This is attributed to instantiation of applications within the GINA framework that is built of an information system where the applications are configured

components whose attributes are stored in information tables, thus no compiling is required. This renders value rapidly to the user in scenarios regarding new enemy capabilities where simple configuration or addition of data prevents the model from becoming outdated quickly, as is the case with rigid AI compiled models. Adaptability refers to the model being able to change itself after receiving new inputs. This ties into flexibility, but the difference is the focus on the model taking self-action. This refers to the model being able to react to new inputs that were not specifically programmed, and to develop actions based on relations. Reflexivity ties into adaptability and flexibility but refers to the fact that GINA itself is a model within GINA. This implies that when a change is made at one level, either by the user or through the model itself, changes are made to all levels based on the vector relations. Interoperability allows it to work with multiple platforms and organizations. This is possible because GINA has an open and simple API and it has a service that is easily configured to any JavaScript Object Notation (JSON) or XML input or output along with file watching capabilities to read JSON,XML, Comma Separated Values (CSV), and more. GINA may exist as a secure enclave in the cloud or on a laptop, making network availability convenient to any system or sensor at strategic or tactical levels. The final benefits, efficiency and affordability, are the most straightforward. It is efficient in use and training because there is no need to learn code, with no maintenance of a specific system or hardware.

This section outlines five key characteristics of GINA that enable these defined benefits and discusses how they are interrelated.

2.3.1 Semantically Configurable

GINA is a "model-based architecture for semantic interoperability" [2], which means that models can be constructed without intensive code. The benefits of this are twofold. First, people with little to no coding experience can now interpret or build models. Second, because code is inherently rigid and error-prone, eliminating the need for it reduces errors or misinterpretations in the translation from idea to code. A vector relational model, on the other hand, is much more effective at capturing semantics.

An example of the semantic assembly for a model can be seen in the implementation of "rock- paper-scissors" [2]. The key takeaway upfront is that the model is data driven, not coded. The model is defined by three objects: "force," "force-type," and "strategy." "Force"

is the players of the game, “force-type” identifies which side the player is on, and “strategy” refers to the move options a player can select. The model also creates three relationships: “selects,” “defeats,” and “engagement.” “Selects” refers to the strategy that a player chooses for an engagement, “defeats” defines the rules of the game, or truth table, and is used to determine the victor, and “engagement” refers to a round of the game itself. For an engagement, there are two sides, and the outcome is determined by comparing each side’s strategies in accordance with the rules defined by the defeat truth table. Therefore, outcomes are determined in real time by comparing the created table, rather than programmed. The article provides an example of how an engagement between two players would go, “in the first row of ENGAGEMENT, FORCE ‘F1’ selected ‘Paper’ and ‘F2’ selected ‘Scissors’ with the results ‘F2’ won since ‘Scissors’ beats ‘Paper’ as specified in the defeats table” [2].

The coded version of the game would require programming every possible choice, outcome, player, and relation. While this is feasible for a game as simple as “rock-paper-scissors,” the complexity drastically increases as the scenario changes. For example, if a new player wanted to enter the game with new strategies or forces, the coded version requires significant changes to account for the new player, new choices the player can make, and the cascading effect of possible outcomes based on the new player’s decision. On the other hand, the GINA model simply adds a player to the “force” list, and the player abides by the rules already established. The same logic applies to the change or creation of rules, relationships, and strategies, where the modifications are captured as table entries in the respective concept. The scalability and efficiency to handle complex scenarios is depicted in the rock-paper-scissors example.

2.3.2 Modeling of Relationships

The second significant characteristic of GINA is its representation of relationships as objects. This serves as a key element in the flexibility and scalability of GINA. While traditional data modeling techniques address objects and relationships as primary components in models, the significance and primacy of relationships do not translate into code. Therefore, the resulting model is limited to the specific, originally coded, relationship. Even if these models contain no bugs, they are rigid and unable to adapt to new relationships, restricting the level of intelligence these models can achieve. GINA uses “vectors” to treat relationships directly as objects, which allows the platform to perform robust set operations more efficiently

than traditional relational databases [2]. An example of these complex operations is seen again through the rock-paper-scissors model. The model has relationships of "defeat," "selects," and "engagement." The authors refer to the example of enabling a force (object) to select (relationship) a strategy (force) based on previous adversarial strategies. While join statements in relational databases are unable to perform such a task, GINA offers the capability due to the semantical build and definition of how objects interact in certain conditions [2].

The GINA model works through the modeling of objects, known as X-types or Concepts; relationships, known as Vectors; and Actions, also known as Services. It was designed specifically to optimize the process of translating a conceptual model into software, improving upon the traditional techniques' time-consuming, imprecise, and rigid methods [16]. The modeling construct of GINA enables it to "configure extensible causal dynamic models, with the ability to infer and adapt, that traditional programmatic computational designs fail to achieve" [15]. Since the models are self-contained and executable, GINA framework can interoperate with any type of system efficiently, and much more cost effectively than solutions built using current software methods [15].

2.3.3 Reflexive

The next important characteristic, the reflexive architecture of GINA, contributes to the scalability and adaptability of the model. Any instance of a GINA model is implemented on GINA architecture, meaning that GINA itself is an application of GINA [2]. Though seemingly redundant, this level of abstraction is critical to the nature of what GINA offers and provides a significant advantage to traditional models. This hierarchical "meta-level" enables a GINA model to be self-descriptive and reflexive, which in turn accelerates the scalability, adaptability, and rapid implementation of the model [2]. This meta-level can be abstracted to as many layers as necessary, thus efficiently expanding the scale of any model. Because GINA models are contained within themselves, it can use reflexive, cascading relationships that enable users to change all levels of the model by adding only one input. Adding something to a top-level model will update or create models below it according to the defined Vector. This ability to "change high level meta-descriptions and then have the resultant modifications propagate down through the various meta-levels without having to change code" is a feature unique to VRDM that renders GINA incredibly powerful [2].

GINA's reflexive nature significantly reduces, or nearly eliminates, the amount of coding required to develop a system because it leverages both model-based architecture and component-based development. GINA's method for employing a component-based development is unique in that GINA translates configurable models into suites of corresponding component objects, thus creating its own components during design. This process creates a semantically consistent executable model that executes without any dynamic automatic code generation. This saves significant compiler and interpreter time, and avoids semantic ambiguity in all levels of the model that results from code translation [2]. Because the conceptual models are separated from code, the user has more flexibility to create models. Instead of needing to produce a complete and correct model using code, GINA gives users the ability to create a small portion of the model. With that, the user can use data, simulate, and observe events before easily correcting the model if need be [16].

2.3.4 Platform Agnostic

The next feature of GINA that benefits the given cyber tactical domain is the fact that it exists in a secure enclave on the web. This means it is agnostic to the device it is on and can be implemented anywhere, on any network, with the guarantee of security. This is incredibly important in terms of interoperability, security, and efficiency [2]. The environment also enables communication amongst all systems through the handling of different types of inputs, including Cursor-on-Target, XML, and JSON and accommodating the syntax for any system. The ability to handle and model all types of inputs makes this characteristic an enabling factor of interoperability and adaptability, as the device can take in and react to new, potentially unexpected data [15].

2.3.5 Metarules

A characteristic that supports all benefits of GINA, especially flexibility, adaptation, and inference, is the utilization of metarules. Metarules are Concepts and Vectors that are coupled to, and extend a GINA model which allow model overwatch at the specified component level, and subsequent decide and act behaviors to modify the model components and have the model behavior evolve. Metarules use "conceptual components and relationships among GINA components to deduce meaning... and describe meaningful relationships among conceptual components that can be used to infer intent" [16]. They are critical structural

elements of conceptual models and enable adaptation significantly more than traditional models because they are not reliant on specific patterns in data. Using metarules results in an incredible advantage over normal models in terms of flexibility and adaptation. Since metarules are capable of deducing meaning and are not dependent on specific patterns, they are capable of self-evolving if network experts gain experience and observe a necessary change, more data is collected, or adversaries change their threat tactics [16]. This is a huge advantage over static models which quickly become obsolete as situations change. Network threats continuously develop and evolve. ML models do not adapt to the change adequately and autonomously, causing the model to become obsolete quickly. With VRDM, new information updates the state of a component [16]. Metarules will then enable changes in associated components based on relations at all levels and then reassess the changed parameters in relation to the new data. These assessments and feedback loops allow the system to autonomously update. This results in models that not only last longer but are more effective [16].

Models execute in the form of different possible states. As information enters the model, the model evaluates what state it should be in and what actions it needs to take. As the data adds instances and entities, the model adapts to new behaviors. This may be seen as changes in state, such as weights causing metarules to trigger or data driven relationships causing changes in instances of entities identified as targets. Next, as the model updates from the new data, metarule-based execution and evaluation will occur up and down the model. Based on the evaluation, the model will further adapt to reflect new information regarding entities, local updates in accordance with metarules, model notification that aggregators have changed, and reevaluation of metarules based on recent updates. As metarules trigger changes to certain components of the model, all related objects will also reflect the change [16]. The relational nature of conceptual models also enables the models to build and update inferences, which includes the ability to instantiate and classify actors as bad or good based on their respective behaviors, while continuously updating the criteria.

An example of this metarule triggering change can be seen in the Lab-Rat experiment [16]. The experiment showed a network classification system, called Behavior-Based Network Management, built through a relational database. The system implemented metarules that could learn and adapt to a rapidly evolving digital domain, rather than rely on assortments of rules such as whitelists and blacklists. The feedback and updating loop that exists due

to the metarules create a system that can respond to new threats and information without the need to recode. The Behavior-Based Network Management (BBNM) example of a metamodel demonstrating system learning and behavior evolution has profound implications on applying a VRDM approach to the multidomain space.

The quick and efficient self-adaptation of the model, along with the transparency of the model, provides human users with the most accurate information. Overall, the VRDM models "create a framework that supports commanders at the strategic, operational, and tactical echelons through the development and broad use translation of an advanced decision support capability for analyzing traffic within a behavior model" [16].

2.3.6 GINA Visual Construct

Having discussed the benefits of GINA and the characteristics that enable those benefits, it is beneficial to see a generic GINA construct. Figure 2.1 depicts key fundamental object components of the GINA model of application that are used to configure new applications [8]. It should be noted now that key GINA concepts are capitalized throughout.

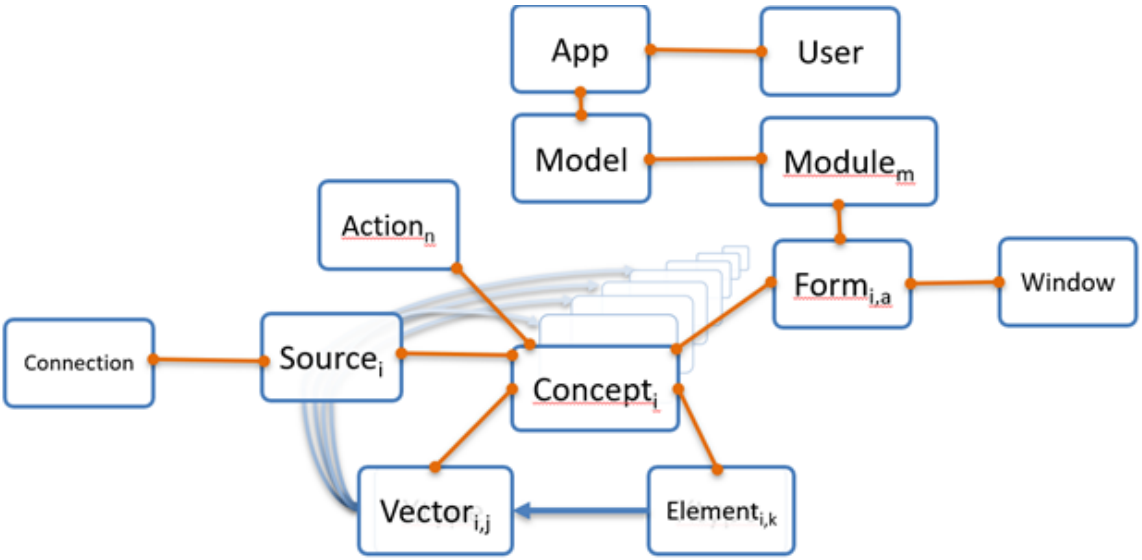


Figure 2.1. This is a representation of the key conceptual objects (not exhaustive) in the GINA model. It is of note that all concepts in the GINA model are instances of the GINA Concept and that the User is an Avatar or object in the model allowing modeled zero-trust to any information or solution. Source: [8].

The diagram depicts multiple Concepts overlaid on top of one another and the multiple arrows going from the "Vector" to the Concepts. The Vector is an objectified relationship governed by specific Concept attributes and exists in a state such that any new Concept instantiated in the model may relate automatically based on having specified attributes. Thus, a change made in one model Concept will be reflected in all related model components.

The image depicts the nine core framework objects of GINA and VRDM models, along with additional representations of the user, application, and module objects. Of these, the key components to VRDM modeling are the "Concept," "Vector," and "Element." The "Concept," also referred to as an "X-type" within GINA, in simple form may be considered as an abstraction of a database table [5]. "Concepts" may also be thought of as a class [17]. The "Element" object may be considered an abstraction of a table column, or attribute [5].

The Element has attributes and stereotypes that indicate the type of the Element [17].

The “Vector” object is used to form and represent the relationship between Elements [5]. “Vectors” are used as directed association to declare relationships between Concepts [17] based on the Concept’s attributes or Elements. When an Element is configured to use a Vector to access a related Concept, the directed association will have the name of the Vector and the associated stereotype. The other end of the association will likewise have the same name as the Element it represents [17]. This approach affords the ability to name the Concept, Vectors and Elements with semantically meaningful terms from the mission domain. Thus the configured underlying model is inherently meaningful to specialists in any domain.

The remaining GINA components are common with what would be seen in other platforms. The “Connection” specifies the type of connection being used and the information necessary such as IP address and name of the source database [5]. The “Source” object specifies the previously mentioned Connection information and the data source table. The Source acts as a directed pointer to the specified database table for accessing table columns as Element objects [5]. The “Column” object, not depicted in the image, is related to the Source and refers to the specific database column for the table specified in the Source [5]. The “Form” is the GINA object that describes the web user-interface page which contains the web page information panels, specified by ”Window” [5]. The “Windows” are typically configured to be of three types: authority, resource, or collection. Authority Windows display limited attributes that typically include human navigable contextual Elements for the referenced Concept. Resource Windows display the Concept’s Elements in the same manner as the authority Window, however usually in a more complete representation of the Concept’s Elements. Collection Windows display related metadata from related Concepts as identified by collection Vectors [5]. The “Action” object refers to the method parameters that are included to list Elements on the target object. The Action gives the owner manipulation abilities on the object [17]. Action is an abstraction of the fundamental actions of Save and Delete that exist in information systems that affords attributes to include transforms. In modeling sequential saves along with filters, relations, and transforms in an information system, one achieves complete capability of data-to-information-to-knowledge-to-process. This executable model framework is unique to GINA.

2.3.7 GINA Conclusion

In summary, GINA is a flexible platform and offers the ability to integrate various types of data across multiple systems in order present information specifically tailored to the user [16]. For these proposed benefits, it offers an intriguing alternative and compliment to traditional modeling techniques, which is worth exploration.

2.4 Recommender Model

The framework of the decision model is based loosely on the recommender model depicted by Almonte, Guerra, Cantador, and Lara [18]. In their research, they reviewed results of using recommender systems, which are information filtering systems that can be used to facilitate software engineering activities and assist in the modelling development process, for Model-Driven Engineering (MDE) [18]. The work served as a systematic mapping review of which specific MDE tasks were best suited for recommendations, and which recommendation techniques and evaluation methods were best to use in certain scenarios. Their focus was more on the analysis and back-end portion, rather than the functioning of the recommender model itself. This thesis uses portions of the recommender model as influence for the model.

Though it is not a direct clone, the model provides lose influence to how ours is defined in terms of recommendations being associated with different levels of hierarchy and influencing different course of action for each level. Their model places a focus on determining if recommender systems are suitable for MDE based on domain, tooling, recommendation, and evaluation domains [18]. The model implemented in this thesis will be most similar to the recommendation domain and will focus on specifying tasks and subtasks associated with different levels of a hierarchy. The different recommendations based on level is what contributes to ubiquity in our model.

Pictured in Figure 2.2 is the recommendation domain of the model, presented by Almonte in relation to the analysis work [18]. As depicted, different recommendations branch out with complexity. In the research, a recommendation was represented by four criteria: recommendation method, user preference, recommended item, and recommended degree [18]. The purpose of this domain is to characterize how recommendations are generated based on the four criteria. The combinations of the input criteria will influence the recommendation

made by the model. Each of the four criteria are further broken down into subcategories that provide amplifying descriptive information. This structure provides a hierarchical setup that was useful for the construction of the model created in this thesis research.

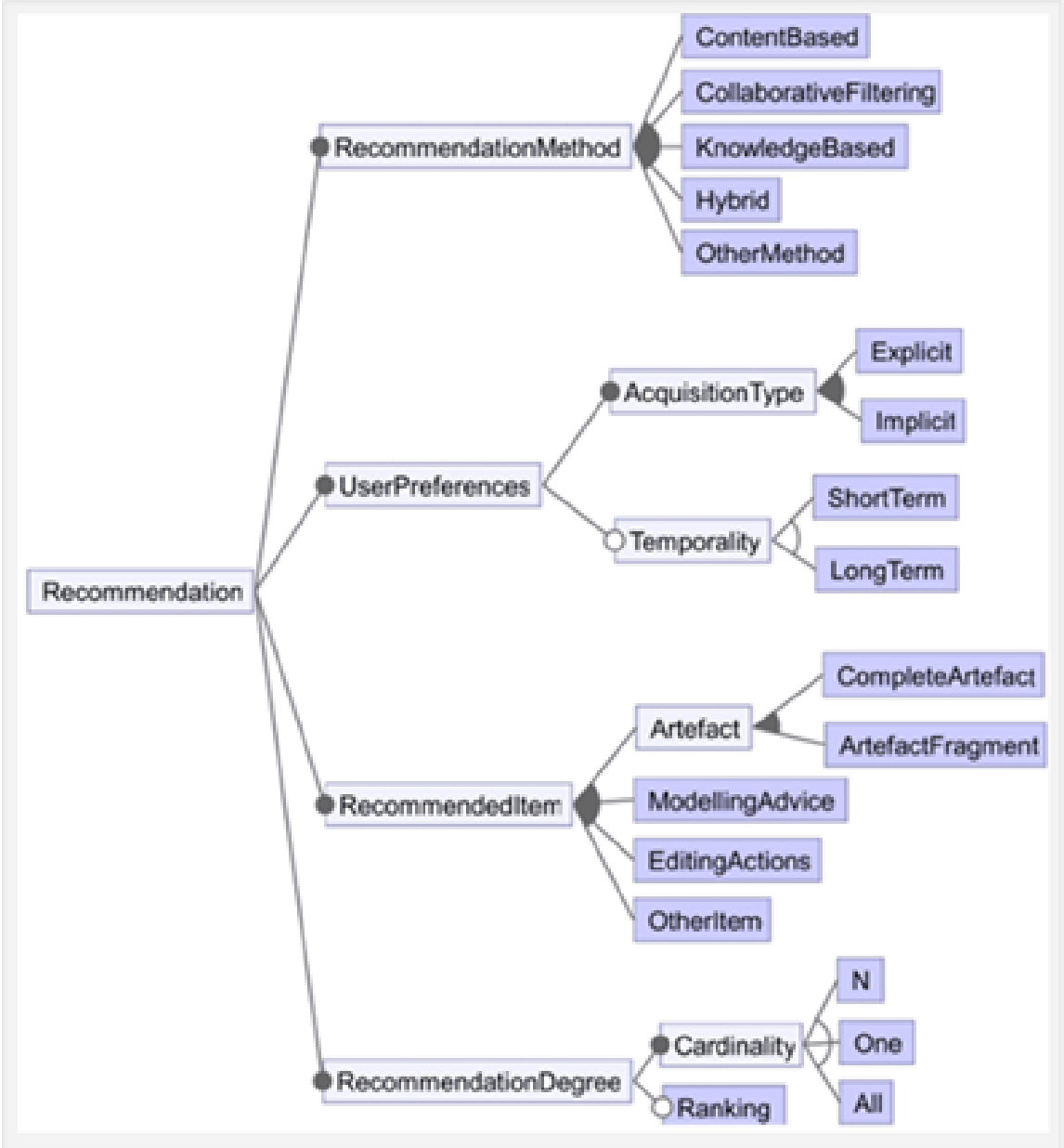


Figure 2.2. The image depicts the recommendation domain of the recommender model created by Almonte et al. The hierarchical structure of the domain represents the influence of different factors on a recommended decision. The network diagnostic decision model discussed in this thesis was influence by this design. Source: [18].

This hierarchical structure influenced the decision model used in this research, showing that each level is concerned with a different set of decisions/recommendations [18]. The hierarchy is further broken down from there, with each set of decisions/recommendations consisting of further subtasks. The first level of the hierarchy separates the user from other scenarios and presents the user with multiple courses of action. The second level of hierarchy recommends a course of action within that section. That recommended course of action has multiple subtasks associated with its completion.

The recommendation degree is another aspect that would work well in constructed model. Given the simplistic nature of the specific use case implemented here, the recommendation degree was not a factor. However, the recommendation degree would be of value with a more complex scenario. The purpose is to give the commander a recommendation on what decision to make. Based on input criteria, the machine could calculate a score that would fall onto a more complex Likert scale. Based on the resulting score, the machine would not only recommend a set of actions to take based on where it is in the hierarchy, but it could recommend how confident it is in that set of actions. The recommendation degree should be incorporated into future models.

CHAPTER 3:
Methodology

This chapter first presents the motivation and utility of the proposed model as it relates to the diagnostics of large scale tactical networks. The preliminary work section discusses the different aspects of diagnosing the health of a tactical network to include the means of data extraction and filtration, division of tasks, and factors of influence. The ubiquitous decision model would act as the base of this large problem, as well as many more.

The chapter then discusses the methodologies by which the decision model will be designed, implemented and tested.

3.1 Preliminary Work

3.1.1 Data Procurement and Manipulation

The intent of this section is to highlight a key capability that would enable the use of tactical data from the Marine Corps MAGTF Warfighting Exercise (MWX). Normally, obtaining an entire clone of a complex network and capturing and filtering all PCAP data is an arduous and time-consuming process. However, using the Dark Stax capabilities addressed in section 1.5.2, the burden is significantly reduced. The Dark Stax technology is capable of making a digital twin of USMC communications network and filtering PCAP data in near real time and publishing in a standard format (JSON/XML). Figure 3.1 depicts how the Dark Stax technology integrates with the Marine Corps network and the GINA model [10].

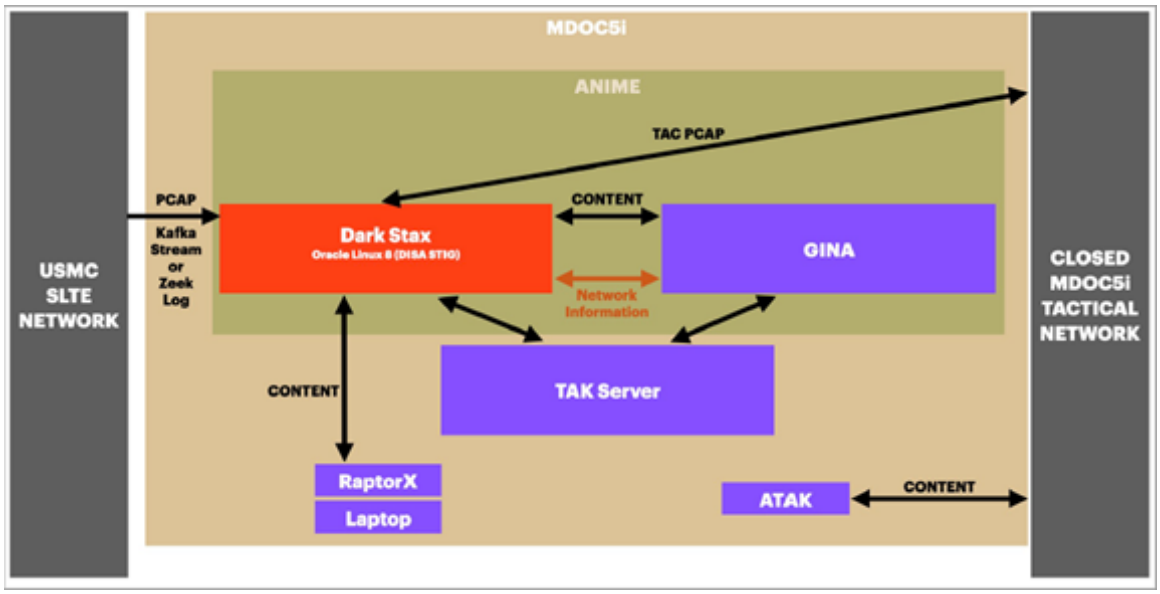


Figure 3.1. The image depicts the proposed methodology for integrating Dark Stax technology and the GINA model with live USMC networks. Though the integration did not occur for the previous MWX, it would offer significant capabilities and should be explored in future exercises. Source: [10].

A connection is made between the USMC network and the Dark Stax server. From the connection, Dark Stax uses causal dynamic modeling of digital twins to filter full PCAP data as a stream of data [10]. Dark Stax is able to filter the full PCAP data for session streams, content translation, and interoperability purposes. From there, Dark Stax can employ the filtered and usable data for a multitude of purposes. Data can be sent to command-and-control platforms and systems to enable the most accurate CIP and COP. The data sharing amongst all linked components, to include GINA models and other MDOC5i systems, allows for a range of real-time information manipulation such as network diagnostics, tactical decision support, and sensor fusion integration. Combining premier AI capabilities, as displayed by Dark Stax, and VRDM capabilities, as found in GINA, offers unprecedented benefits to the Marine Corps. The red arrow depicted in Figure 3.1 represents the connection by which the Dark Stax server would filter the PCAP data and would provide the desired information for input to a constructed decision model in GINA.

For this specific effort, the Dark Stax data filtering of real time systems was deemed outside of scope, and a simulator is used to publish JSON data to the GINA service where it is read into the decision model for a proof of concept. To procure data for this research, a data simulation will publish chat data payload to the GINA service to demonstrate decision making using streaming data source. The fabricated JSON messages were designed to resemble tactical XMPP messages that would have been filtered from PCAP. These messages followed the same structure as OpenFire tactical messages [19].

3.1.2 Overview of the Graph Model

Figure 3.2 depicts the general overview of the model structure. This overview displays the model conceptual components and the process of only one mission for one unit. The function of the graph model here is to provide an organizational knowledge management framework in which the decision model will operate. The main organizational components are unit, mission, objective, and session, these provide enough context to cover a broad spectrum of decisions. Each mission can have a hierarchical break down and each unit can have multiple missions. The purpose of this overlay is to show preliminary whiteboard design of the graph structure and the expansive capacity of the model. While the research will only test one use case to prove functionality, this model can be expanded to include any concept or data from any domain.

The expansive capacity of the model is visible in the various components of the graph. Each component object may have a hierarchical or type organization. For example, the unit concept enumerates different types groups within an organization that relate to mission. In this example, for mission, the different warfighting functions are enumerated. Each unit may have multiple different missions associated with it. Each mission can also be hierarchically instantiated to contain subtasks and further levels of subtasks.

The decision model constructed in this thesis will contribute to this concept. Image 3.2 shows the utility from a knowledge management perspective that the GINA decision model could have in organizing various mission sets for different units. In the one-to-many unit-mission relationship, there is no limit to the number of units that can be added to the model or the number of missions that can be added per unit. The data driven model maintains context that illuminates a specific unit and their related missions, so that the users are not

overwhelmed by unnecessary information of all unit-mission data at once.

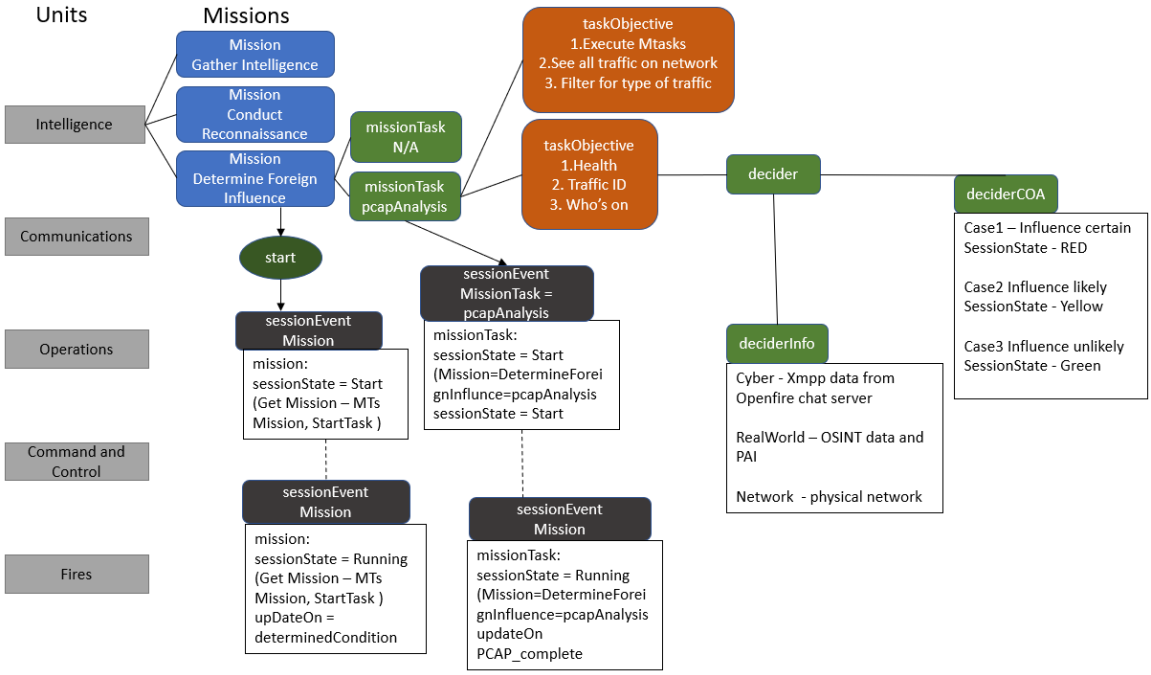


Figure 3.2. Overview of the Graph Model

The left most column is a list of units that are expected to interact with the system. Each one has a specific set of missions that they are concerned with. Users of the system can select, or be associated with, which unit they are a part of. The model will then only be concerned with the associated tasks. This prevents the user from needing to be concerned with information that does not pertain to him or her. Each unit can have an unlimited number of defined missions that they are concerned with.

Figure 3.2 also illustrates the inherent knowledge management of a model based approach where the model manages the concepts, e.g., military units. New Concept instances can be added readily by adding the new instance specification to the underlying table. The model further breaks down missions of the intelligence section. Now, the data informed model only shows related and relevant information. The infantry and fires section do not need to be concerned with those tasks that are not related to them specifically. The model knows what

exists and connects what needs to be connected, but the large lists of tasks do not clutter the user's workspace.

The next column is "missions." This is a list of tasks specific to the identified unit. There is no limit to the size of this list. Missions could be either predetermined or determined by specific input criteria. The specification of mission allows the unit to focus in on a specific group set of tasks and subtasks associated with that specific mission, therefore executing the proper actions when necessary.

In this instance, the mission is "determining foreign influence." The mission is one mission that would be associated with intelligence groups, along with "gather intelligence," "conduct reconnaissance," and any other missions added. Within this specific mission, there is a further breakdown of potential subtasks. In this instance the identified subtask is PCAP analysis, in which the intelligence team will examine PCAP information for a given area.

The next section of interest is the "session event" section. Overall, this section visually represents the criteria that determine when a specific mission task starts and concludes within different designated sessions. The left column within the section represents the generic working of the session event. The selected mission is designated, followed by the criteria to initiate and terminate the mission. The top box represents the start of the task while the bottom box represents the criteria for ending the task.

As seen in the right column, the specific mission selected in this case is the "PCAP analysis" as part of "determine foreign influence." The top box designates the initiation of the task, while the bottom box denotes that it will conclude at the termination of the specific PCAP stream.

"Task objective" is the next section of interest. This section designates amplifying tasks and decisions that contribute to the overall mission task within the mission. This again is not limited in number as each mission task can have multiple task objectives. A concise way to view the task objectives are by specific process execution and for what the goal of the process is.

In this case, the task objectives are to execute specific mission tasks and to determine amplifying network information. Specific mission tasks may include "see all traffic on

network" and "filter for type of traffic." Amplifying tasks may include determining the health of the network, who is on the network, and the ID of the network.

The "decider" section is broken down into two subsections, "decider information" and "decider course of action." The information delineates the types of information that should influence the recommended decision and the source that the information should come from. The decider course of action states the possible recommended courses of action that the machine can recommend based on the defined criteria.

In this example, decision information includes the three realms of influence which include cyber, real-world, and network. The cyber domain consists of XMPP data which will be retrieved from the OpenFire chat server. The real-world data refers to Open-Source Intelligence (OSINT) and Personal Activity Intelligence (PAI) data that is collected. Network data refers to information specific to the physical network. The decision course of action for this specific example is to recommend that the alert status be set to normal—green, heightened—yellow, or emergency—red.

3.1.3 Hypergraph

A hypergraph allows conceptual design and view of systems and information. The hypergraph depicted in Image 3.3 is a high-level view of the main entities and the interplay of influence between XMPP and PCAP, PAI, OSINT. This model is an in-depth analysis of the "decider information" section presented in the hierarchy. This specific graph shows the interactions of the three influential categories, real-world, cyber, and network, as well as different aspects that comprise each category. The "network" category is likely the most straightforward, but nonetheless important. The network category is comprised of physical components that would make up networks, such as routers, switches, servers, and firewalls. The purpose of this category is to provide context as to what is occurring with the physical components in relation to other events.

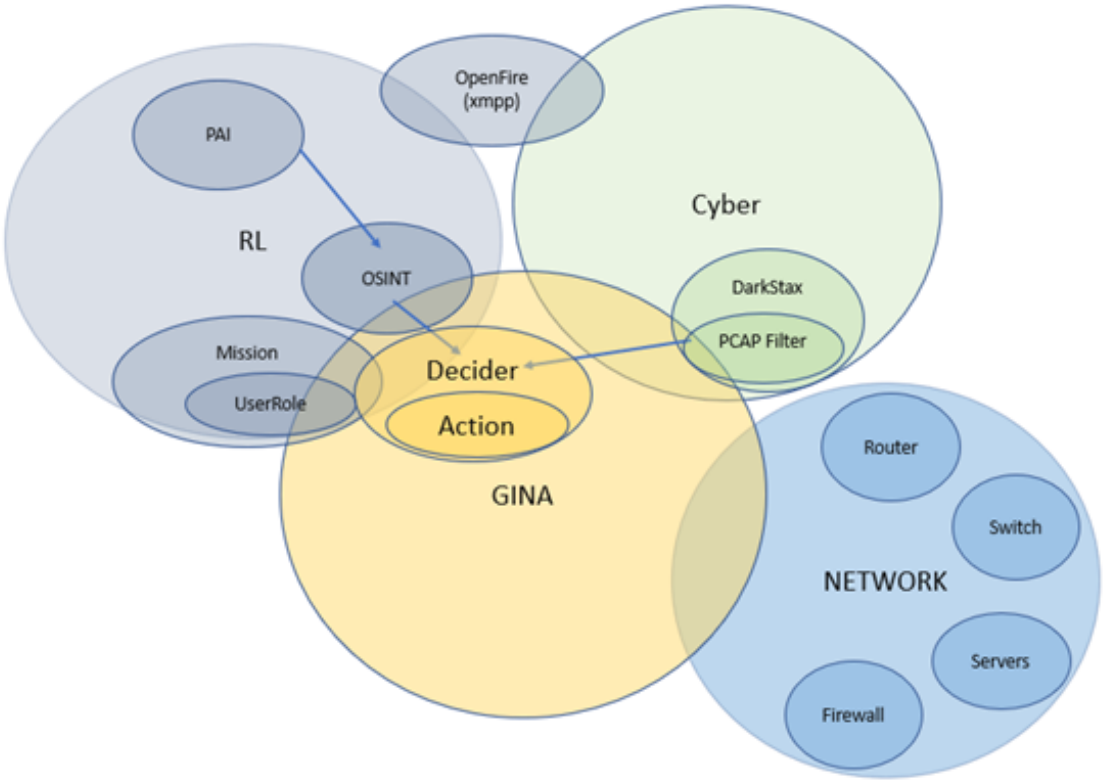


Figure 3.3. The hypergraph depicts the key entities that comprise the GINA decision model. This is a breakdown of the "decider information" section depicted in Figure 3.2. The three main domains of influence are real-world, cyber, and network. The model in this thesis will be comprised of only a part of the cyber category, specifically XMPP traffic. These three domains should be considered a starting point for a decision model designed for large scale network diagnostics.

The "real-world" domain is arguably the most dynamic because it can be influenced by a multitude of dimensions and events. In this depicted model, the specific real-world influences depicted are OSINT, PAI, and mission. OSINT refers to all events that may have an impact on the designated network, either intentionally or by circumstance. The potential immensity of this area is likely to not only overwhelm decision makers, but also prevent decision makers from seeing all events that affect the mission and therefore make decisions without the most

accurate picture. Including this section enables the ability to determine the impact of events, calculate the significance of the event both individually and in combination, and present that summary to the decision maker. The PAI subsection is of similar characteristic as OSINT. The domain itself can be vast and therefore misunderstood or improperly accounted for. The model enables a more organized and more accurate description of the effects PAI may have. The mission section gives context to the decision makers intent and the purpose of aggregating the intelligence. As stated in the graph model section, different missions will have different subtask and goals. The context provided allows the model to determine what OSINT, PAI or other real-life scenarios affect the mission the most, as different missions are affected differently by different events. The mission context seen here allows the model to properly weigh the events and determine the overall effect of the information to the given mission. Of further note, within the mission subcategory is another category called "user role." This is used within the system to affect privileges and actions. The user would be associated with a unit, as part of the hierarchy, that has correlating missions. The specific role the user associates with would have a varying level of permissions, based on the impact the role has to a specific mission.

The last domain, "cyber," captures all cyber related events and determines the potential impact of the captured criteria. In this model, the primary source of cyber information is the PCAP stream from the network at hand. As discussed in the hierarchy section, the PCAP from the entire network would be taken in to DarkStax. Dark Stax would then filter the PCAP based on a defined set of parameters or guidelines in order to present via pubsub the PCAP as readable information to a GINA service for the model to then make a decision as to the impact of the information.

Also visible in this depiction is the existence of XMPP data from the OpenFire chat server, the chat server used by the Marine Corps at MWX. The XMPP data is a relevant part of both the real-world and cyber information sectors. The existence of the messages will be seen in the PCAP data filtered by Dark Stax, as part of the cyber section. For this given exercise, acXmpp messages are created to represent OpenFire traffic that would have been filtered. For purposes of this research, the model will only be concerned with receiving data in motion to the GINA service and analyzing the acXmpp messages. The description of this hypergraph shows how this use case fits in to a larger scenario of determining network health.

3.2 Thesis Focus

The problem that this thesis aims to solve is the capability gap of achieving data-to-decision-to-processed actions.

The specific use case focus is the battlefield network monitoring of XMPP chat for network diagnostics. The framework depicted above portrays a specific, while complex, scenario regarding large scale network diagnostic classification in live USMC networks. The scenario proves validation of the model and is an example for federalizing data and centralized multi-domain decision making. However, this use case is only a validation of the model itself. It is of utmost importance to understand that the model is not at all limited to that scenario. The scope of this thesis is to develop a ubiquitous decision model in the GINA framework that would be useful in multiple scenarios. The key focus of the model is to transform data to knowledge, knowledge into decisions, and decisions into actions.

The decision model uses truth table type logic to assign data to any mission, thus extrapolating the functionality of transforming data into action. The model focuses on achieving the ability to take action, rather than simply graphing various information together. The thesis demonstrates and validates the model using a simulated practical operational network setting to create real time behavior based on data sensed from the network.

The key contribution of the thesis is a validated methodology for using truth table type logic to assign data to any scenario. This should provide universal appeal for all missions.

3.3 Methodologies

This section discusses the design methodology, implementation methodology, and testing methodology that will be executed in order to build and validate the ubiquitous decision model. The decision methodology is to use hypergraphs to explain how concepts relate. The specific focus is on the relation between data to a decision using the relation between a data Concept, truth table Concept, and subsequent status Concept. The implementation methodology is to use the GINA construct, specifically focusing on the application model, to develop all necessary concepts and relations. The testing methodology will be to simulate network payload and use a cursor on target simulator to send data to the system for processing.

3.3.1 Design Methodology

This section discusses the logic behind the proposed decision model construct. Overall, hypergraphs are used for high level design and to visualize what are key concepts and component interactions. An example of this can be seen in section 3.1.3 where a hypergraph depicts the large networks diagnostic model that this decision model will help solve. The specific purpose of this decision model is to transform data into decisions. Figure 3.4 illustrates the relation between three Concepts that transforms data into information and decisions with the ability to cause action, i.e. automatic processes.

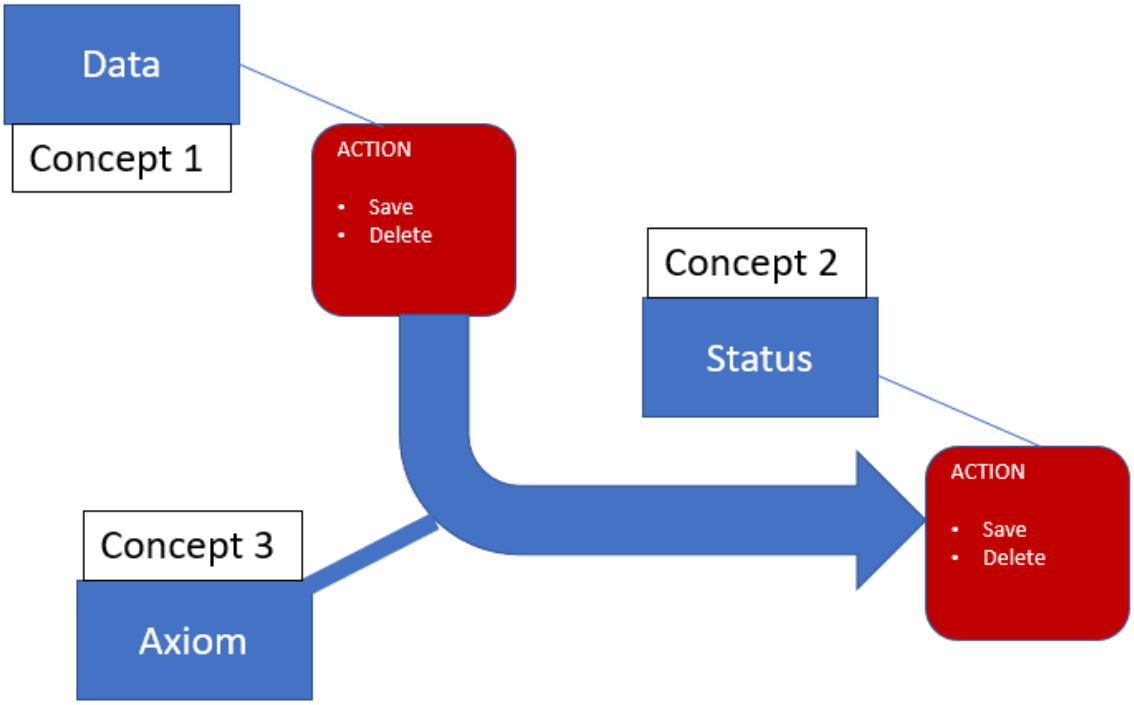


Figure 3.4. This image portrays an abstract view of the proposed model design. Three concepts are used to enable the data to decision and transformation to action.

The three Concepts represented in this model are a Data, Axiom or truth table, and the Status Concept. Using these three Concepts and associated relationships, the number of decisions that can be made on the data input is unconstrained. For this use case, Concept one is the

XMPP Data that receives the input messages to a GINA service. Concept two is the Status Concept that contains the history and updated information to capture status of new messages. Concept three is the Axiom, a truth table that contains all possible information states and establish knowledge correlating to the state driven axioms. The Actions of Concepts Data and Status are sequentially linked through specification of Linked Action. As Data makes a decision and is actuated to 'save,' Concept two is sequentially actuated to implement the decision data into its 'save' or executable process. Since Concept two can make a history log of the actions and statuses, it has data to create new information and new decisions that may be applied to influence its own system behavior back on Concept 1, i.e. over watch intelligence.

This methodology can be expressed in further detail by referring to Photo 3.5. This image represents the full decision model, encompassing the process of data to information to decision to action. To make the decision model function properly, only three Concepts are needed, Data, Axiom (truth table), and Status. This model is abstracted so any mission objective Axiom (truth table) and mission objective Status can be addressed, as the Mission and Objective Concepts (not shown here) provide modeled information management and prevent data overload on the user. In other words, a Mission has an Objective that has a Status. The Data has attributes that may be logically determined into information state attributes and related to an Axiom for added knowledge. The Action of 'save' on Data sets in motion the movement and transform of data elements to the Status Concept for saving in specified content and form. It is noted that instead of saving to the Status Concept, a command line could also be assembled and executed with the data and information from DATA.

It is important to describe the separation in the decide and act states. The relevant components to a decision are the data Concepts and related knowledge Concept of state-axiom. These two Concepts alone provide enough information to make a decision. The data Concept creates information based on the instantaneous first layer of analysis. In this case, the first level of analysis is Natural Language Processing (NLP), but that can change based on scenario requirements. This first level of analysis turns data into information.

This is where logic or transformations may be applied to the data columns. When data and information are combined with the axiom Concept, the truth table, knowledge is accessed.

This is the state of Graph technology, and in itself provides tremendous insight to decision making and complex systems.

Data to decision, including the transformation of data to information and knowledge, may happen with the just the Data and the Axiom model Concepts. Here, however, GINA uses the abstracted notion of action and sequentially links actions from the same or other Concepts. GINA also enables transformation of data that is passed from one Concept's Action to another's. In this model, linking the Data Action 'save' to the Status Action 'save' sets in motion the ability to move the decision and knowledge to separate Action 'save'. This sequence of saves on multiple Concepts adds complexity of data mapping and transformation required for any execution or update of any table. In GINA, this complex series of saves, transforms and data specifications is more easily described semantically as a named behavior, e.g., "AlertAnalyst." The result is an intelligent automation, with no limit to the steps or linked actions and complexity.

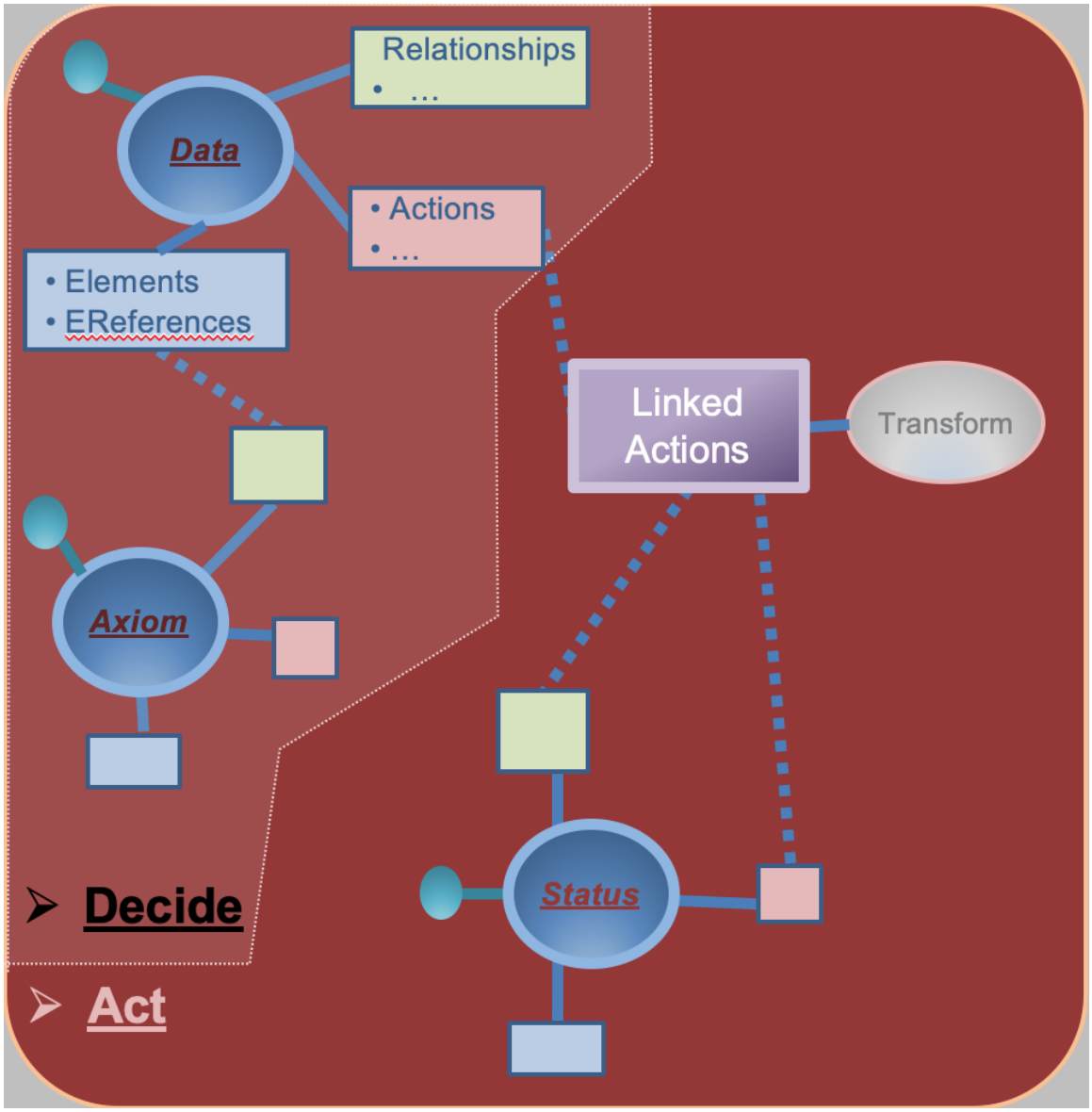


Figure 3.5. This image depicts a more specific portrayal of the proposed design methodology. The decision portion of the model is supported by the Data and Axiom Concepts. The Linked Action coupled to the Status Concept enables the possibility of execution and automation.

3.3.2 Implementation Methodology

To effectively implement the ubiquitous decision model, the discussed design framework is implemented in GINA. This section discusses the implementation methodology used to successfully build the model.

As review, Image 3.6 and Image 3.7 depict the GINA framework structure. Image 3.6 depicts that GINA exists in a secure enclave and offers tailored capabilities to different users. The entire framework consists of user and data interface, which is comprised of web servers and services; the application model, which is characterized by facilitating a zero trust architecture, executable processes, configured conceptual objects, and federated graph based applications; and the storage servers, which take advantage of Microsoft security features.

This entire framework is used to implement the decision model, though the specific focus is on the application model within GINA, depicted in Figure 3.7. The application model consists of application organization constructs, conceptual constructs, middleware constructs, and user-interface constructs. The most significant is the conceptual constructs. Each Concept in GINA is referred to as an Xtype and is inherently linked to an Action, Source, Vector, and Element. This provides the structure to implement the proposed three Concept decision model. Each required Concept is created as an Xtype, while all necessary relation components are created as the associated construction objects.

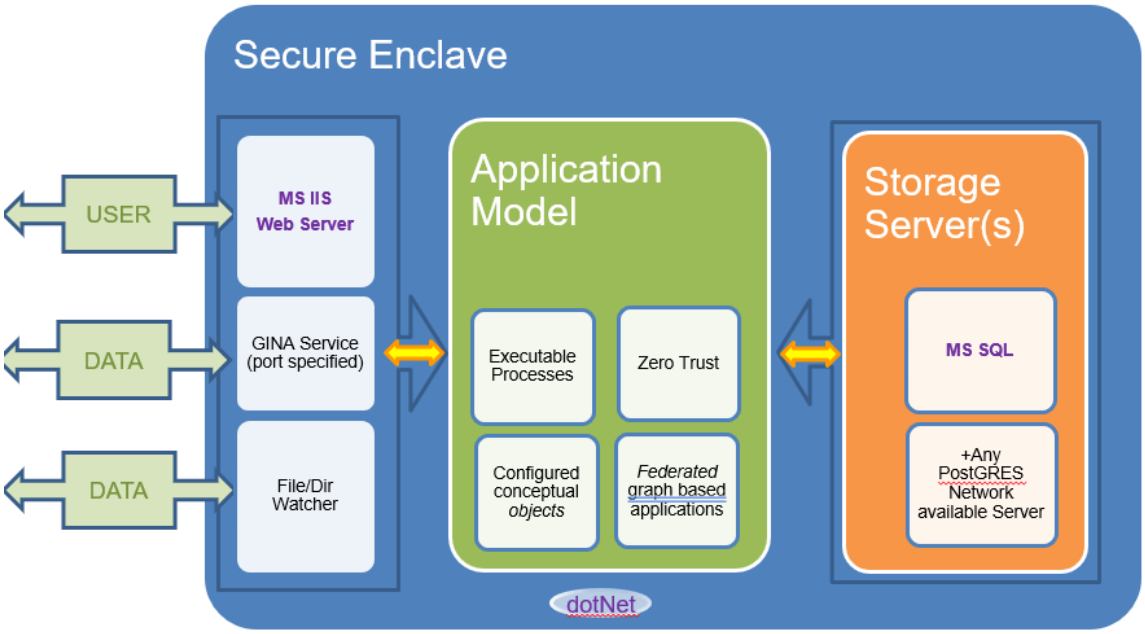


Figure 3.6. The functional components of the GINA framework are shown above. GINA is a secure enclave that is web accessible for configuration, development and execution. This thesis focuses on configuring a decision model that exists within the GINA application model.

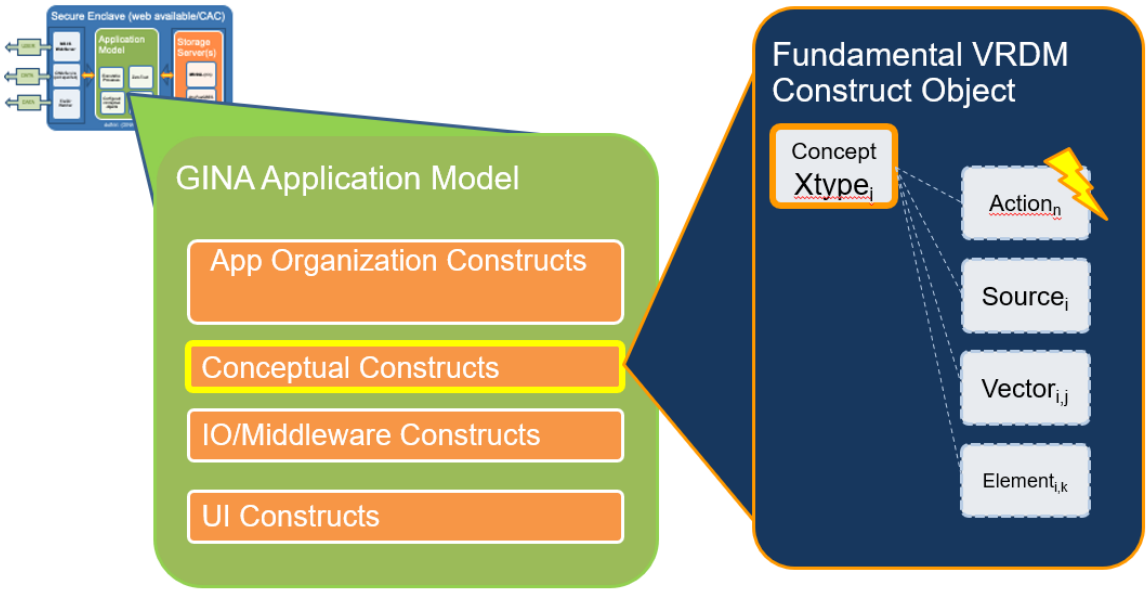


Figure 3.7. This is a breakdown of the functional areas for the GINA model of application. The GINA model of application consists of: organization constructs, conceptual constructs, middleware constructs, and user-interface constructs. The constructs are made from foundational conceptual objects called Xtypes that are inherently linked to GINA model components Actions, Sources, Vectors, and Elements.

To better describe the functionality of the model and the connection between the design and implementation, all aspects of Image 3.5, presented in the design methodology section 3.3.1 are described here. Image 3.5 represents the three Concepts necessary to construct the decision model. Concepts are the building blocks of the GINA framework. These three Concepts comprise the decision model of this GINA application. The Axiom Concept and Data Concept comprise the decision portion of the presented model. Each GINA Concept has associated Vectors, Elements (attribute), Actions, and a Source. Data is transformed to information using logical constructs on specified Data Elements, creating new Data Elements, e.g., 'isLow,' an attribute of type bit that does a string comparison on the Data Element 'payload,' thus achieving an information state. Then the Element Reference "objectiveState" is added to the Data Concept. This is informed from the Axiom (truth table)

values as determined by Vector relationship using the NLP driven information state of specified Data Elements, e.g., 'isLow.' The right hand side of the image, which includes the Status Concept, represents the "act" portion of the model. As discussed earlier, the execution of a list of Actions enables process and automation. Further, the logging of the data and information that is saved to Status creates an opportunity for over watch and learning behaviors, i.e. data to be analyzed and applied to later action. These elements, along with the Transform, "Tx," element, that may alter or add to data in motion e.g., regex, allows any prescribed action or command line execution to take place.

GINA Construct in UML

The GINA base model construct is seen in Image 3.8, which is a limited view of the GINA application model construct introduced in section 2.3.6 represented in UML, implemented by Dr. Thomas Anderson using the StarUML platform [12]. The purpose of this figure is to provide a visual representation of the base GINA components discussed in above, seen in images 3.6 and 3.7. This representation offers an model and data architecture view of the base GINA components in a UML structure, which can be expanded to enumerate the verbose configuration of an entire models. This is critical for creating engineering artifacts to document and describe the construction of GINA model based applications.

For reference, the four functional constructs of the application model exist as components of the base UML model. The conceptual constructs consist of Concepts inherently linked to Actions, Sources, Vectors, and Elements, represented by red, yellow, purple, and blue boxes respectively. The middleware construct is represented by the Source and Column. The user-interface constructs, that configure the web interface, are represented by the Forms, Windows and Field. The application organization constructs are represented by the Application, Module and Model. The UML construct as a whole is the GINA model of application. As each box in the UML represents an object class, to make a new application, one creates a new item by enumerating a new instance in the desired object. The attributes of each component have sufficient information to provide context required to inform and execute a particular application and model. This model based approach is utilizing an underlying information system, where each conceptual object has attributes saved to an underlying table. Further, the model may be navigated on with a web browser for both development and execution of applications.

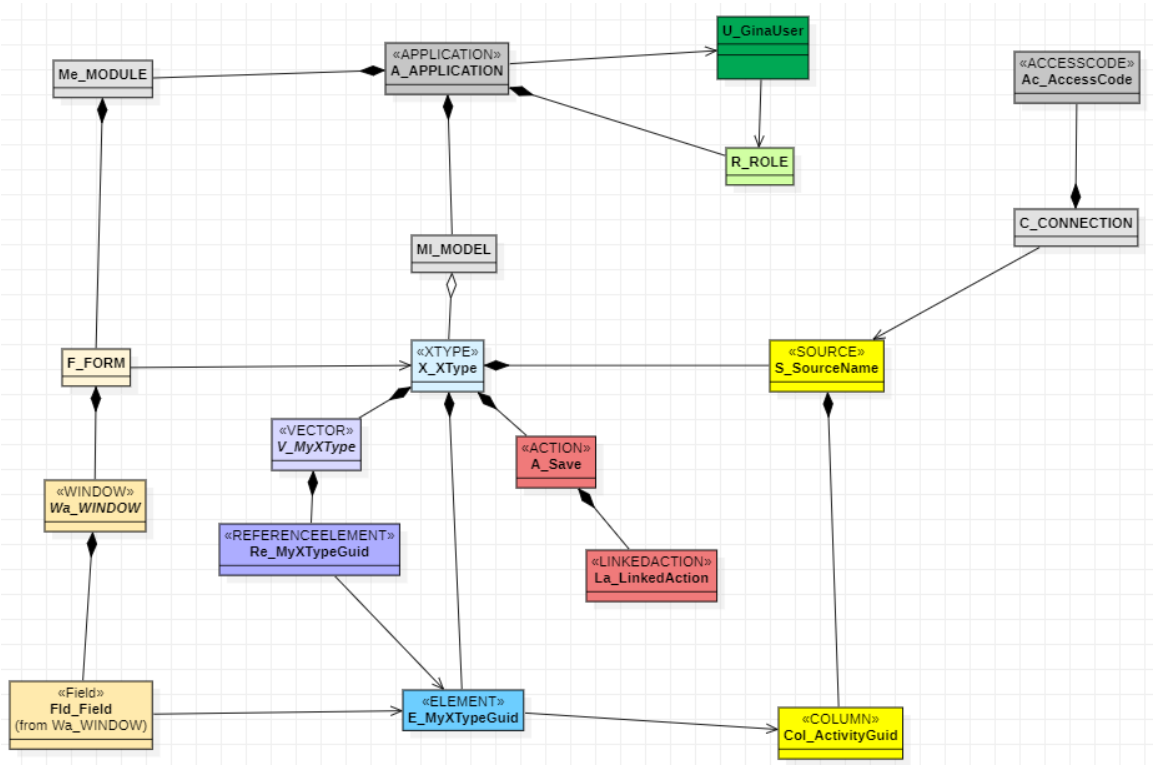


Figure 3.8. GINA Base Model represented in StarUML. The model constructed in this thesis will use this as the starting point to construct the implemented model.

The final model will be a configured instance of this structure and components, and may be considered as a 'coupled model' in that each component configured for this decision model is inherently linked to the constituent GINA construct from which it is instantiated. At any point in the decision model application, the developer may directly access the underlying construct for modification or verification. The implementation will occur in the GINA platform, and a visual interpretation will be constructed using StarUML for architectural evidence and reproducibility. While the UML model serves as a visual aid and layout for this research it does save its design in JSON format. USARMY United States Army Network Enterprise Technology Command (NETCOM) is working on (in progress 2022) a GINA Application Programming Interface (API) that can take this UML design as input, thus

creating a design-to-implementation with no coding.

Full Model in UML

Having discussed the base GINA framework in acUML, the proposed decision model logic described above is represented in the same manner in Figure 3.9. The model includes the three Concepts necessary, Data, Axiom and Status, and the associated Elements, Vectors, Sources, and Forms that comprise the Concepts, as well as the necessary relations between the Concepts. This is the model that will be implemented in the GINA framework, discussed in Chapter 4.

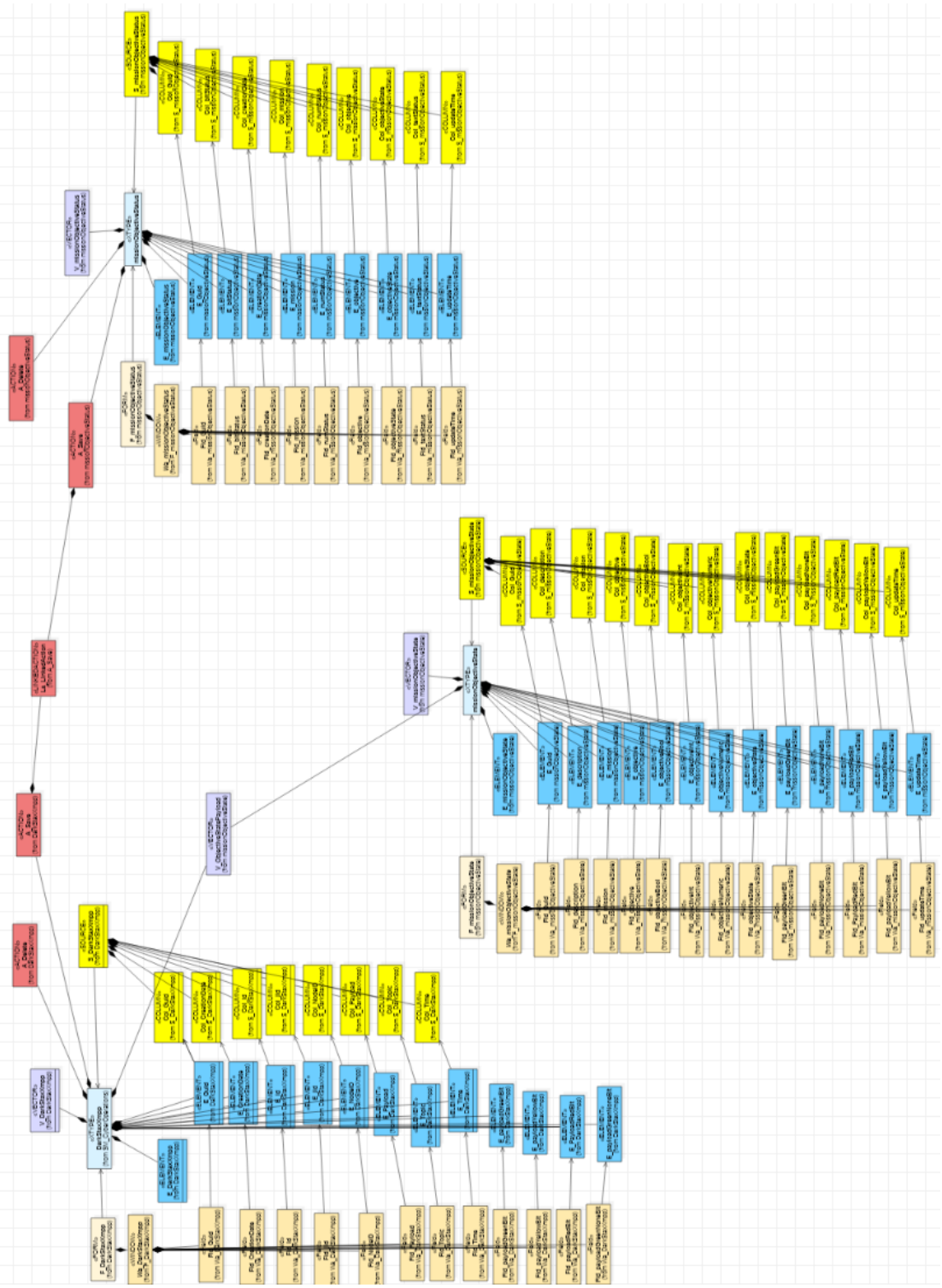


Figure 3.9. This UML representation includes all necessary components of the decision model to be implemented in the GINA framework. While the model appears crowded, the structure is consistent and acts as a guide to implementation. Further, it may be collapsed to emphasize particular components or relationships.

The figure depicts the three Concepts in the same order as Figure 3.4. Concept one is the "DarkStaxXmpp" Concept which acts as the Data Concept. It includes all attributes associated with the Concept listed in the blue boxes, to include "DSXmppGuid," "CreationData," "id," "jid," "NodeID," "Payload," "Topic," "Time," "payloadGreenBit," "payloadYellowBit," "payloadRedBit," and "payloadNoneBit." It also includes the associated pertinent fields and columns relating to the elements. Also visible is the self-reference vector and self-reference element.

Concept two is the "missionObjectiveStatus" Concept which acts as the log of previous actions and status. It also depicts all required Elements, associate Fields, associated Source Columns, and the self-reference Element and Vector. The elements necessary for the status Concept are "mObjectiveStatusGuid," "bitStatus," "creationDate," "mission," "numStatus," "objective," "objectiveState," "textStatus," and "updateTime." The "numStatus" and "textStatus" fields are not used for this research. They are included as prepositioning for future use cases.

Concept three is the "missionObjectiveState" Concept which is the truth table or Axiom Concept. It includes the same features as the previous Concept. Necessary Elements include "mObjectiveStateGuid," "description," "mission," "objective," "objectiveBool," "objectiveNumeric," "objectiveInt," "objectiveState," "payloadGreenBit," "payloadYellowBit," "payloadRedBit," "payloadNoneBit," and "updateTime."

The Actions and Linked Actions that relate the "DarkStaxXmpp" and "missionObjectiveStatus," and the "objectiveStatePayload" Vector that relates the "DarkStaxXmpp" and "missionObjectiveState," is also visible in the diagram. Chapter four will explain how to implement the entire model.

3.3.3 Testing Methodology

The methodology used to test and validate the model with the XMPP use case is to use simulation based testing to observe output against known labeled input and verify proper functionality.

In order to definitively verify the model, the test case is designed to be a simple interpretation of the parameters. The intent is for this test case to be straightforward and not subject to

misinterpretation. Completion of this exercise will prove the model functions in terms of input, output, and decision recommendation from given criteria. This alone shows that a ubiquitous model was created in the GINA platform and functions properly. However, it alone offers little in terms of context or operational applicability.

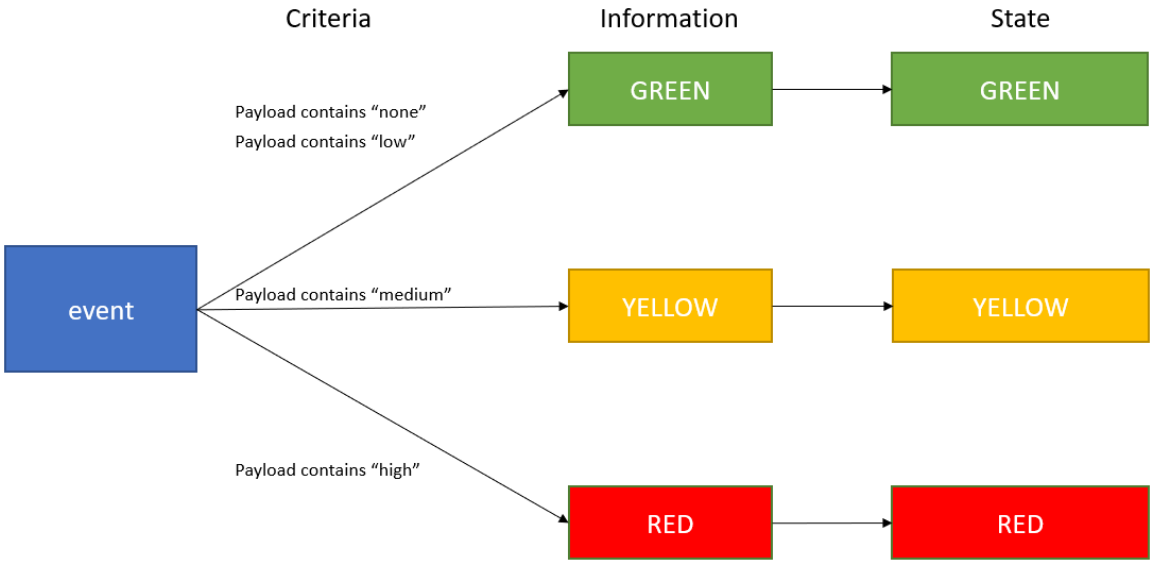


Figure 3.10. This is the proposed test scenario to conduct during implementation. It is intentionally overly simplistic to ensure the model works as intended. Given that this model is intended to be used as the base for other models, it is vital to ensure functionality before adding complexity.

Data Input The experiment is designed to accept both XML and JSON formatted messages. For the simplistic scenario, XMPP messages were created organically and will be the only input to the model. The input will not result from PCAP filter or any complex data cleaning. In line with the goal of simplicity, the model will only take four messages as input. Each message will meet the criteria of a different expected output. The four messages will be the same except for the payload category. Four messages will be input, and a decision will be made for each.

Criteria For this specific test, simple criteria were created to prove functionality of the model. The content of the payload is the only section that will affect the decision. The

simplistic criteria is seen as follows:

- If Payload contains “traffic is none,” then the status is green, truth table determines that objective state is green.
- If Payload contains “traffic is low,” then the status is green, truth table determines that objective state is green.
- If Payload contains “traffic is medium,” status is yellow, truth table determines that objective state is yellow.
- If Payload contains “traffic is high,” status is red, truth table determines that objective state is red.
- If Payload does not contain key words, status remains the same.

Output The output is the objective state status of the network. The status of the network should be “red,” “yellow,” or “green.” Based on the determined alert status, a specific course of sub-actions could be recommended. For this simplistic model, the actual sub-actions are not of concern. The model stops at the path to different notional sub-tasks. Since the input consisted of labeled data, determining if the model correctly classifies input will be straightforward.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 4:
Implementation

This chapter represents implementation of the ubiquitous decision model that was created in GINA's VRDM construct. The chapter describes the overall approach of implementing the model and will document the steps needed to do so.

The first section depicts the entire model and discusses the interconnected Concepts. The next section depicts a step-by-step instruction on the procedure to create one of the Concepts depicted using the GINA platform. Since the Concept implementation is the same for all Concepts, only one will be shown, but it is imperative to remember that the steps need to be followed for each Concept.

The decision configuration section then breaks down how each of the three Concepts relate to each other as well as implementation instructions. In addition to the three Concepts, the section discusses two relational aspects necessary for the decision functionality and provides instructions to configure the relations. The section also goes through a step by step of how to implement the input channel that will feed the data into the model.

The final section will depict how the input channel was configured. This was the configuration that allowed the model described to intake and analyze the network traffic. It is the connection between JSON and XML data and the GINA decision model. For this specific experiment, the JSON and XML messages were created organically. However, the input channel is capable of absorbing massive streams of data generated by live networks.

4.1 Model Creation

This section displays the overall model utilized and highlights the interactions and functionality of the Concepts. This is an expansion on the methodologies presented in the methodologies section. Following the description of the entire model built, the section will provide a step-by-step guide to instantiating a Concept in GINA.

To reintroduce the model, Image 4.1 shows the implemented model. To reiterate the overall

Concept, the three Concepts work together to create a truth table based decision model that turns information into action. As an overview, “DarkStaxXmpp” is the Data Concept and serves as the entry point for the XMPP data upon being ingested JSON or XML though a GINA service (IP and Port) and conducts initial analysis through NLP and logical operators. This permits capturing data, then transforming to information. The new information is stored in Elements that are added to the Data Concept and are evaluated just-in-time. A comparison is made of the incoming XMPP messages that determines the true or false values of certain bit states. For example, a Data Element of data type bit, called “isRed” is created which evaluates the incoming Data Element “Payload” that contains the text chat. The “MissionObjectiveState” is the Axiom Concept or truth table that relates the Data Elements (e.g., bit states) with the known meaning, in this case the “objectiveState.” Based on the state of the various Elements, the corresponding “objectiveState” is determined and then shared through the “objectiveState” Vector. The specific message instance is thus associated with that objective state. These instances are saved by the XMPP Concept, and also saved by the “missionObjectiveStatus” Status Concept, which acts as a log for all previous inputs and associated “objectiveStates.” While this model merely updates a status table, any number of updates or command line executions may be performed as well.

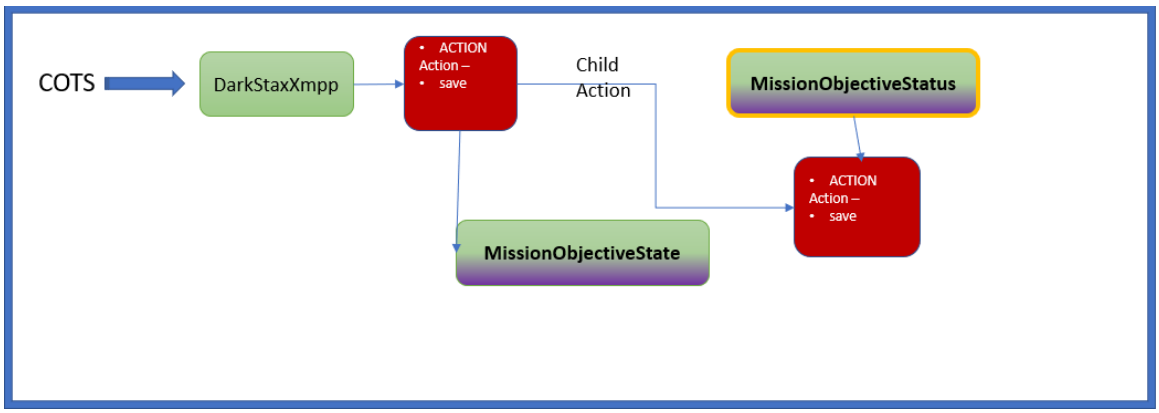


Figure 4.1. The image portrays the design of the decision model being implemented. The implementation puts this framework into the GINA construct.

4.1.1 Concept Creation: Step-by-Step Guide

All Concepts instantiated in this model follow a similar set of instructions. Since the implementation is repetitive, step-by-step instructions are provided for only one of the three forms. However, it is important to remember that all interconnected GINA objects must be instantiated using this procedure. The implementation demonstration will include the start at the "back-end" Structured Query Language (SQL) database up to the finalized form in GINA. The example shows the implementation of the "missionObjectiveState" Concept. This example assumes that model components of Application, and Model have already been instantiated (configured) and thus provide named context when needed, i.e. MyConcept belongs to MyApplication and is in MyModel.

The first step is to establish the backend of the Concept to store data using a table in SQL. GINA can access any PostGRES information system that is web available, however, for this example, we will create the table in MSSQL Server using a script. The code can be seen below:

```
USE [CyberOperations]
GO
/****** Object:  Table [dbo].[missionObjectiveState] ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[missionObjectiveState](
[guid] [uniqueidentifier] ROWGUIDCOL  NOT NULL,
[objective] [nvarchar](50) NULL,
[mission] [nvarchar](50) NULL,
[updateTime] [datetime] NULL,
[objectiveState] [nvarchar](50) NULL,
[objectiveNumeric] [float] NULL,
[objectiveInt] [int] NULL,
[objectiveBool] [bit] NULL,
[description] [nvarchar](max) NULL,
[payloadRedBit] [bit] NULL,
```

```
[payloadYellowBit] [bit] NULL,
[payloadGreenBit] [bit] NULL,
[payloadNoneBit] [bit] NULL
) ON [PRIMARY] TEXTIMAGE_ON [PRIMARY]
GO
ALTER TABLE [dbo].[missionObjectiveState] ADD  CONSTRAINT
[DF_missionObjectiveState_guid]
DEFAULT (newid()) FOR [guid]
GO
```

Once the code is input into the query section of SQL Server, it is ran using the "Execute" command, displayed in Image 4.2.

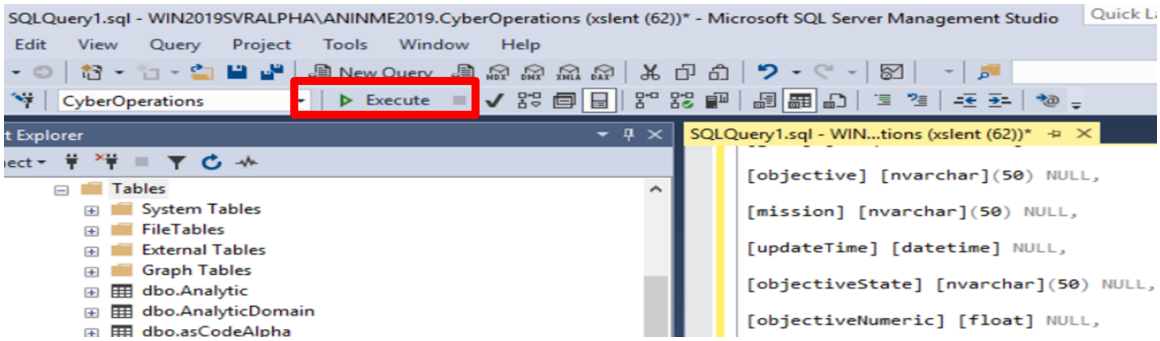


Figure 4.2. Execute command in SQL

This should create a table that exists in the SQL Server database, as seen below in Figure 4.3.

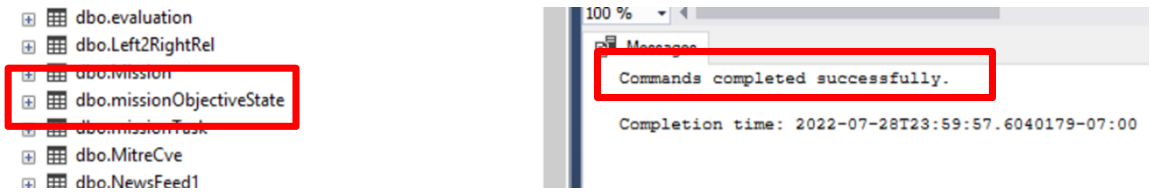


Figure 4.3. SQL notification for successful table creation

Once the table is created in the database, the next objective is to link the table and the GINA database server. Fortunately, there are three pre-made scripts for this task. These scripts are "CreateSourceForTable_Step1," "CreateXtypeForSource_Step2," and "CreateFormForXtype_Step3." They are located in the "anime722_dss" database, under the "programmability" section, in a folder called "stored procedures." The contents of the stored procedures are included in the appendix. The above instructions are displayed visually below in Image 4.4.

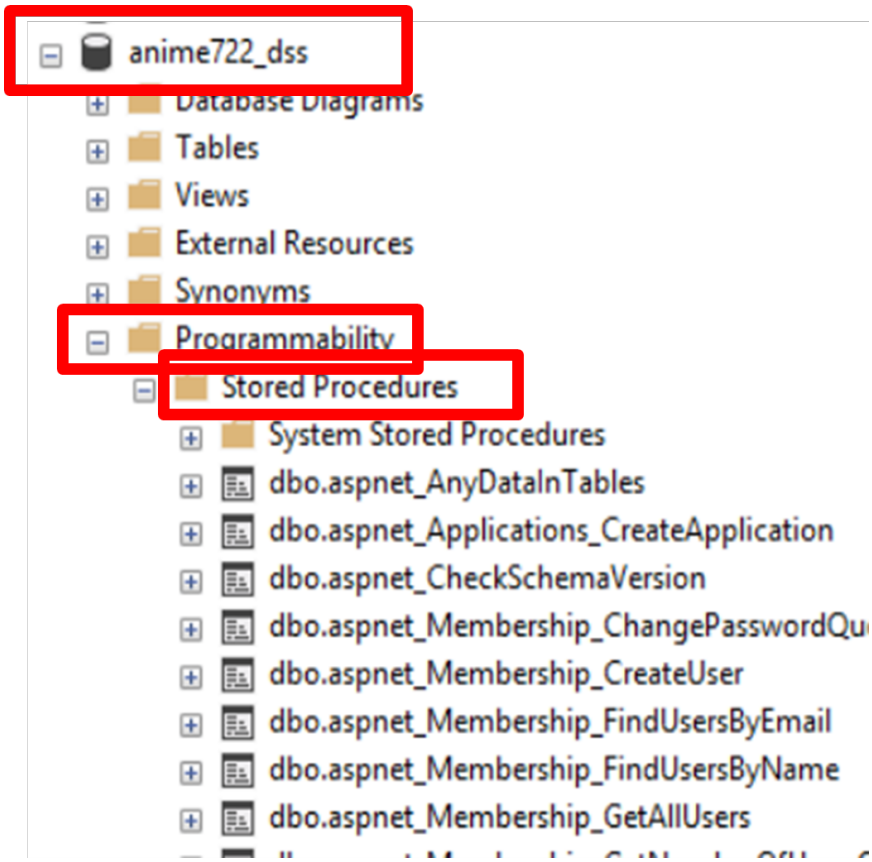


Figure 4.4. Stored Procedure Location

The first step to instantiating a Concept in the GINA framework is to create the GINA Source and the Source's Columns for the Concept by executing stored procedure annotated Step1. The purpose of the stored procedure is to configure the GINA Source and the GINA Source Columns for the specific Concept with pertinent information such as database connection information, table name, and unique identifier. Locate the "dbo.CreateSourceForSqlTable_STEP1" option, and then click "Execute Stored Procedure."

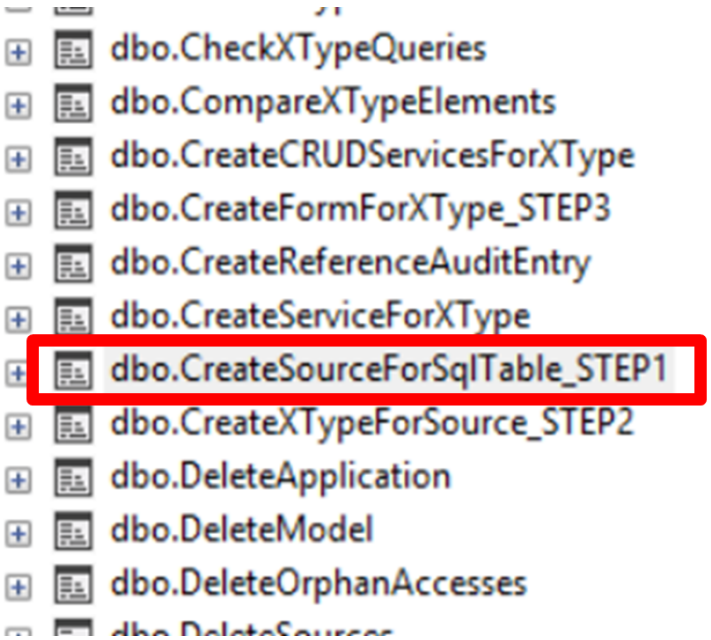


Figure 4.5. Stored Procedure One Location

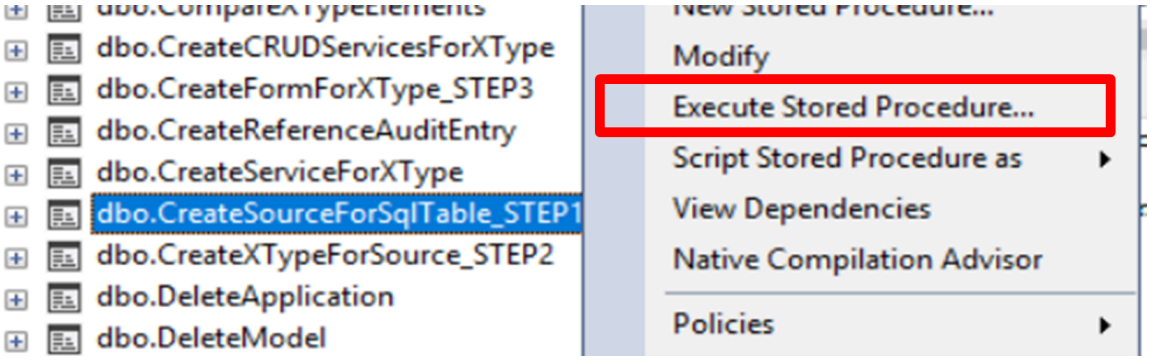


Figure 4.6. Stored Procedure One Execute

This will bring up an input table with six information fields required to create the Source and Columns. The naming convention is that the Source name should be the same as the table name. Here, Model represents the parent model in which the Concept is being

instantiated, which in this case is “cyberOperations.” In this case, Connection name is the same as Model name, as a GINA Connection was established explicitly for a database called cyberOperations that was constructed to hold all appropriate data. Table is the name of the table being referenced in the database. The unique key must be specified, usually as “MyConceptGuid.” The create field is always “yes” to indicate the desire to create the table. An example can be seen in Figure 4.7.

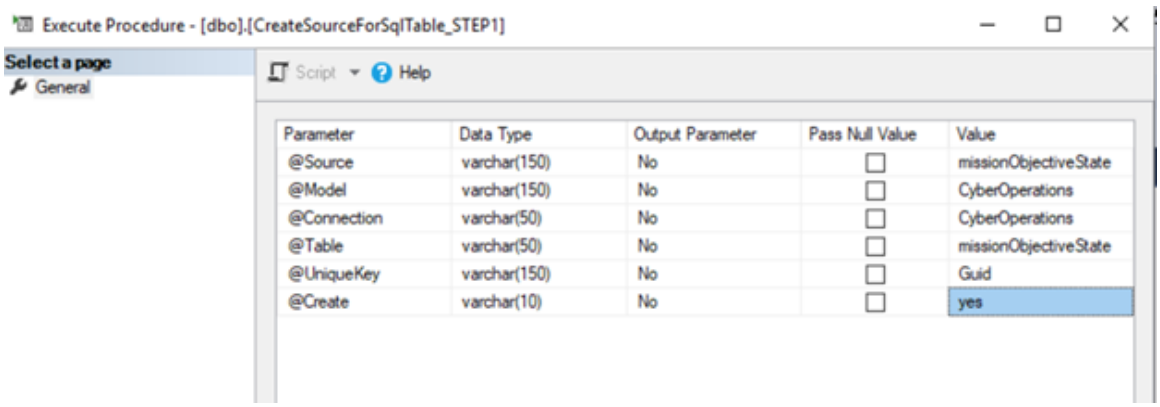


Figure 4.7. Stored Procedure CreateSourceFromTable_Step1 finalizing instructions

Executing the stored procedure produces the following result, with the SQL script contents based on the specified table:

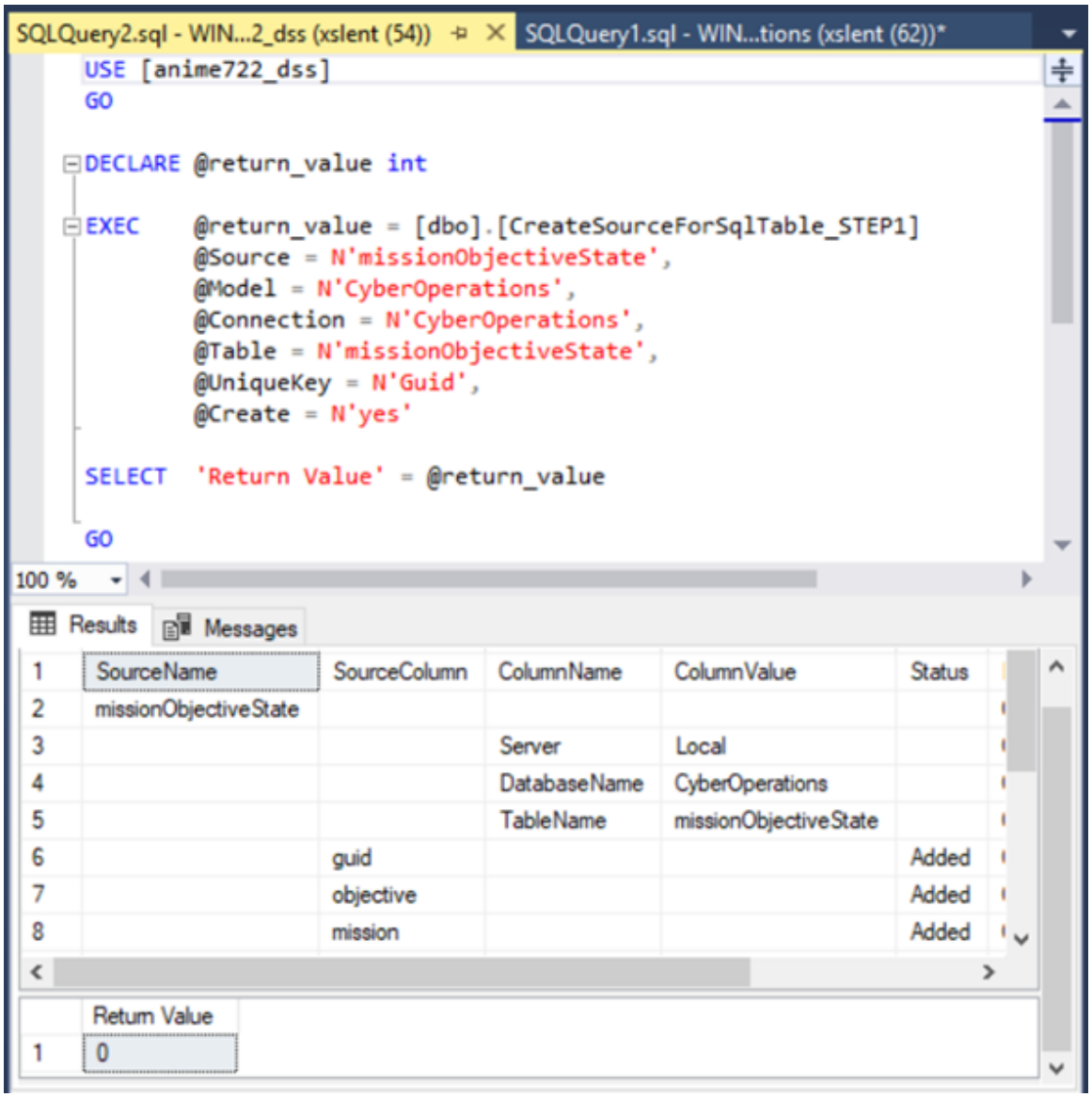


Figure 4.8. Stored Procedure CreateSourceFromTable_Step1 success view in SQL

Upon completion of step one, GINA model components entailing a Source named 'missionObjectiveState' and all respective Gina Columns that reflect the table columns are configured in the GINA Framework. The same process can be followed for the stored pro-

cedure in step two. The purpose of this stored procedure is to create and configure the "Xtypes" and respective "Elements" for the model.

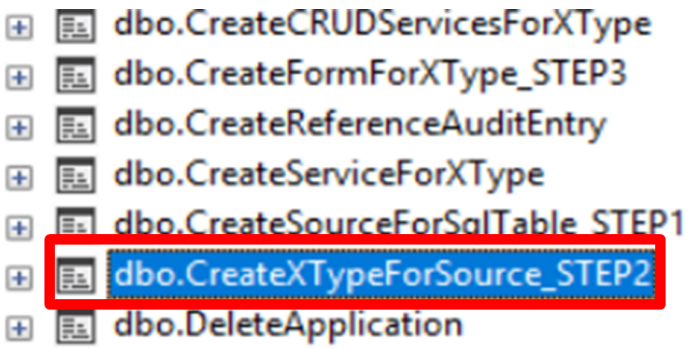


Figure 4.9. Stored Procedure Two location is easy to find in the dss stored procedures due to the annotation of "STEP2."

Upon execution, the associated input table should be filled in as follows:

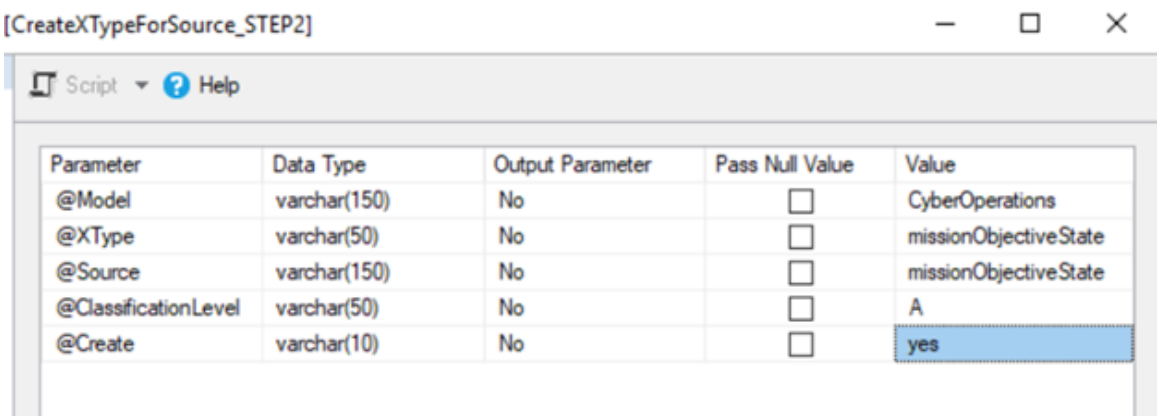


Figure 4.10. Stored Procedure CreateXTypeForSource_STEP2. finalizing instructions

The "model" should represent the model or application in which the work is being done, which in this case is "CyberOperations." The "Xtype" and "Source" should each con-

tain the same name as the table created, in this case “missionObjectiveState.” Lastly, the “classification level” should be marked as “A” which is used to signify unclassified material.

The result should look similar to the following:

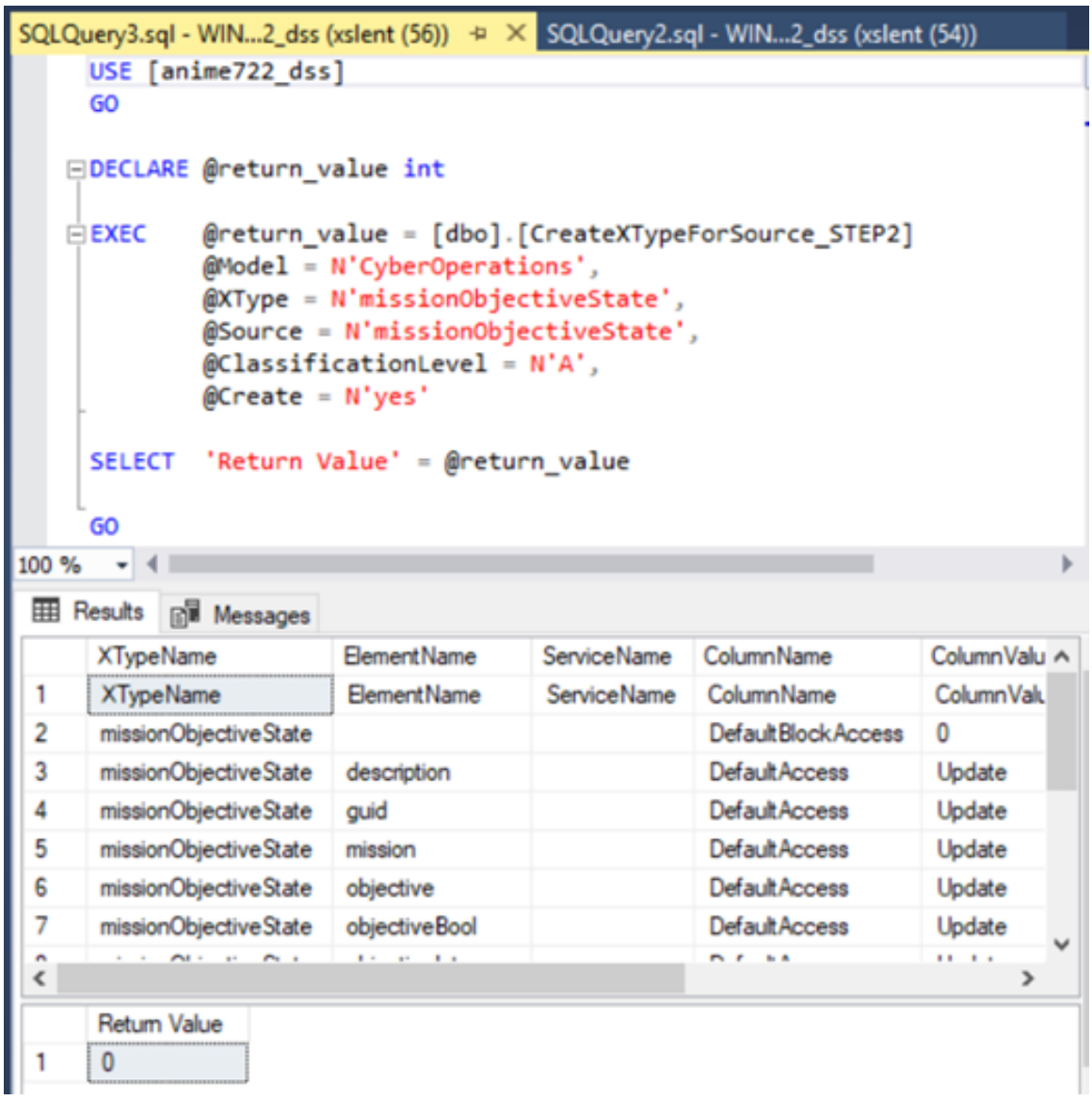


Figure 4.11. Stored Procedure CreateXTypeFromSource_STEP2 success view in SQL

Before the final stored procedure can be executed in SQL, preparatory work must be done in the GINA framework. The self-reference vector and element must be created to make the Concept viable and before the GINA Form for the Concept is configured using the third stored procedure. This is done by logging into GINA through the web interface as a User with developer rights. As a developer in the GINA application, developer capabilities are viewed and accessed. This is an example of the zero-trust model-based aspect of GINA. Navigate to the Advanced Tools Module in GINA. In the GINA application web UI, web navigable Applications are in the left pane and Forms (or sub-Modules) on the Right. A step by step process is depicted in Image 4.12.

This will be the page that appears after logging in. Click on "Development Tools" Application.

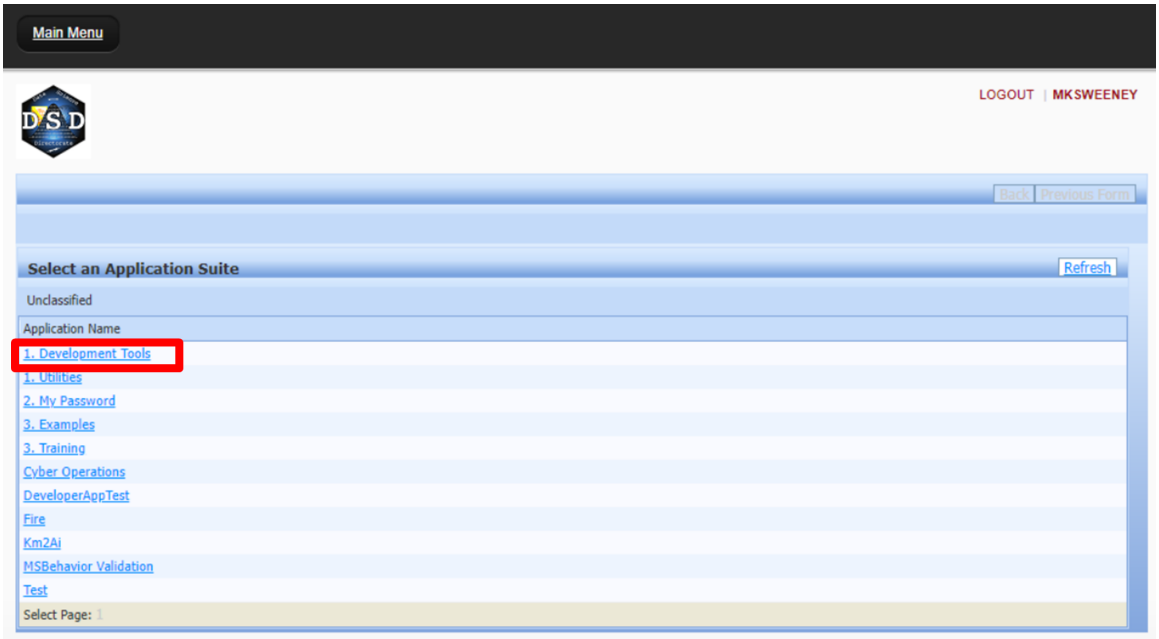


Figure 4.12. A developer's view of a list of GINA Applications. In this case Developmental Tools is selected.

Then click "Advanced Tools" Module from the next page to see the Module's Forms.

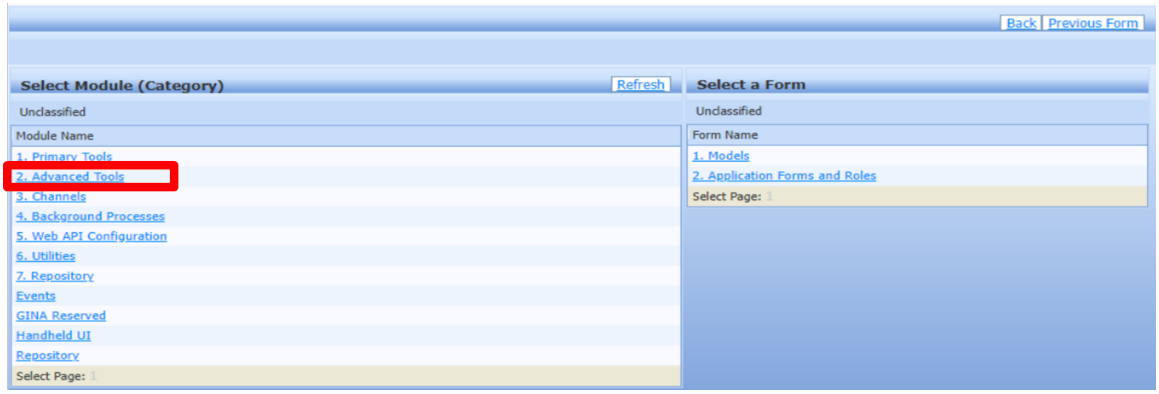


Figure 4.13. The default view shows Forms on the right that relate to the top Module on the navigation pane (left). The second Module in the list, Advanced Tools is to be selected.

Selection of the "Advanced Tools" Module yields the related Forms on the right hand side. The selection of the "Source Search" Form navigates to the GINA Form that expresses the data for the GINA Source, where a by-name search may be executed. The search will verify the correct completion of stored procedure one.

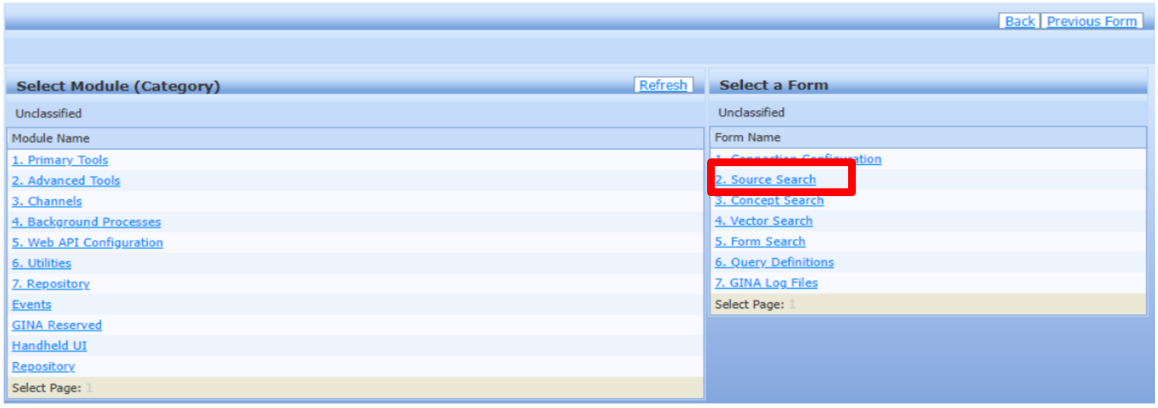


Figure 4.14. The GINA Form Source Search is selected from the Advanced Tools Module.

From there, search for the newly created table, “missionObjectiveState.”

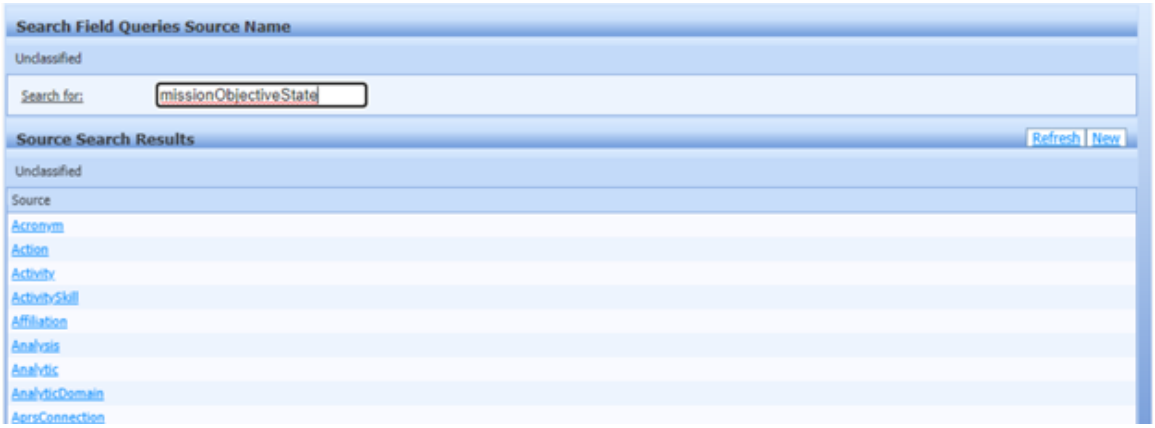


Figure 4.15. Source Search for new table to confirm proper execution and configuration of the GINA component.

This result listing the new Source, confirms the successful completion of stored procedure one. To instantiation of the XType Concept from step 2, and create the required self-reference elements, return to the Advanced Tools Module by using the “previous form” navigation button and navigate to the “Concept Search” Form.

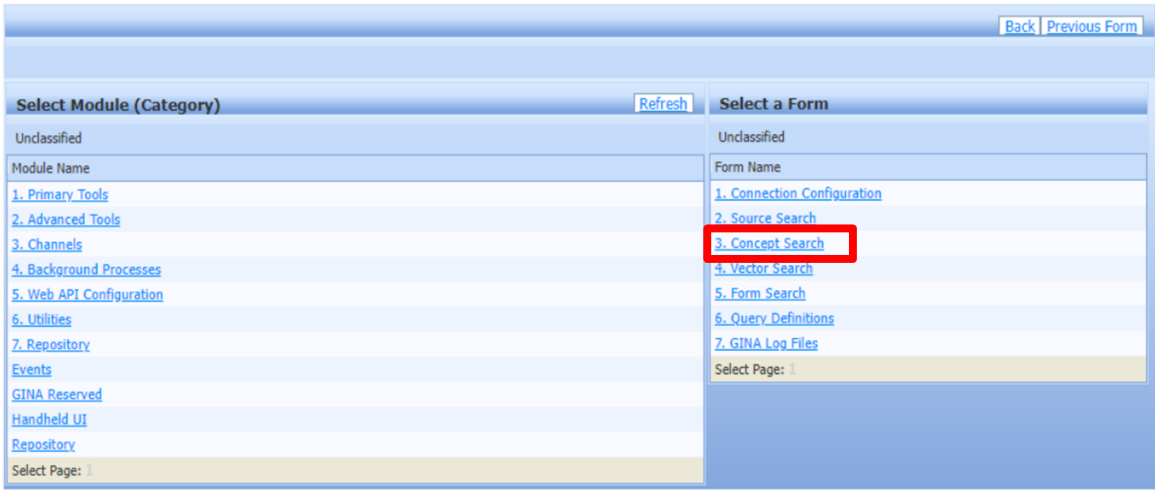


Figure 4.16. Selecting the Concept Search Form by clicking the hyperlink.

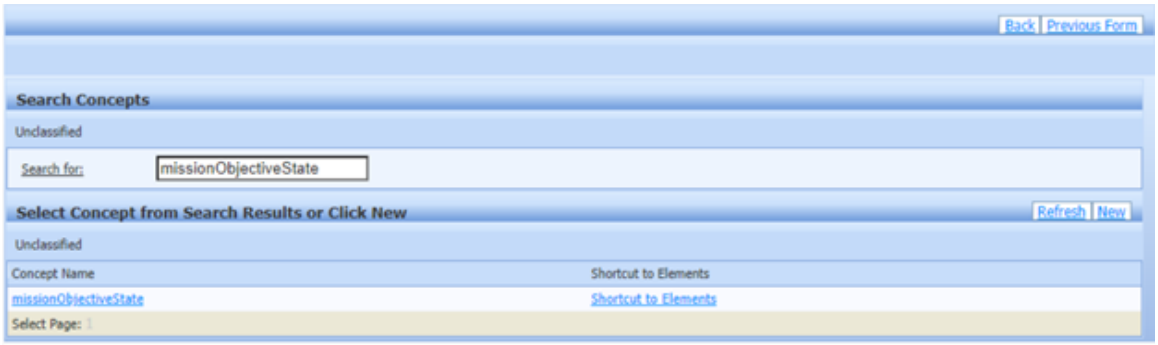


Figure 4.17. A search for the new Concept yields the item, and it is also web navigable for viewing or editing.

This confirms proper completion of stored procedure two. The new Concept must be configured with a self-reference element, that uses a self-vector to become a viable and accessible conceptual object to the federated information space in GINA.

After selecting the Concept "missionObjectiveState" hyperlink, the "Concept Details" page will appear. First, make the Vector. Select the "My Vectors" hyperlink.

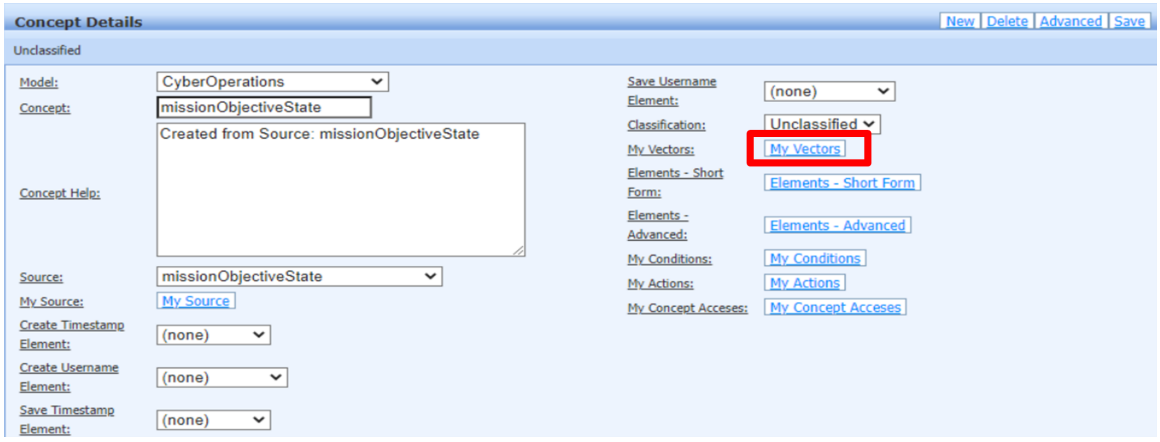


Figure 4.18. Selecting the "My Vectors" hyperlink.

The next page will give you the option to create new vectors by selecting “New,” as seen below in Figure 4.19.

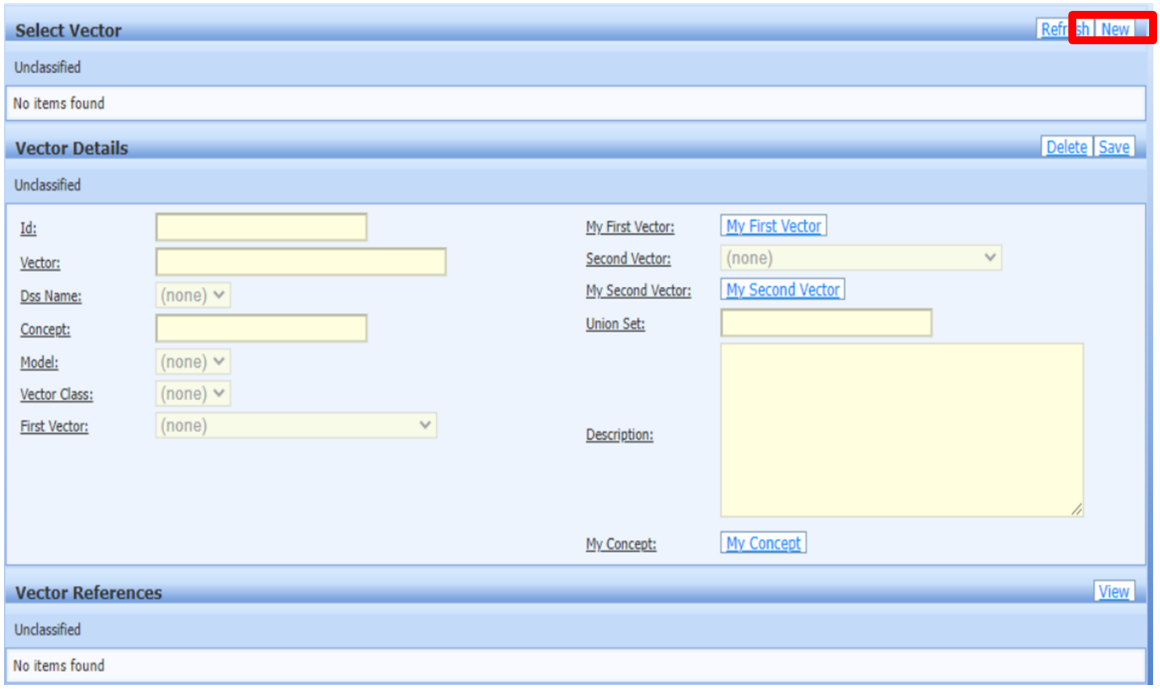


Figure 4.19. Create new vector. GINA maintains context, in navigating to Vector from a Concept, GINA will automatically propagate the context and thus reduce data entry.

From here, the self-reference Vector and the Vector Reference can be created. The self-reference Vector is created by selecting the "new" option. The "Id" field will be auto populated. The "Vector" name field should describe the relationship semantically, in other words, make sense to a specialist or operator. In this case, the relationship should describe "self" and thus be the same name as the Concept it belongs to, as it is the named relationship to itself. GINA maintains context, in navigating to Vector from a Concept, GINA will automatically propagate the context and thus reduce data entry. This is true for "Dss Name" and "Model." The "Vector Class" default "proximity" is correct and left alone. Once that information is input, select the "save option." Figure 4.20 presents the completed example for "missionObjectiveState" self-Vector.

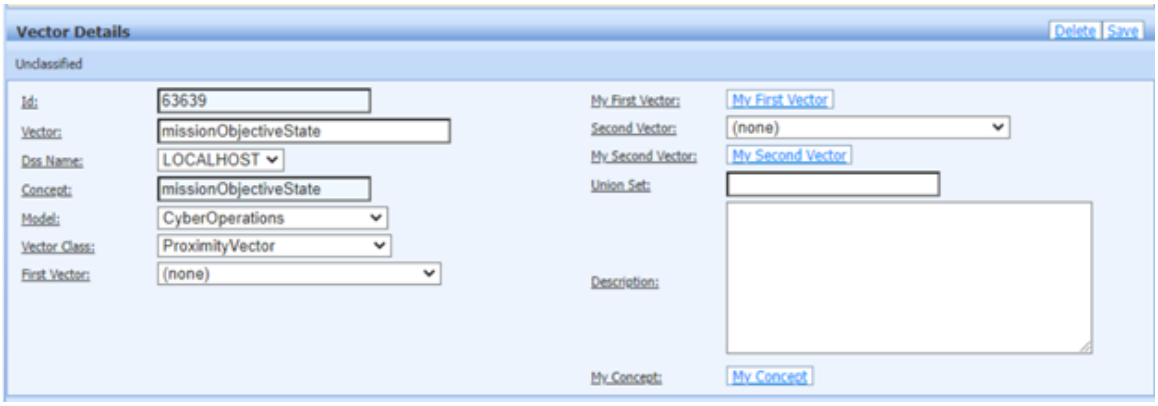


Figure 4.20. How to create a new Vector using the Vector Form.

The vector reference element is created by selecting the "view" option in the "vector references" section.

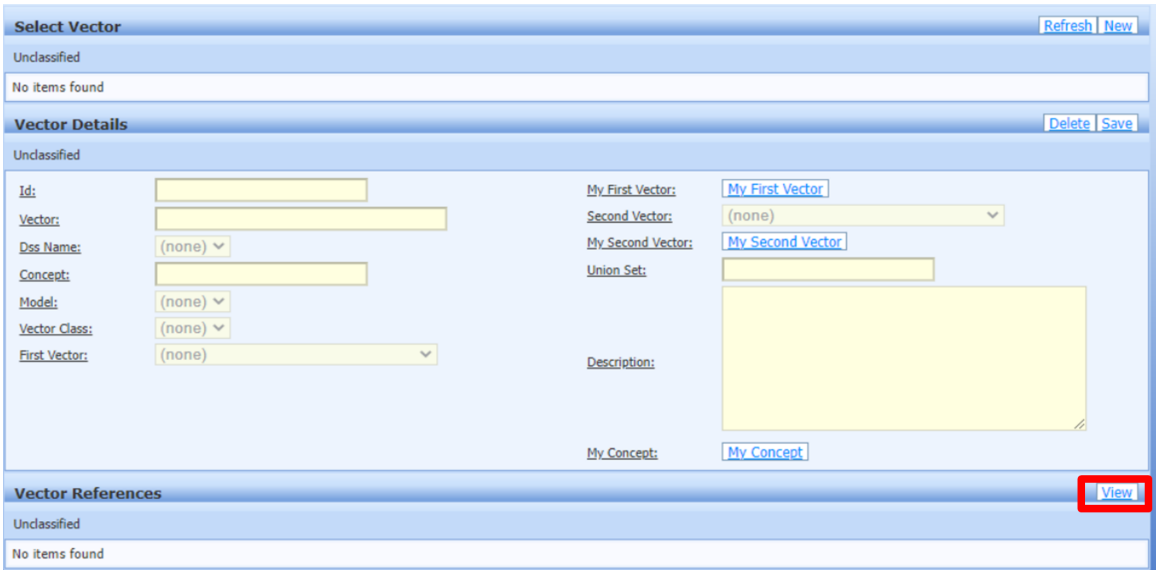


Figure 4.21. Vector Reference location

The resulting page will show the associated references, which will be empty to start. These

are the Elements or attributes of the Concept by which the Vector relationship depends. In the case of the self-Vector, the relationship to itself in its entirety is through the use of the GUID. In other words, when a Concept is accessed in GINA, its baseline response yields all rows as it is filtering on the unique GUID. Select the "new" option to add the reference element.

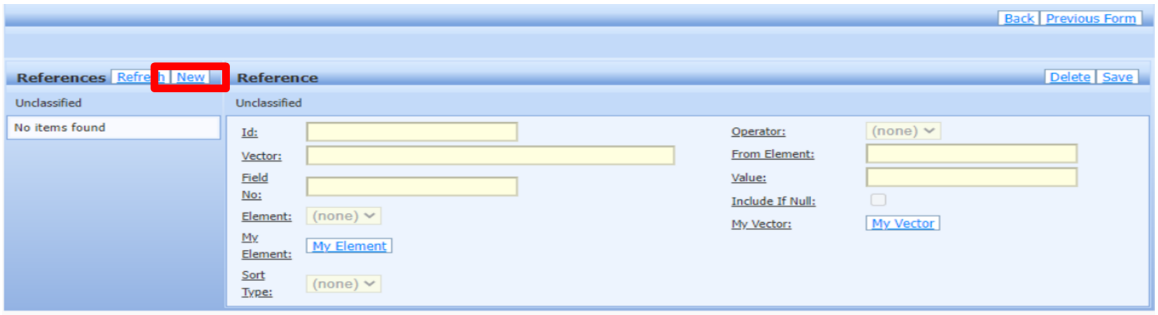


Figure 4.22. Create new vector reference by specifying the Element to filter on.

While it seems obvious, the possible Elements available to relate to a Concept are constrained to the Concept's Elements. These are specified using the drop down where "guid" is selected as the "Element" and also input for the "From Element" text field. The Vector name is auto populated due to GINA propagating context from the "Vector." The "Field Number" field is arbitrarily set to "10," and the "Value" field can be left blank as it is not required here.

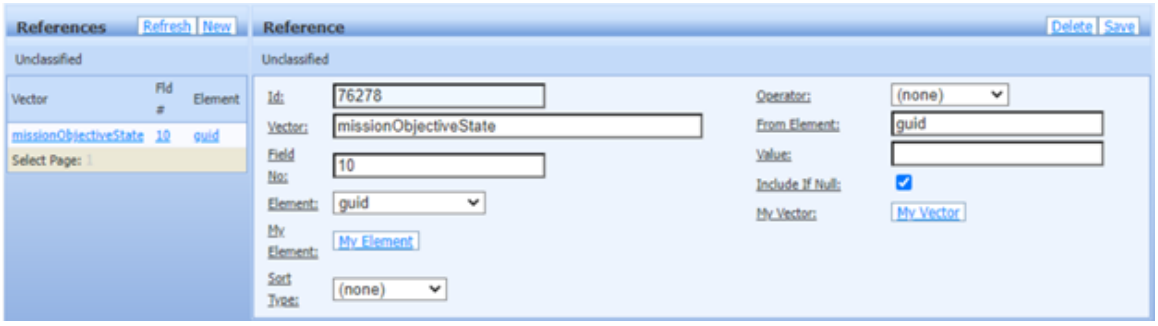


Figure 4.23. How to fill in new Vector Reference form. The References are the Element or attributes that specify how a Concept is related to.

This concludes the creation of the self-reference Vector and the Vector Reference. The next step is to create an Element for the Concept that is the first extension beyond the 'core' data. The Element is the first addition of a knowledge Element component to the Concept, and that is of "self." Thus, it is obvious that the convention is to name of the Element the same as the Concept, and self-Vector. This Element is configured to use the self-Vector to related to or "query" itself, and thus provide is entire table or Concept when used a the Reference Element.

The next step is to create the self-reference element for the Concept. The process begins in the same manner as the authority and resource fields, by accessing the shortcut menu from the form page, this time selecting the "concept" link.

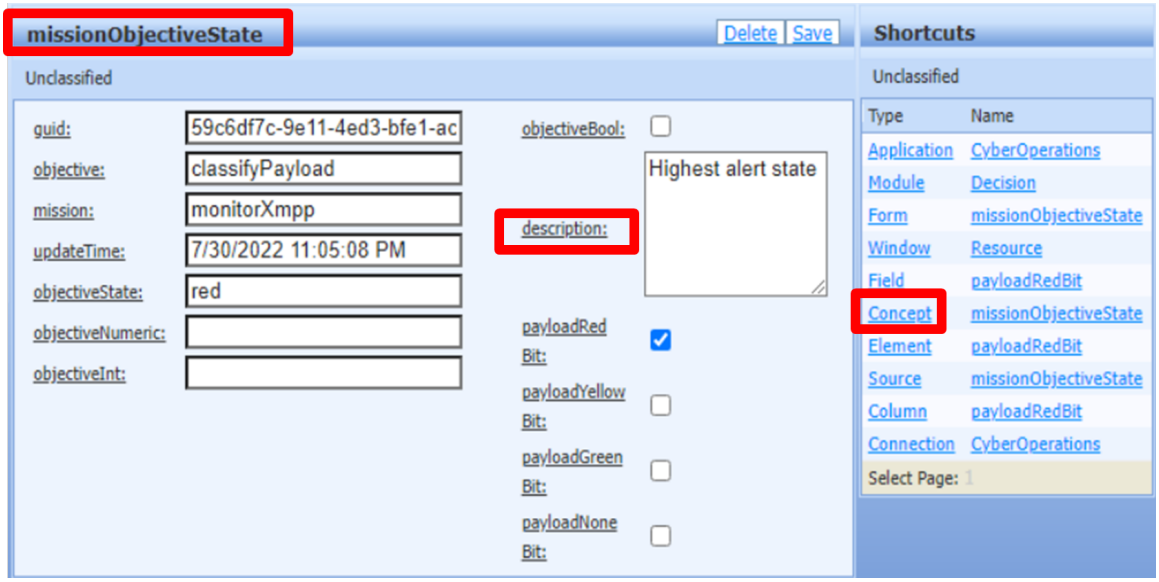


Figure 4.24. Access the content page from the shortcuts menu

From the Concept page 4.25, select “Elements-Advanced” which provides access to the element manipulation interface.

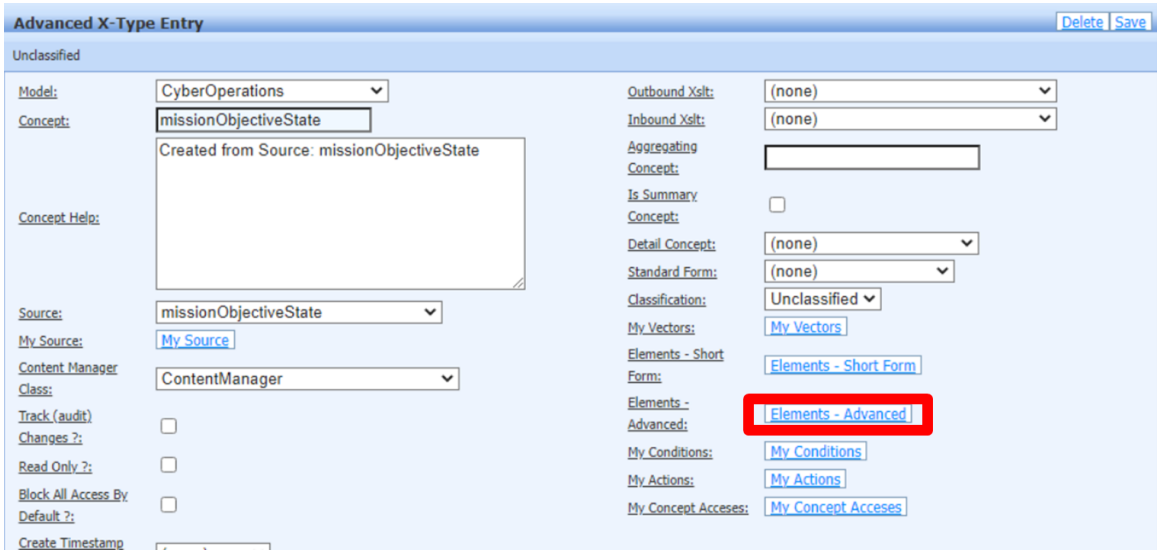


Figure 4.25. Access the advanced element maintenance page from the XType Concept page

To create the self-reference Element, select "new." Then fill in the appropriate information, as depicted in Figure 4.26 for "missionObjectiveState." The name should be the same as the Concept, as it is the self-reference. The field number should be "99999," the standard for self-reference elements. The data type is "content." The associated column and "default from element" are both "none." The "Content Concept" and "Reference Vector" should also have the same name as the Concept.

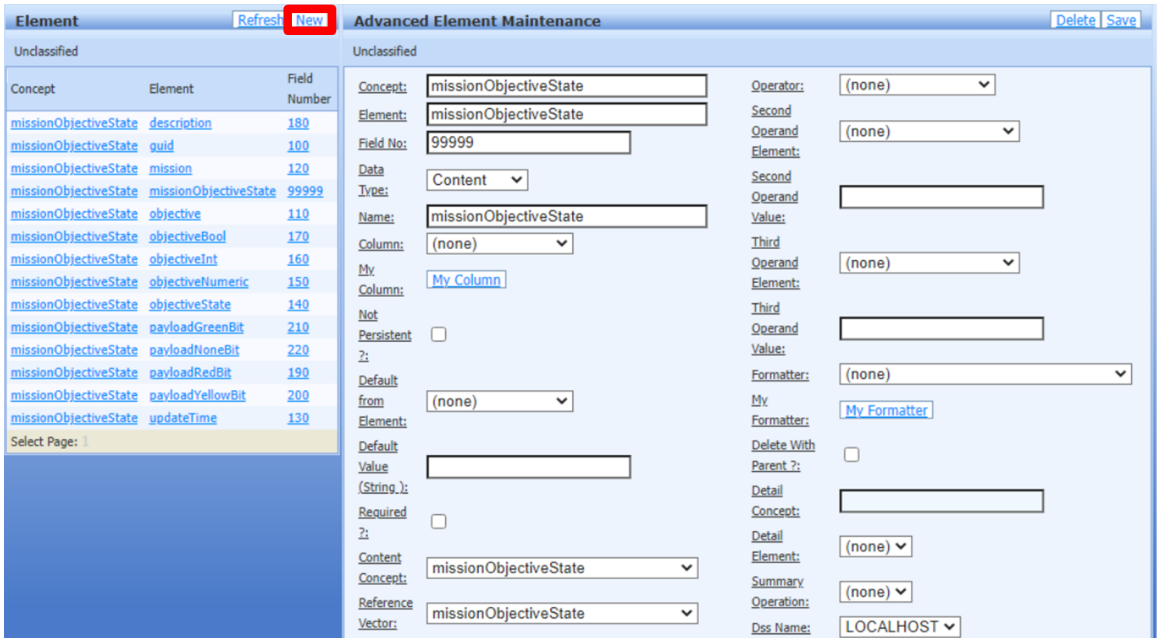


Figure 4.26. Self-Reference vector creation example

At this point, the XType Concept is viable in the GINA framework and the last stored procedure can be executed in the SQL database.

The purpose of stored procedure three is to configure the GINA UI Components: the Form, the Form's Windows, and the Window's Fields. It is called "CreateFormForXType_STEP3" and is located in the same DSS programmability section, just as stored procedures one and two are.

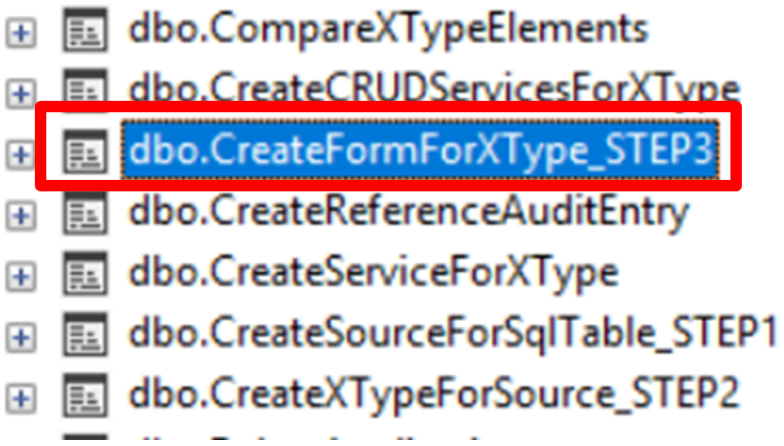


Figure 4.27. Stored Procedure Three, "CreateFormForXType_STEP3" to auto configure the GINA UI components Form, Window and Field.

Selecting the "execute stored procedure" option produces a table to capture required information to configure the Form, Windows, and Fields. In this straight forward case, the "XType" and "Form" has the same name as the database table created. It is noted that the GINA Source and Column components allow for mapping any named database table or column into the XType Element ontological specification, and likewise, Form names have no constraints. In simple cases, the Concept's name has inherent and clear meaning and is used for all components. The "Application" is the "CyberOperations" application that contains the organizational Models and Module components in GINA. The "Module" component in GINA is an organizational component for Forms, and thus an Application may have Modules for function, developers, users, admins, supervisors, etc. In this case, the Forms are organized under the "Decision." Module.

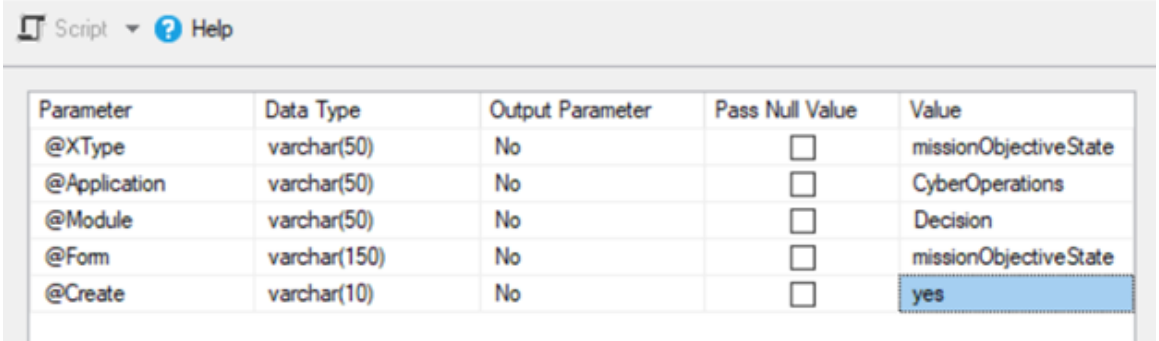


Figure 4.28. Stored Procedure Three finalizing instructions

Successful execution of the stored procedure produces the SQL script seen in Image 4.29 and GINA now has a configured Form for the Concept that is available for navigation as seen in Image 4.30. The script will configure all of the Concept's Elements to be expressed as Fields on the Resource (metadata) Window of the Form, however, it leaves configuration of the Authority (navigation) Window entirely up to the developer, as the navigable Element is dependent on the Concept and how a domain user chooses the abstract to navigate on.

The GINA Image 4.30 is showing the list of created Forms within the Decision model. At any time, this Form can be accessed by starting at the GINA home page and selecting Application "Cyber Operations," followed by Module "Decision." This emphasizes that the entire GINA development environment as well as the configured solution is fully transparent as a web navigable information graph technology, and the GINA framework affords a hierarchical organizational structure that is easily understood and navigated through as an Application that has Models with Concepts, and Modules with Forms, where Forms are the expressions of Concepts.

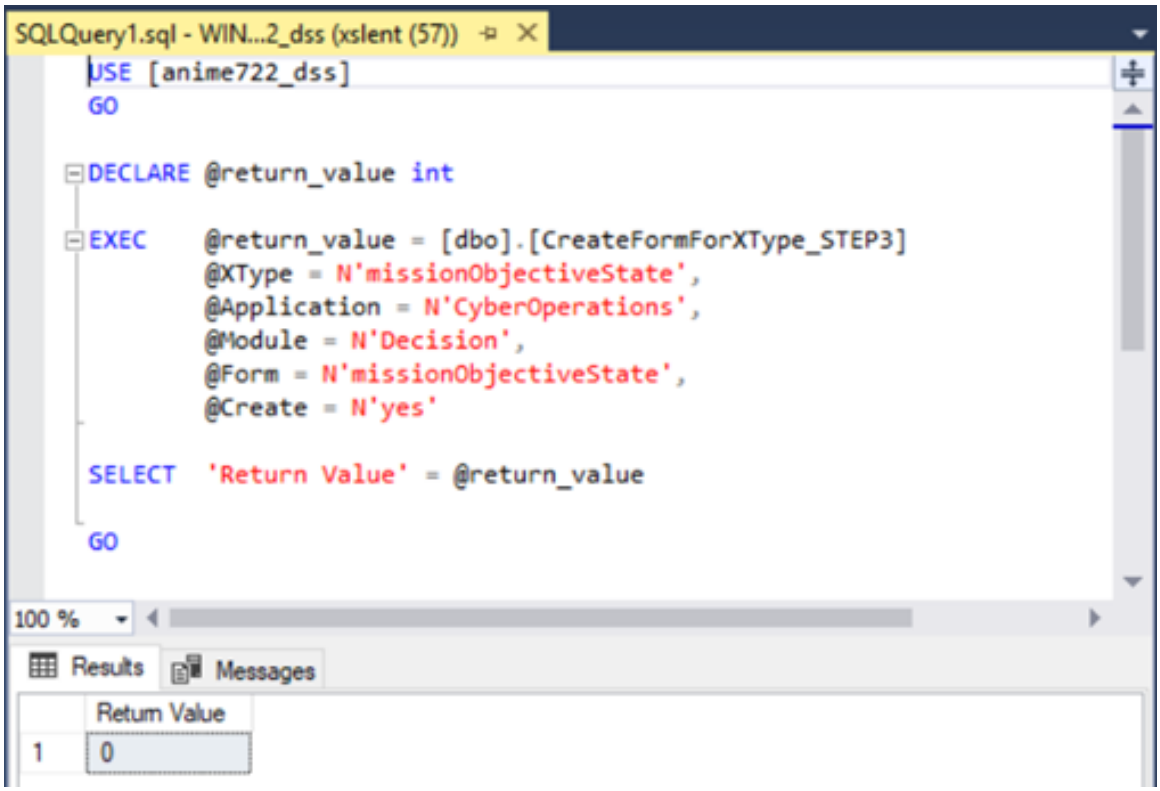


Figure 4.29. Stored Procedure Three, CreateFormForXType_STEP3, successful notification in SQL.

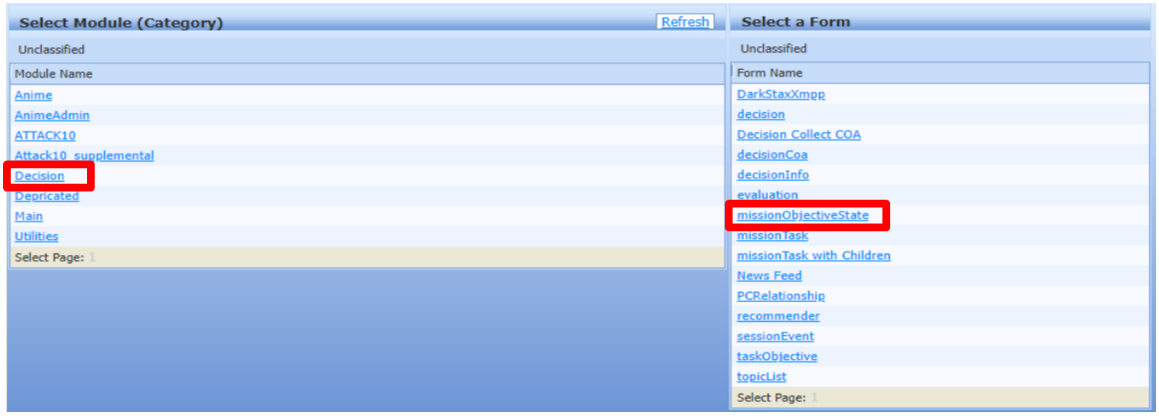


Figure 4.30. The new configured Form is now viewable and available in the GINA Module "Decision" as seen above.

The next step is to create the Authority navigation Field. A convenient capability of the GINA framework is that an Application's configured components are inherently linked to the underlying GINA model components, and a developer has the ability to navigate to any of the primary GINA model components from the Form they viewing by clicking on the field names which, for developers, are hyperlinks with the context of the XType that is being expressed. This may be thought of as a zero-trust coupled model that is user and role dependent. From the "missionObjectiveState" Form, select any of the Window Fields to bring up the "Shortcuts" menu . Within the "Shortcuts" menu, select "Form."

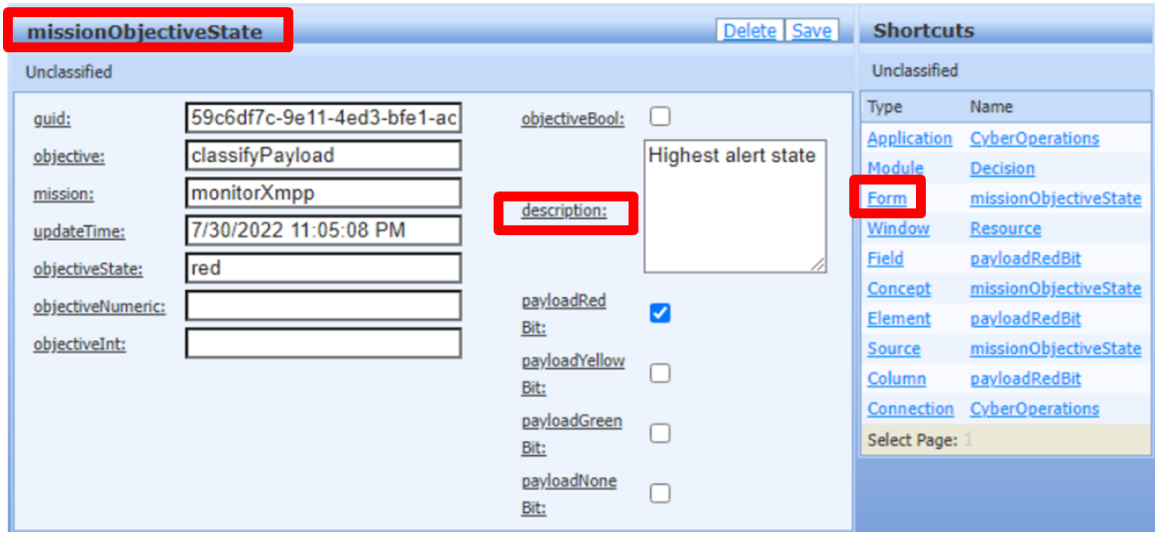


Figure 4.31. Developer access to the Shortcuts menu affords the option to navigate to primary GINA model components with the context of the expressed XType. Here the Form option is chosen to gain access to this Forms Windows.

The Form window for "Form" displays the Form name and the underlying XType and allows for a description. At the bottom of the window is a collection view of the Form's Windows as a hyperlinked list. The stored procedure has created both the Authority (navigation) and the Resource (metadata) Windows, select the Authority Window to specify the navigation Elements to be used.

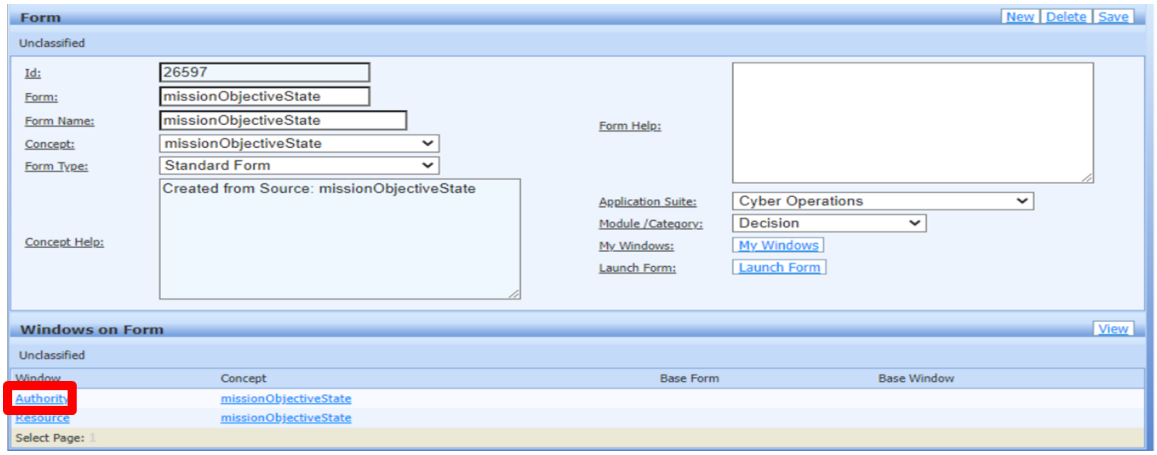


Figure 4.32. The Form view of the "missionObjectiveState" Concept and the Authority and Resource Windows that were instantiated. The blue text indicates hyperlink navigation.

On the Authority Window page, select the "view" button at the bottom right side of the screen to navigate to the Fields Detail page where the Elements are specified as navigation Fields.



Figure 4.33. View the Fields on the Authority Window to add a specified Element as navigation Field.

From the Fields Details page, create the new Field by selecting "new" and filling in the information as depicted in Image 4.34. The "Form" window, again contextually auto-populated, should designate the parent Form, which is "missionObjectiveState" in this

example. The “Field Number” informs the order of presentation of Elements on the Window, from top to bottom, and left to right in a two column configuration. The smallest number on the top-left and largest on bottom-right. As there are only one-three metadata Elements at most used in navigation for any Concept, the Field No. has left-to-rightness meaning to the display of information on the Authority Window. The “Element” drop down allows choice of any of the Concepts Elements, and it is up to the developer to choose a meaningful abstract. The “Name” field designates the human readable title of the column. The “Field Type” drop down menu allows designation of the functionality that the column will have, in this case it is “Select and Navigate.” Upon saving, the Form is navigable, the Authority Window now displays specified information and allows navigation that expresses the entire (or specified) metadata on the Resource Window for the Concept.

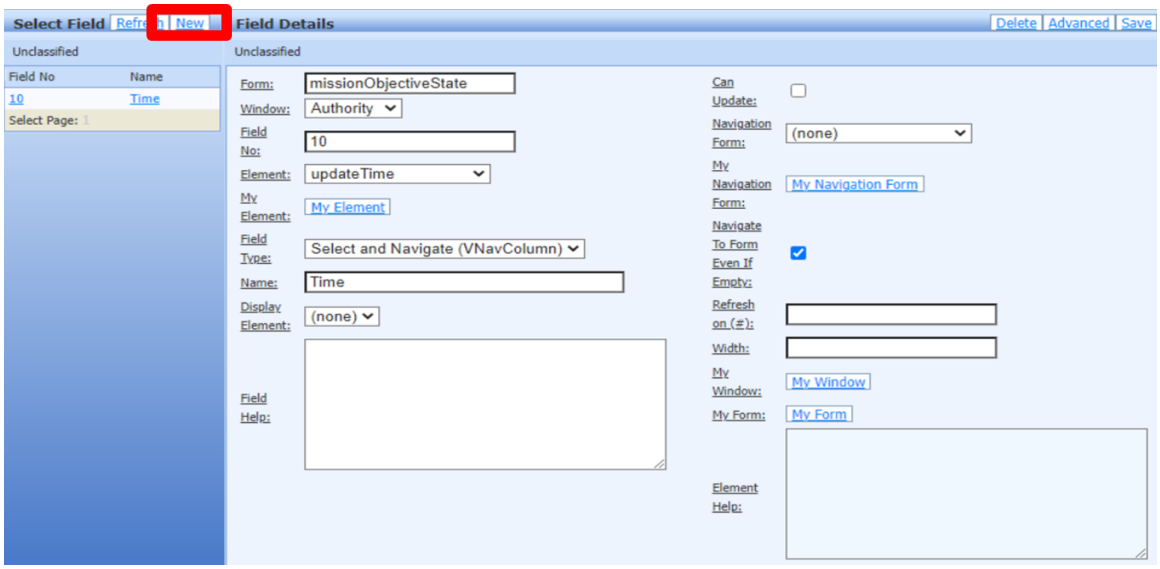


Figure 4.34. Authority Window's navigation Field creation example

Now that the Authority Window is created and configured with a navigation Field, the next step is to verify Resource Window Fields and create and if necessary. In most cases, the Resource Window Fields will be created properly during execution of the CreateForm-ForXType_STEP3 stored procedure and may only need re-ordered according to developers preference. To verify that the Fields were created, begin by accessing the Forms developer

Shortcut window by the same process described for the Authority Window Field configuration. Note, as a developer any of the Field names are hyperlinked and will have the context driven by the Form's underlying Concept. Thus any hyperlink name on a given Form will yield a Shortcut list containing a Form hyperlink with the same (correct) Concept context. Select "Form," then select "Resource" Window hyperlink from the Form's collected windows.

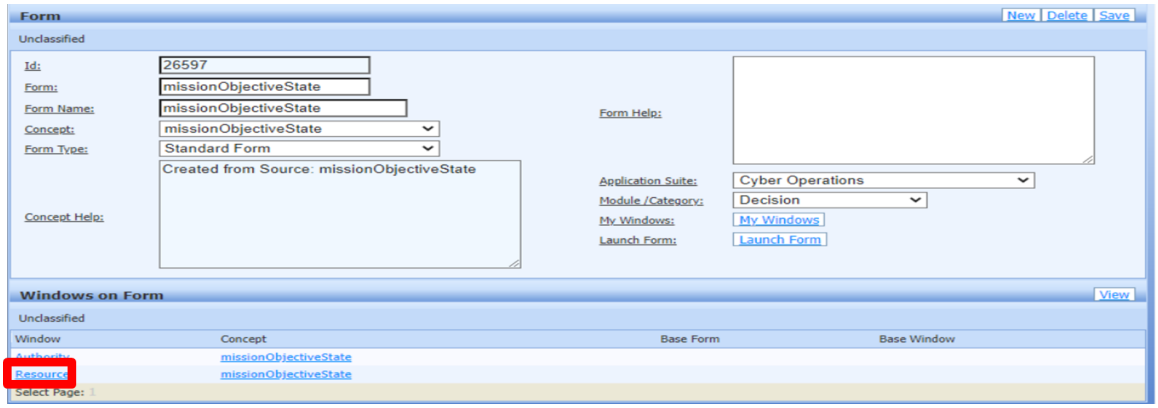


Figure 4.35. Accessing the Resource Window from the Form view with Windows enumerated as hyperlinks at the bottom.

On the Resource Window page, verify that there is a corresponding Field for all Elements that exist in the Concept enumerated at the bottom of the page. In this case, all Elements associated with the "missionObjectiveState" Concept do exist as Resource Window Fields, as shown in Image 4.36.

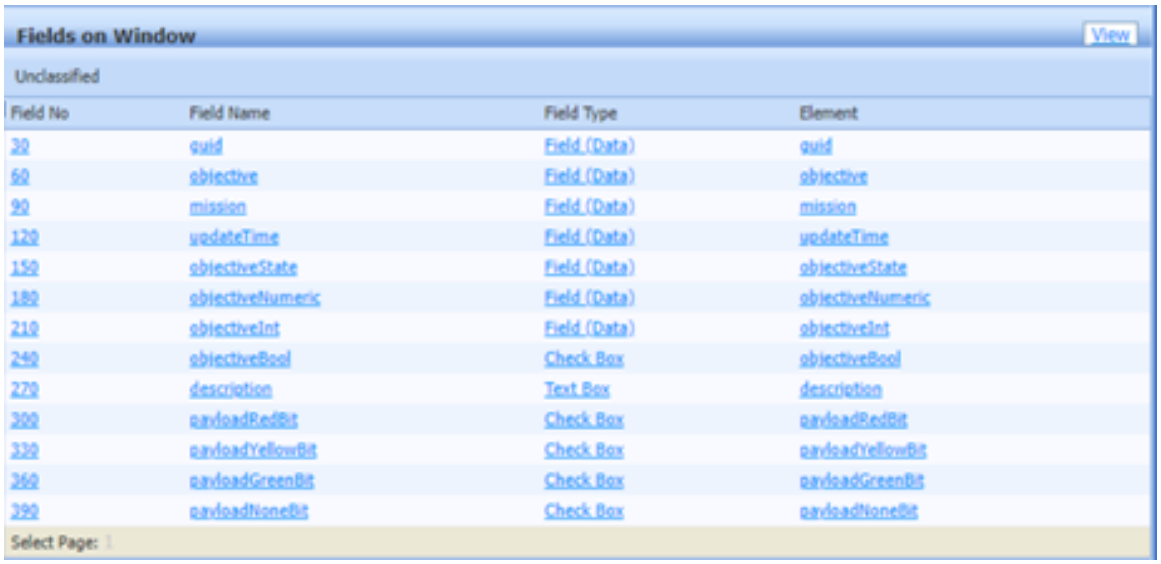


Figure 4.36. Verify Resource Window Fields are enumerated at the bottom of the Resource Window page.

4.2 Decision Implementation

The GINA components required to ingress data from the GINA service are comprised in a Channel configuration. This allows named Fields to map to named Channels that map to named Concepts, thus allowing Channel Fields to be instantiated that map to or from any standard JSON or XML file. The decision and output portion of the model only require configuration of relationships between the Data, Axiom and Status Concepts in the model. If the output is to be forwarded to another application, an output channel can be created to required specifications. However, the end state of this test is to determine that the model accurately classifies the inputs and displays the information internally and acts, i.e., moves information (data, information and knowledge) to a Status update.

The three Concepts involved with the decision portion of the model are the Data, "DarkStaxXmpp," Axiom, "MissionObjectiveState," and Status, "MissionObjectiveStatus." This section discusses each Concept and documents configuration specification required for the decision model.

4.2.1 Dark Stax Xmpp Concept

For "DarkStaxXmpp," the first thing of note is the ability to create the decision parameters in the elements directly, i.e., data-to-information. The "DarkStaxXmpp" XType Concept may have Elements added and thus extend the Concept to include new attributes that yield information. This is due to GINA Element attributes that permit a "new" Element to be informed by a "data" Element and apply logic, operators or transforms yielding a "new information" Element. In this model, four "new information" Elements are created within the "DarkStaxXmpp" Concept, one for each alert status. Each of them using the incoming data Element of "payload" and executing a logical operation to instantiate their named information state. For this scenario, the new elements will be "payloadRedBit," "payloadYellowBit," "payloadGreenBit," and "payloadNoneBit."

For this model, the additional "information" Elements are true/false based on a logical operator of comparison, thus, the "Data Type" for these must be specified as "binary." The parameters used for the comparison must be specified on the Element. These new Elements are configured with a "Reference Element" that is the data Element which is to be evaluated, in this case "payload." The "operator" then needs to be set to "matches," indicating that a determination of true or false will be made if the payload matches certain parameters. The "operand value" is then configured with the string criteria that would trigger this element to be true. In this scenario, the "Operand Value" would be configured with "traffic is high/medium/low/none" for the four respective elements.

Once all four elements are created with their respective truth value criteria, Fields must be created for each Element on the "DarkStaxXmpp" Form's Resource Window for the Concept. For each new Field, the Element must be specified by the drop-down menu, and the desired human-readable title must be annotated in the "name" field. In this case, the Fields are named by the color they contain, so the Element "payloadRedBit" will be named "red." This configuration will display the four information Elements on the DarkStaxXmpp Form with true/false values.

To implement the new "information" Elements, navigate to the Elements configuration page by using the "Shortcut" menu available to developers via any of the Form's hyperlinked Field labels. Select the "Concept" hyperlink in Shortcuts. Then, from the Concept page select "Elements-Advanced." Once at the "Advanced Element Maintenance" page, select

"new" to create a new Element. Fill in the new Element with the necessary information. Of note, the "Data Type" must be "binary" to enable the true/false functionality. Then specify the "Reference Element," "operator," and "operand value" windows. The Reference Element indicates what data Element is to be analyzed. Mission operators and specialists advise what type of comparison is made, i.e., what information is important from the data. The operand value designates the logical operation and criteria used to analyze the input Element. An example for "payloadRedBit" is depicted in Image 4.37. In this case, the Payload data Element is being analyzed to determine if it has text that matches the criteria text of "traffic is high." This process is repeated for all four "information" Elements.

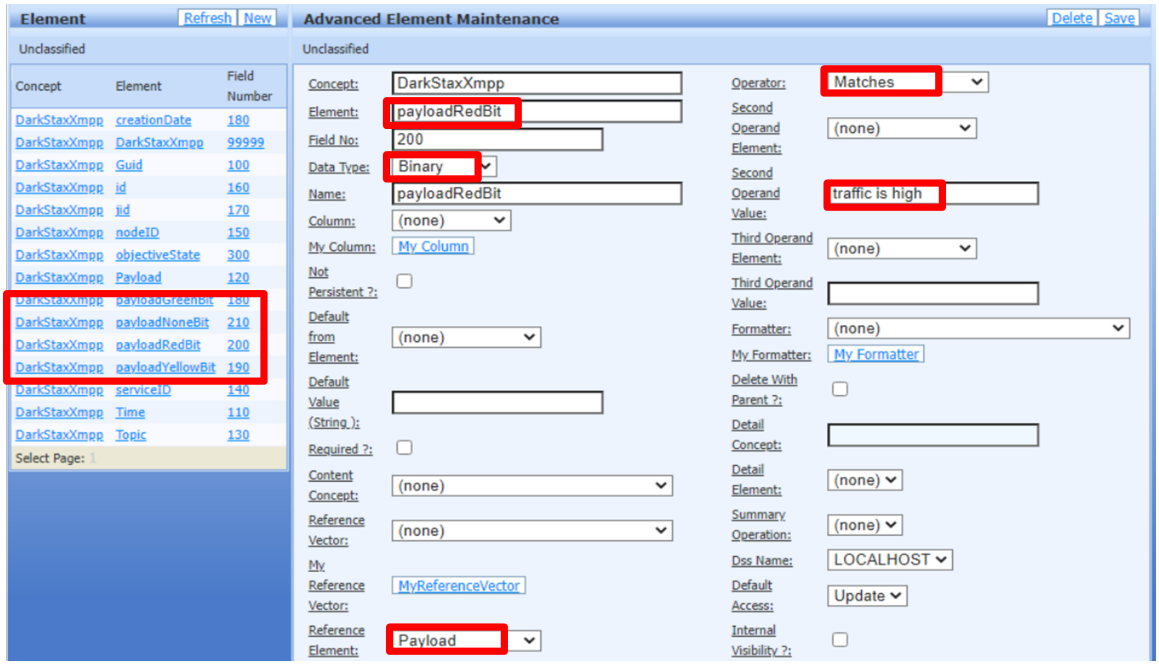


Figure 4.37. The image depicts the creation of information Elements of type bit based on analysis of the Payload Element in the DarkStaxXmpp Concept.

Then, use the "Shortcuts" menu again to select "Form," then select "Resource," then "View," and finally "New." Once at the "Field Details" page, fill in the new Field with the necessary information. Each new Element must have a corresponding Field for representation on the Window. The Field must specify the Element it expresses, the type of Field it is, and the

human readable title of the Field. The “red” Field, which corresponds to the “payloadRedBit” Element, is depicted in Figure 4.38. This process is repeated for all four Elements.

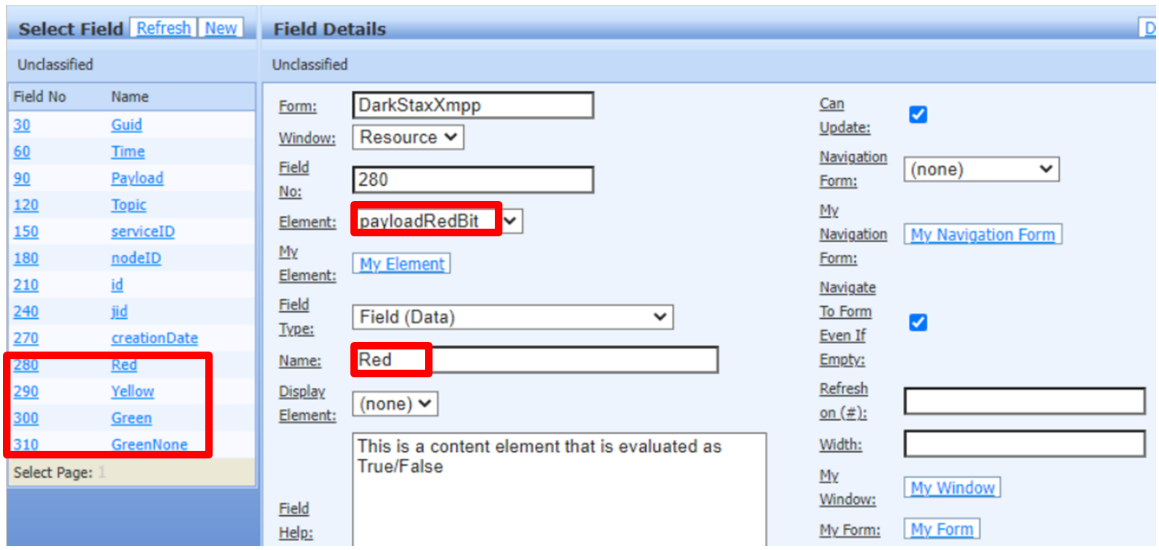


Figure 4.38. The image depicts the configuration of payload bit fields associated with the added information Elements in the DarkStaxXmpp Concept.

Upon implementation, the Form for the DarkStaxXmpp Concept will display the four Fields with true/false values for each added information Element. This can be seen in Image 4.39.

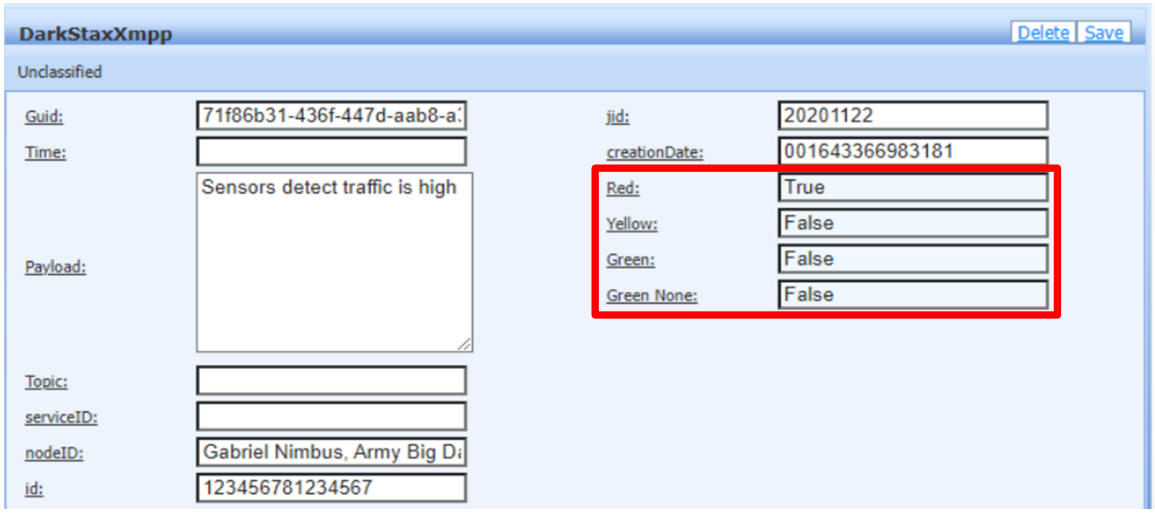


Figure 4.39. The image depicts how the Form for the DarkStaxXmpp Concept should look and function in response to input if configured correctly.

To conclude this subsection, the functionality and significance of the "DarkStaxXmpp" Concept is summarized using Image 4.40. The figure shows the relation to the design methodology for the decision model construct.

The Data Concept manifests the transition of data to information and then to knowledge through relationship(s) to truth tables. The section outlined in blue represents raw data Elements that entered the model as standard XML from a pubsub server. For this scenario, the Xmpp data is outline in the section on OpenFire messages.

The section outlined in yellow highlights the logical analysis and transformation to information. This is an instantaneous result that is constructed based on logical operators applied to the data. This is the extent of refinement that can be done with only one Concept, hence the necessity to relate or query an additional Concept or truth table.

The section in red represents the display of the data to knowledge transition. Knowledge can be gained by comparing the data and information to a state table. The knowledge gained from the truth table, in this case, is the objectiveState which is determined by the state of information and data of the Xmpp Concept. Now with data, information and the knowledge

of meaning of the data and information, further action may be executed. The possible actions and domains to act is unlimited and enabled by the knowledge.

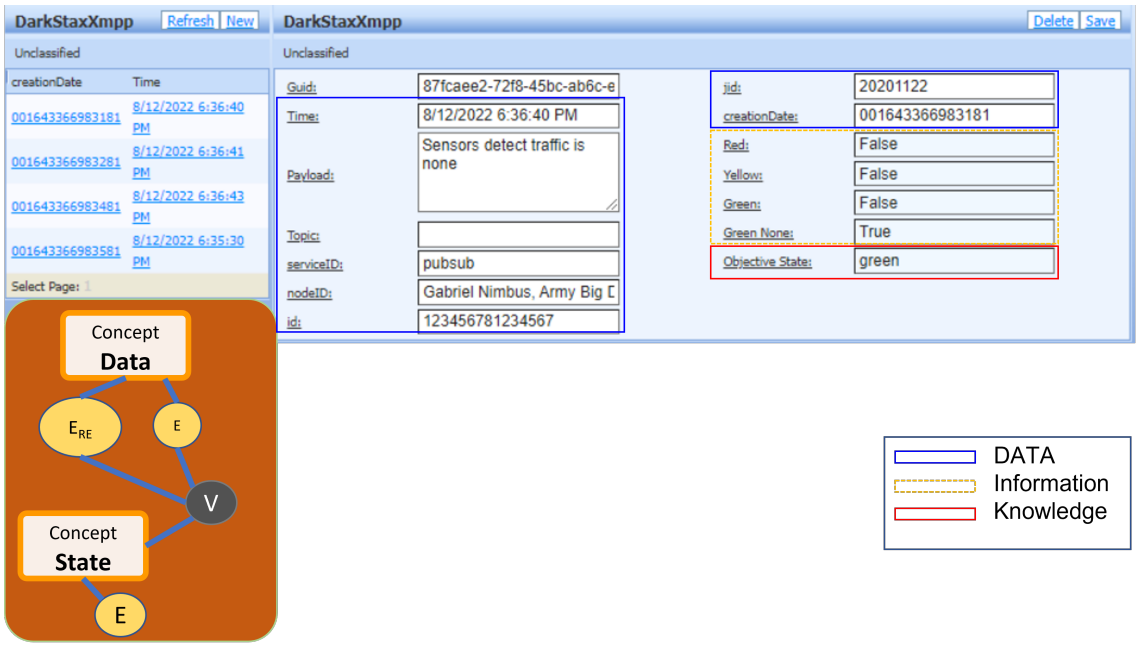


Figure 4.40. The DarkStaxXMPP Concept displays transition of data to information and knowledge. Raw data is displayed based on the input. Information is derived based on an initial set of logic applied to the raw data fields. Knowledge is obtained through relating the data and information a state table and deriving meaning.

4.2.2 Mission Objective State Concept

The Concept "missionObjectiveState" is the lookup Concept in the decision model. Its purpose is to act as a truth table that asserts the meaning of the state of information, i.e., the specific state of the payload bit Elements mean a certain alert status, here represented as a color. The Concept contains the four "information" elements from "DarkStaxXmpp" Concept whose states determines the meaning or status of the Payload. However, these elements do not determine themselves based on comparing the input to a set of conditions. Instead, these elements are used to construct the truth table by providing the proper objective

state that correlates to the truth or status value. For this Concept, the bit state of these elements may be input via check box. The meaning of the bit state is determined by the input values of the “DarkStaxXmpp” Elements. As a message is analyzed by “DarkStaxXmpp,” it evaluates to a bit state based on a set of criteria. A “knowledge” Element, “objective state,” is created that is informed through a Vector relationship to the truth table with the corresponding meaning or objective state. The “objective state” element signifies the level of the alert status. For example, if the “objectiveRedBit” is true, the “objective state” is given the proper corresponding value of “red.” This pattern repeats for “yellow,” “green,” and “none,” to create the truth table.

In the implementation, these elements are designated true/false data type, as opposed to the binary data type used for the objective bits in “DarkStaxXmpp,” and have the field type designation of “check box” in field details. Here, the payload bit Elements are tied to columns in the database, since they must populate known truth combinations and corresponding meaning to be used for future look ups.

After navigating to the “Advanced Element Maintenance” Form via the “Shortcuts” menu, the four payload bit elements are created with the specified “true/false” data type and the proper corresponding Column from the associated Concept Source.

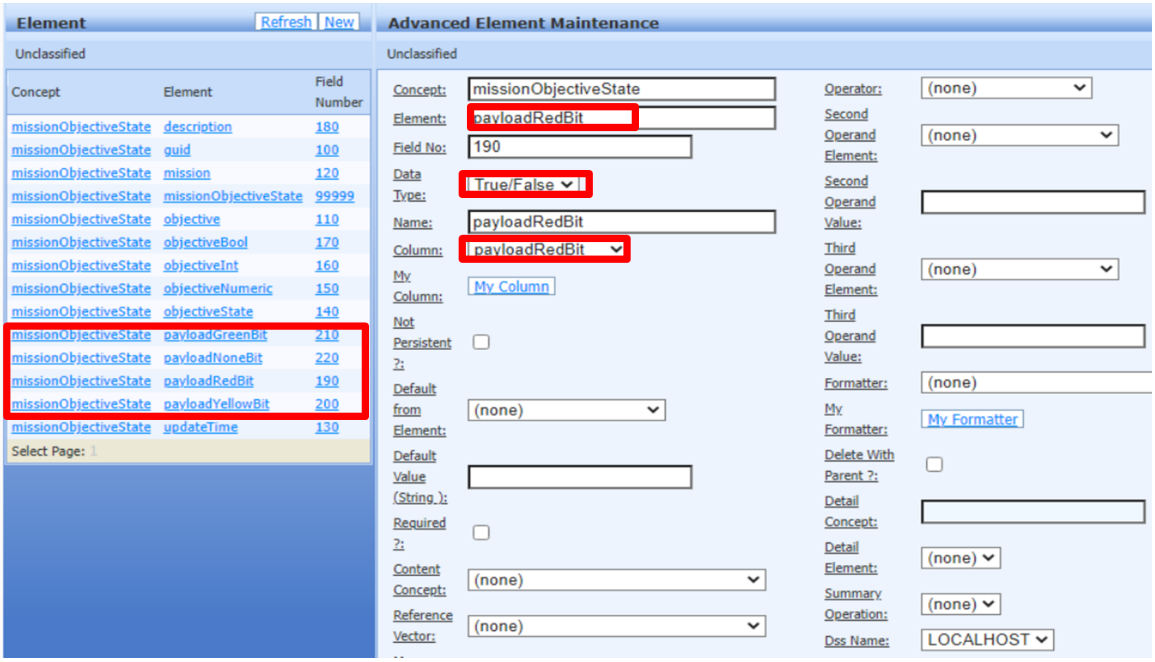


Figure 4.41. The image depicts the configuration of the new payload bit Elements in the MissionObjectiveState Concept.

Navigate to the Form page via the “Shortcuts” menu and then “Resource Window” to view and create Fields. These Fields are of type “Check Box.”

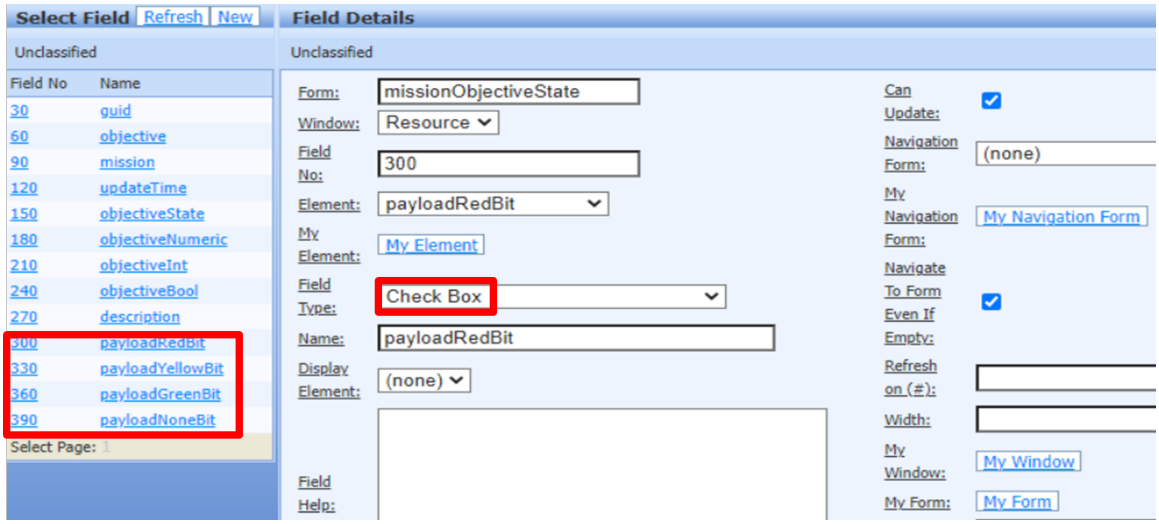


Figure 4.42. The image depicts the configuration of the new payload bit Fields associated with the Elements in the missionObjectiveState Concept.

Once all are implemented, the truth table can be created directly in the Concept Form of "missionObjectiveState." To do so, create four new instances, one for each correlating objective state and payload bit. For example, as seen in Image 4.43, the "payloadRedBit" is indicated to be true, so the objective state is indicated to be "red." While this example uses a simple relation, the correlating objective state could be that of an alert status or an appropriate subtask based on the truth value. A related instance is to be created for all four objective states. This completes the truth table and lookup Concept.

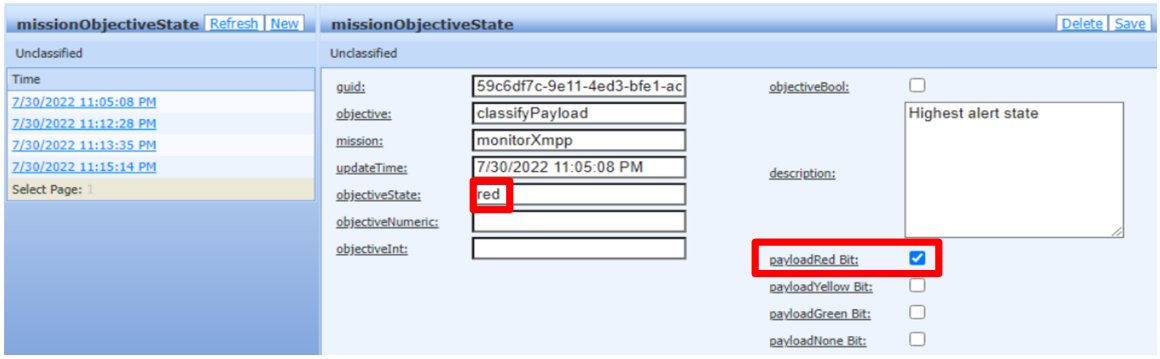


Figure 4.43. The image depicts how the MissionObjectiveState Concept page should look and function in response to input if configured correctly.

The functionality and significance of this Axiom Concept is expressed and associated with the design methodology in Image 4.44. The purpose of the mission objective state Concept is to establish the truth table for the specific mission. The mission and objective Concepts provide a knowledge management framework, so the truth tables may capture use cases beyond that of monitoring XMPP battle field chat. This allows the model to accommodate any objective. Of additional note, the "objectiveNumeric," "objectiveBool," and "objectiveInt" Elements were added to the model for cases where information derived from data involves other data types than bits, but are not used for this use case. They are pre-positioned to accommodate broader objectives.

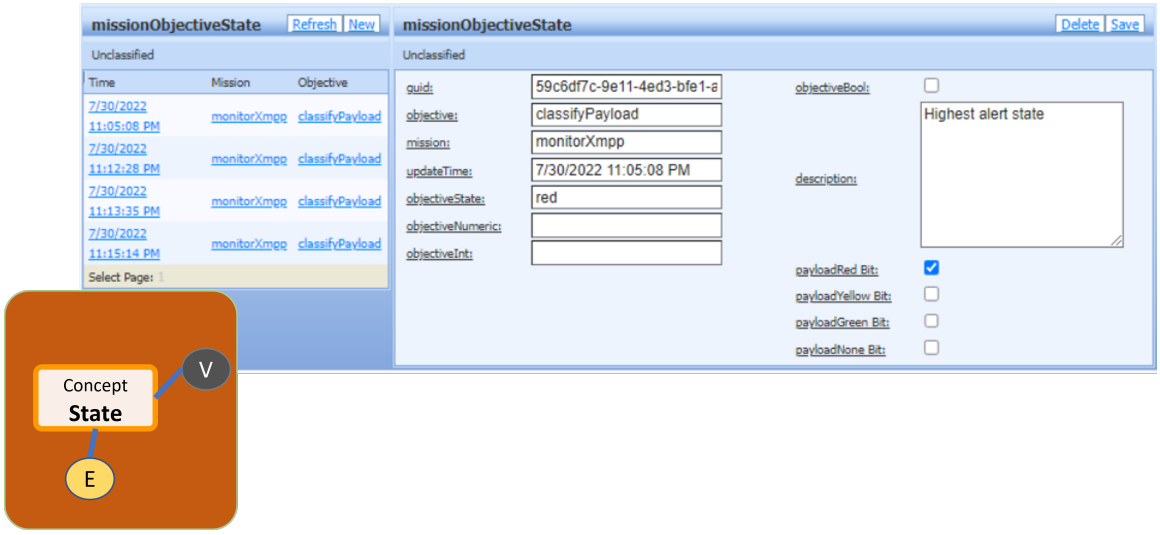


Figure 4.44. The objectiveState Vector relates to the truth table based on the four information bit Elements in order to return knowledge and meaning to the Data Concept.

4.2.3 Mission Objective Status

The "missionObjectiveStatus" Concept is created to record data input classifications and decisions. The Concept keeps the history for data stewardship and further analysis if needed. It may instead or in addition to, execute command line operations, as it has all the data, information and meaning that is also enhanced with transforms to accommodate execution of any process or save. While beyond the scope of this thesis, the meaning in the truth tables may entail causal inference, opening up limitless exploration of system cause and effect.

The Status Concept is connected to the "DarkStaxXmpp" Concept through linked actions. The specifics of configuring the linked actions are discussed in section 4.2.5. The purpose of the linked action configuration is to sequence the actions of "DarkStaxXmpp" and "missionObjectiveStatus" in a parent-child manner, so that "missionObjectiveStatus" saves input message classifications subsequent to the Data save. Further, the Vector is configured in a specific way as to not overwrite old inputs with new ones, thus enabling the capture of all new Data states.

Other than the linked action configuration, which actually belongs to the Data Concept, the implementation of this Concept is straightforward, as it follows the documented step-by-step process closely. When configured, the Status Concept Form should appear as depicted in Image 4.45 after multiple messages have been analyzed by the model.

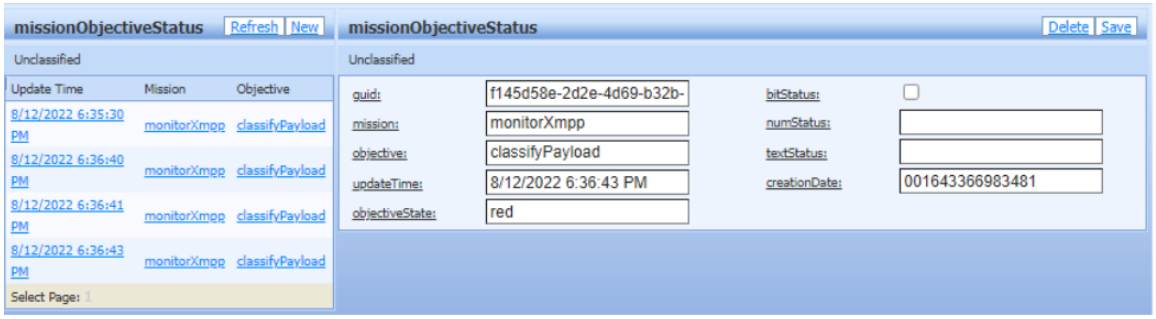


Figure 4.45. The Status Concept Form, is shown above, after data has been read into the application and processed. The model analyzes and saves the input to the XMPP Concept and executes the system behavior to save the meaning of the data or objective state in the Status Concept. The model is data informed.

4.2.4 Objective State Element and Vector

While the three Concepts of Data, Axiom and Status are important, the relationships (Vectors) between them are just as vital to a properly functioning model.

The first relational aspect necessary to discuss is the “objectivePayloadState” Vector, which must be created. This Vector is owned by the “missionObjectiveState” Concept and includes a Reference for each of the payload bits. To create this Vector, use the “Shortcuts” menu from the “missionObjectiveState” Concept page and select the “Concept” hyperlink. On the Concept Form select “My Vectors.” Create a new vector called “objectivePayloadState,” as depicted by Image 4.46.

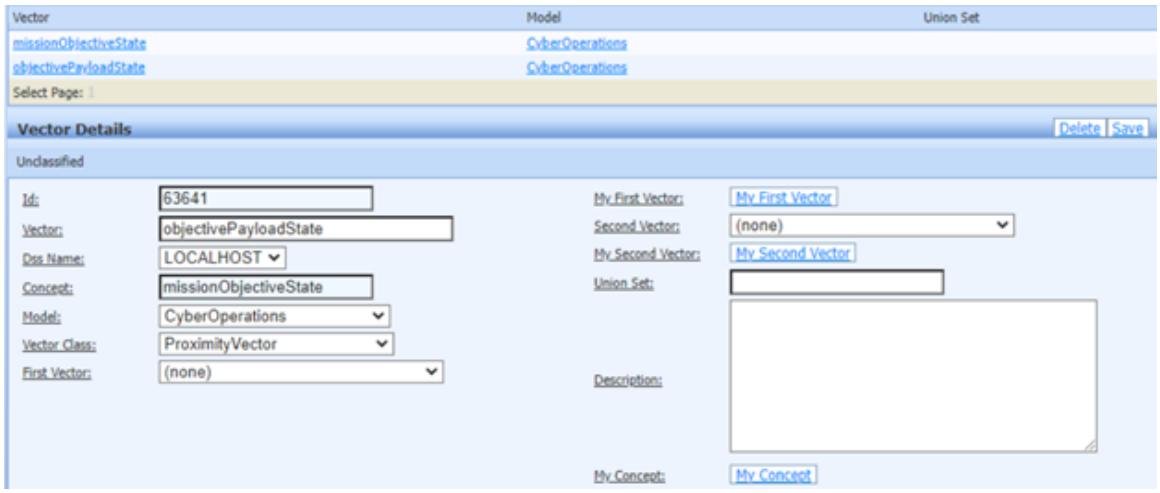


Figure 4.46. The image depicts the configuration of the "objectivePayload-State" Vector in the missionObjectiveState Concept.

From that Vector, create References for each payload bit Element. This can be done by selecting the "view" option next to "Vector References" and then selecting "new." The References are configured with the appropriate information bit Element and "from" Element. Implementation of the red bit Reference is seen in Image 4.47.

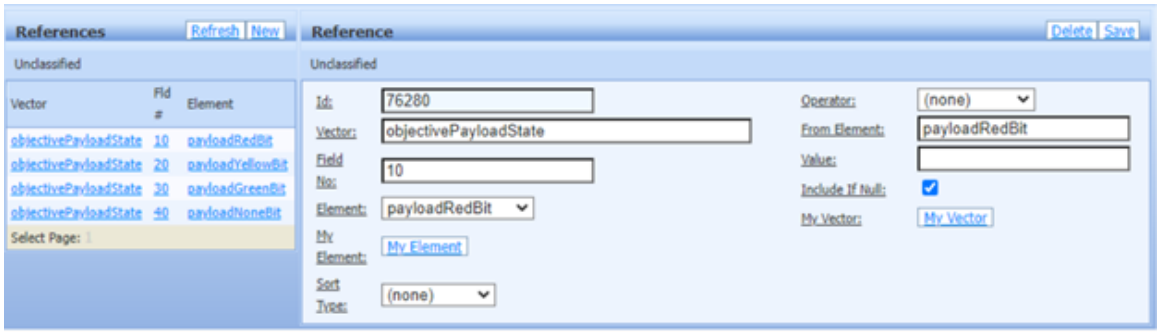


Figure 4.47. The image depicts the configuration of the "objectivePayload-State" Vector References.

The second relationship aspect to discuss is the “objectiveState” element. “ObjectiveState” exists in both “missionObjectiveState” and “DarkStaxXmpp” Concepts. It is a reference element that uses the “objectivePayloadState” Vector to select appropriate Objective State values from the truth table maintained in “missionObjectiveState” and match against the current instance of the four input XMPP information state bit Elements. This is how an instance of data input, the OpenFire message, is classified. The four information Elements are evaluate the data Element "Payload" content as true/false as to containing a phrase. These information Elements are then compared to the truth table in “missionObjectiveState,” where the corresponding Status is returned that matches the state of the four information bit Elements.

For example, a message containing the phrase “traffic is high” matches criteria to make the red information Element, “payloadRedBit,” in “DarkStaxXmpp” true. According to the truth table, a true “payloadRedBit” with the other bit Elements False, should result in an objective state of “red” based on the truth table created in missionObjectiveState. That objectiveState Element on “DarkStaxXmpp” Concept is informed via the “objectivePayloadState” Vector. The “DarkStaxXmpp” bit Elements use natural language processing to compare criteria to incoming message content and determine if bit statuses are true or false. The objectiveState Element is configured with the objectivePayloadState Vector to map the bit states to the truth table in the “missionObjectiveState” Concept allowing access to a corresponding status which informs the Element. Thus meaning and knowledge may be attained from truth table Concepts in any mission domain where the information state has meaning or implications.

To create the “DarkStaxXmpp” Element “objectiveState,” navigate to the “Advanced Element Maintenance” Form using the “Shortcuts” menu from the “DarkStaxXmpp” Concept Form. From there, create the new “objectiveState” Element, ensuring to select the newly created “objectivePayloadState” Vector as the Reference Vector.

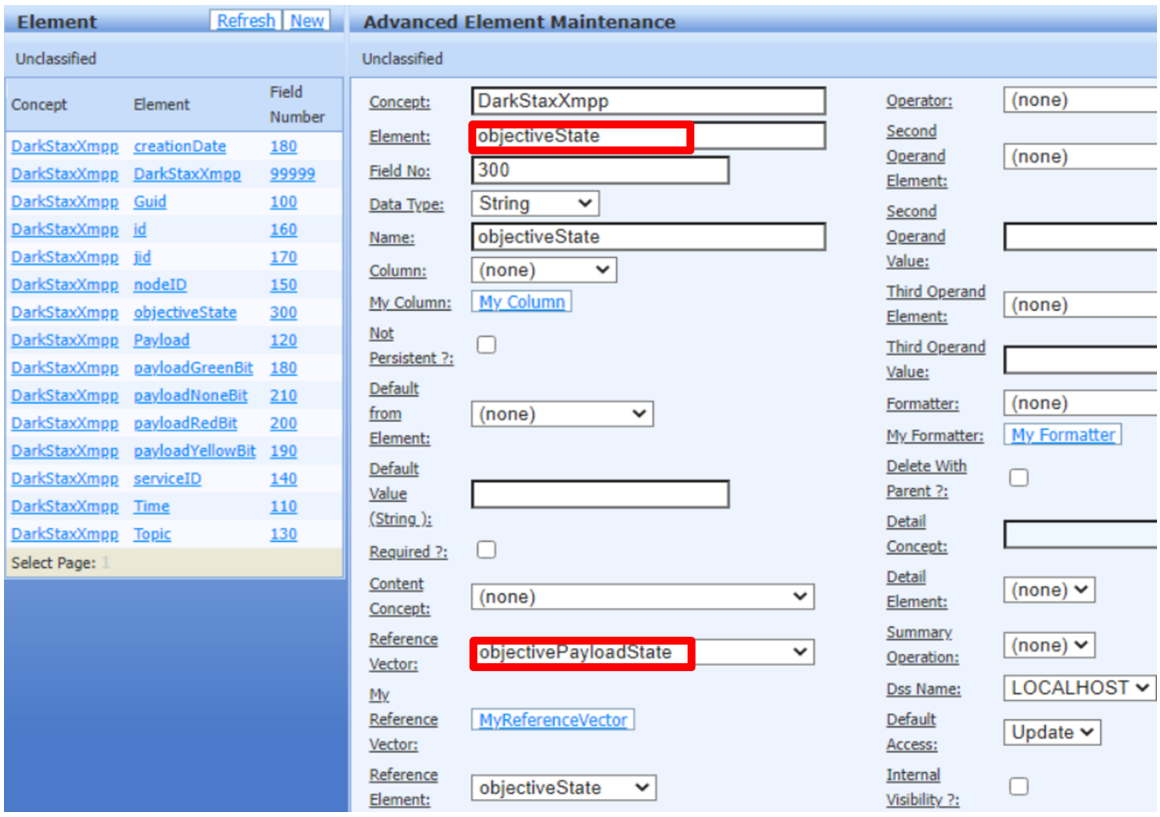


Figure 4.48. The image depicts the configuration of the objectiveState Element in the DarkStaxXmpp Concept.

The purpose of Figure 4.49 is to relate the configuration to the design. The right side of the image is the configuration for the “ObjectivepayloadState” Vector. The Vector, along with the associated objective state Reference Element, are two significant components that relate the Data Concept with the State Concept. Depicted in the left side of the image, the Reference Element is “Ere” figure and the Vector is “V” link.

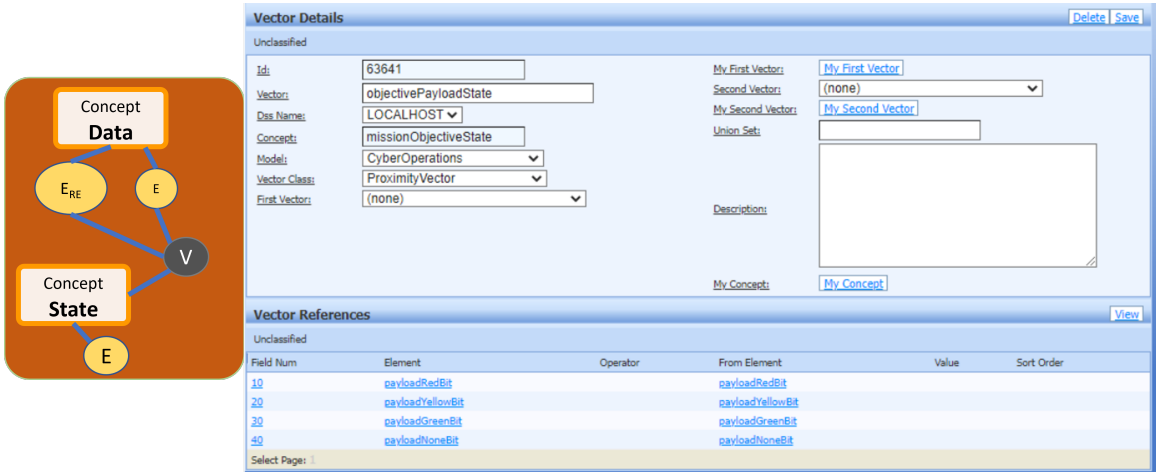


Figure 4.49. The "objectivePayloadState" Vector and Element link the Data Concept and State Concept.

4.2.5 Linked Action Details

As previously mentioned, another detail crucial for process and action is the linked action relation between "DarkStaxXmpp" and "missionObjectiveStatus." A linked action must be specified between the "DarkStaxXmpp" Concept and "missionObjectiveStatus" Concept in order to set in motion sequential processes in the underlying information system. The first step required for the implementation is the creation of another Vector. The Vector is the "mObjectiveStatusNone" Vector which acts as the required target Vector to the linked action and always has a value of true. To ensure the constant truth value, the element is configured with the condition that the "guid" should never be null. Since a new "guid" is created for all inputs, this condition holds. The purpose of ensuring the Vector always has a true value is to allow the "missionObjectiveStatus" Concept to make a new entry. Otherwise, it would overwrite the previous entries with the current ones and not maintain the history.

To create the Vector, access the "Shortcuts" menu from the "missionObjectiveStatus" concept page. From the "Shortcuts," use the "Concepts" hyperlink and then the "My Vectors" hyperlink. From the Vectors page, create a new Vector called "mObjectiveStatusNone."

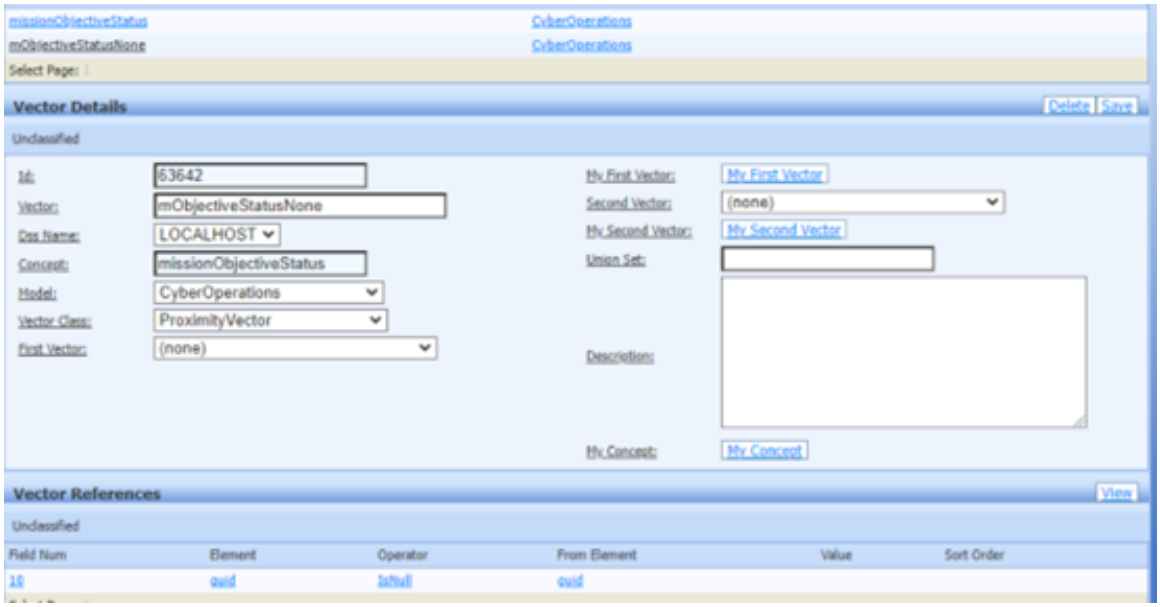


Figure 4.50. The image depicts the configuration of the mObjectiveStatus-None Vector in the missionObjectiveState Concept.

To ensure the Vector functions properly, the associated Reference needs to be configured such that the truth value holds. This is done by selecting the operator option of "IsNull." It is also pointed out that Vector References or relationships can implement any logical operator for specific relationship filtering capability.

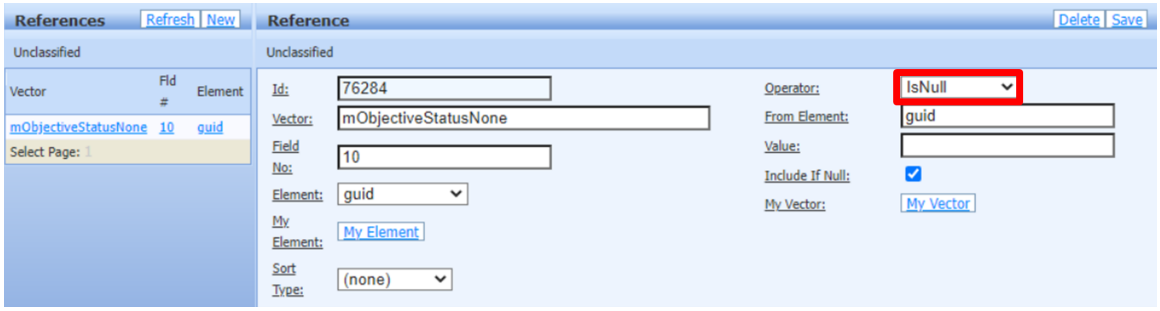


Figure 4.51. The image depicts the configuration of the mObjectiveStatus-None Vector References. Of importance is the "IsNull" indication for the operator value.

Once the Vector is configured, the linked action can be implemented. The linked action ensures proper sequential process takes the proper action in result to different inputs. Specifically, the linked action ensures the "missionObjectiveStatus" Form is updated with desired knowledge subsequent to the save action taken in the "DarkStaxXmpp" Concept.

To create the linked action, start from the "DarkStaxXmpp" Concept Form. From "Shortcuts," select the "Concept" link followed by the "My Actions" link. Select the "save" Action. On the "Action Details" page, there is an option to "view" linked actions.

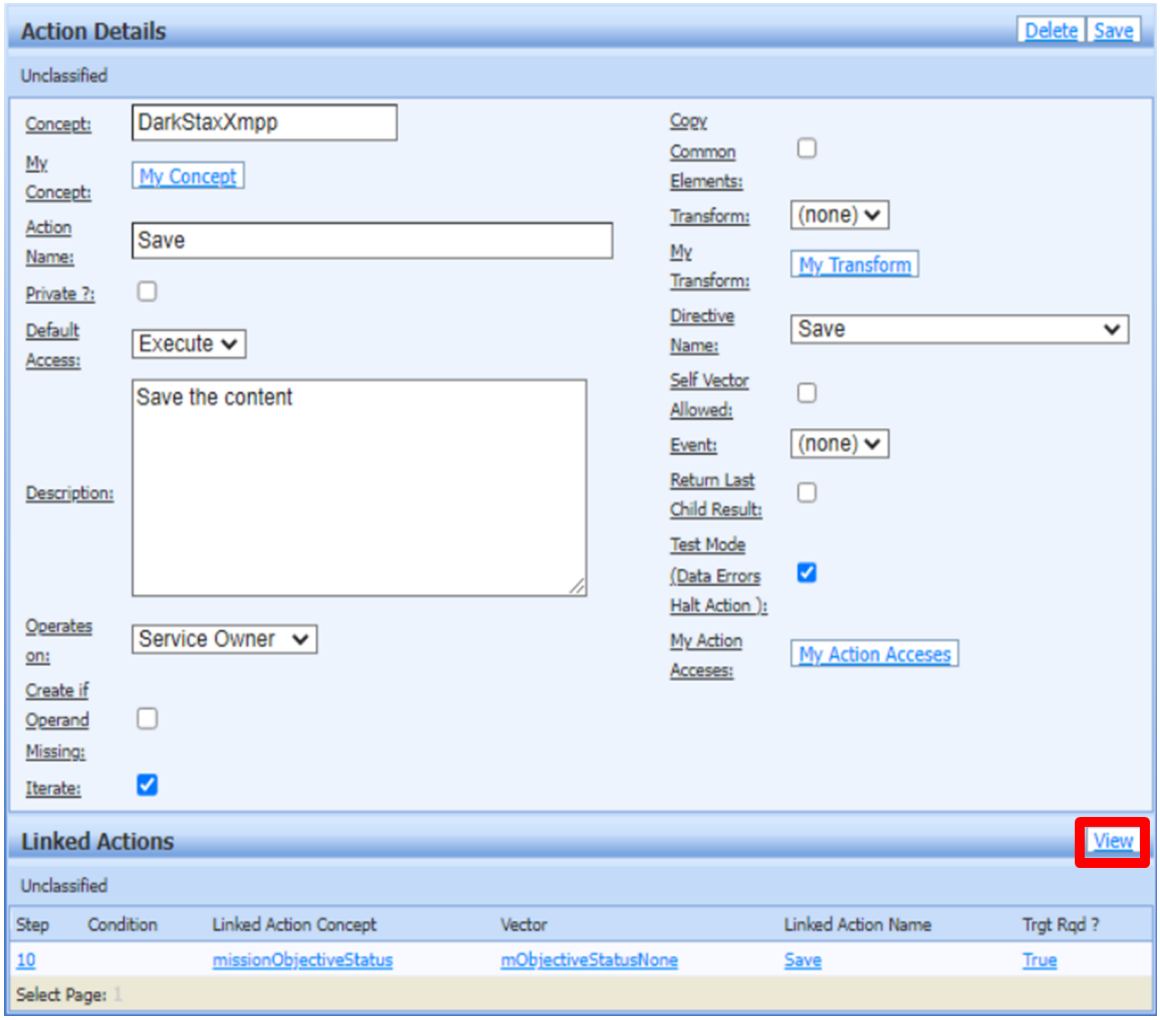


Figure 4.52. The image depicts the location to access the linked action list for the DarkStaxXmpp Concept.

From the "Linked Actions Details" page, configure a linked action that triggers the "save" linked action of "missionObjectiveStatus," this is subsequent to the save Action of "DarkStaxXmpp." Because of the complexity that is inherent in Vector related Concepts, the complex Actions usually are named by the system behavior that they cause. Though subtle, this is often different from traditional computer science methods. However it is actually the sequential executions of filters and analysis in real time across a graph based model of

Concepts and attributes that then sequentially execute actions in any Concept that relates to it is quite straight forward. In the human dimension, there is recognition of a task or action, there is no enumeration of all the micro instructions that are required to accomplish the task.

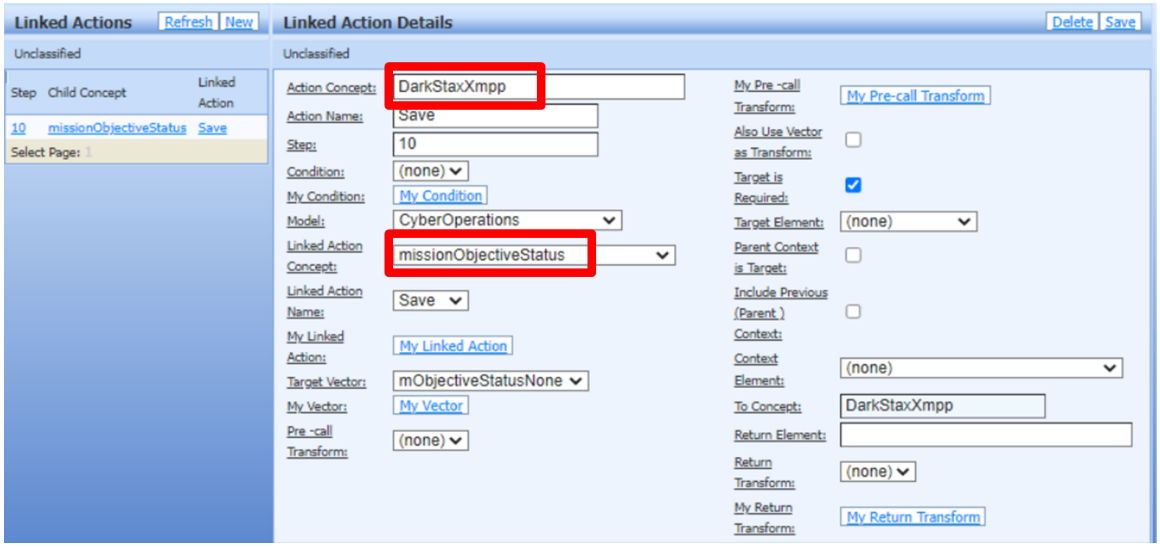


Figure 4.53. The image depicts the configuration of the linked action between DarkStaxXmpp and missionObjectiveStatus.

To summarize, the Actions and linked action are the primary components that relate the Data Concept with the Status Concept and cause sequential process to actuate in the GINA framework. As Image 4.54 shows, the save action that belongs to the "DarkStaxXmpp" Concept is linked to the subsequent save action that is part of the "missionObjectiveStatus" Concept. This linked action provides an additional layer of data transformation, which enables the execution of data to decision to complex action in all dimension.

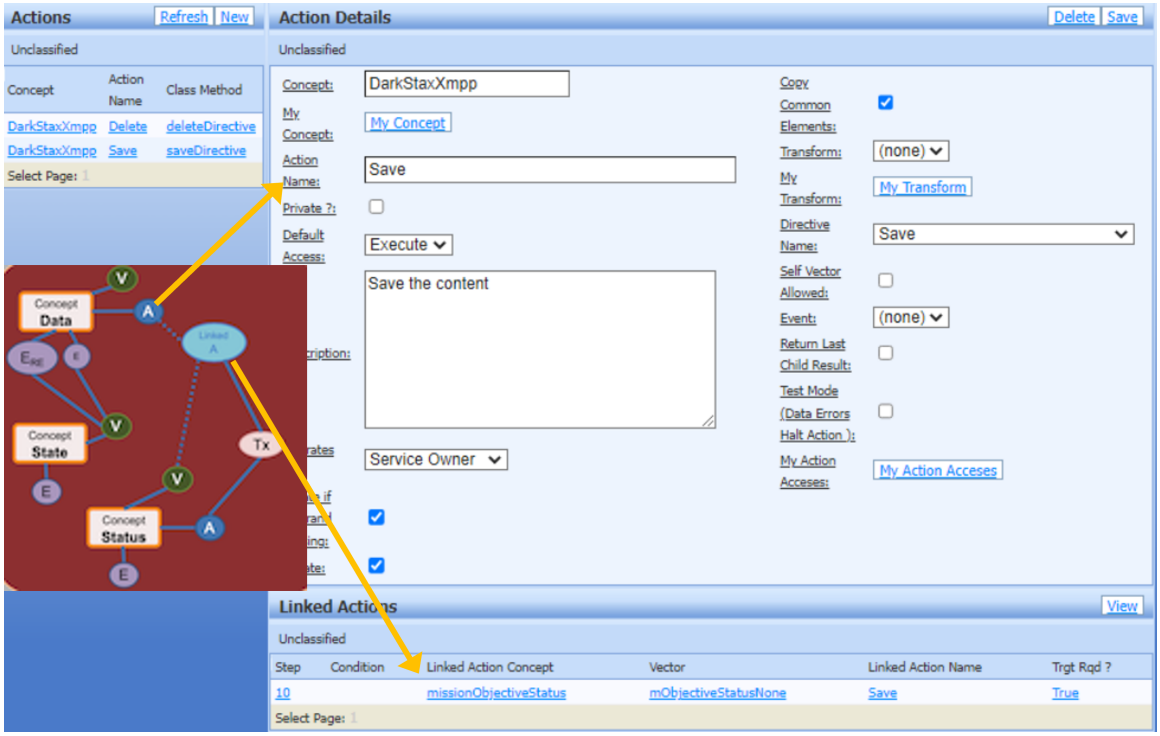


Figure 4.54. The action and linked actions relate the Status and Data Concept and enable decisions to inform and execute sequential action.

4.3 Input Channel Implementation

This section depicts the configuration of the input channel. This is the path that allowed the model described to intake and analyze the network traffic. It is the connection between JSON and XML data and the GINA decision model. For this specific experiment, the JSON and XML messages were created organically. However, the input channel is capable of absorbing massive streams of data generated by live networks.

Ingest channels are created for both JSON and XML messages to allow the model to accept both forms as input. For this research, the tests are conducted using the XML format. The ingest channels can be configured to read a data stream from any Transmission Control Protocol (TCP) or User Datagram Protocol (UDP) device by simply adjusting the channel's

connection to use the desired protocol. This channel, like the decision model, is intended to serve as a base model that can be easily adjusted to fit scenarios.

Figure 4.55 represents the components involved in the ingest channel and their relationships. These components and relations will be discussed further in this section. The main steps to properly set up the ingest channel are to configure the components in Figure 4.55 and configure the node type listener specific to this exercise, the COTS program configured to read a specified file from a specified folder.

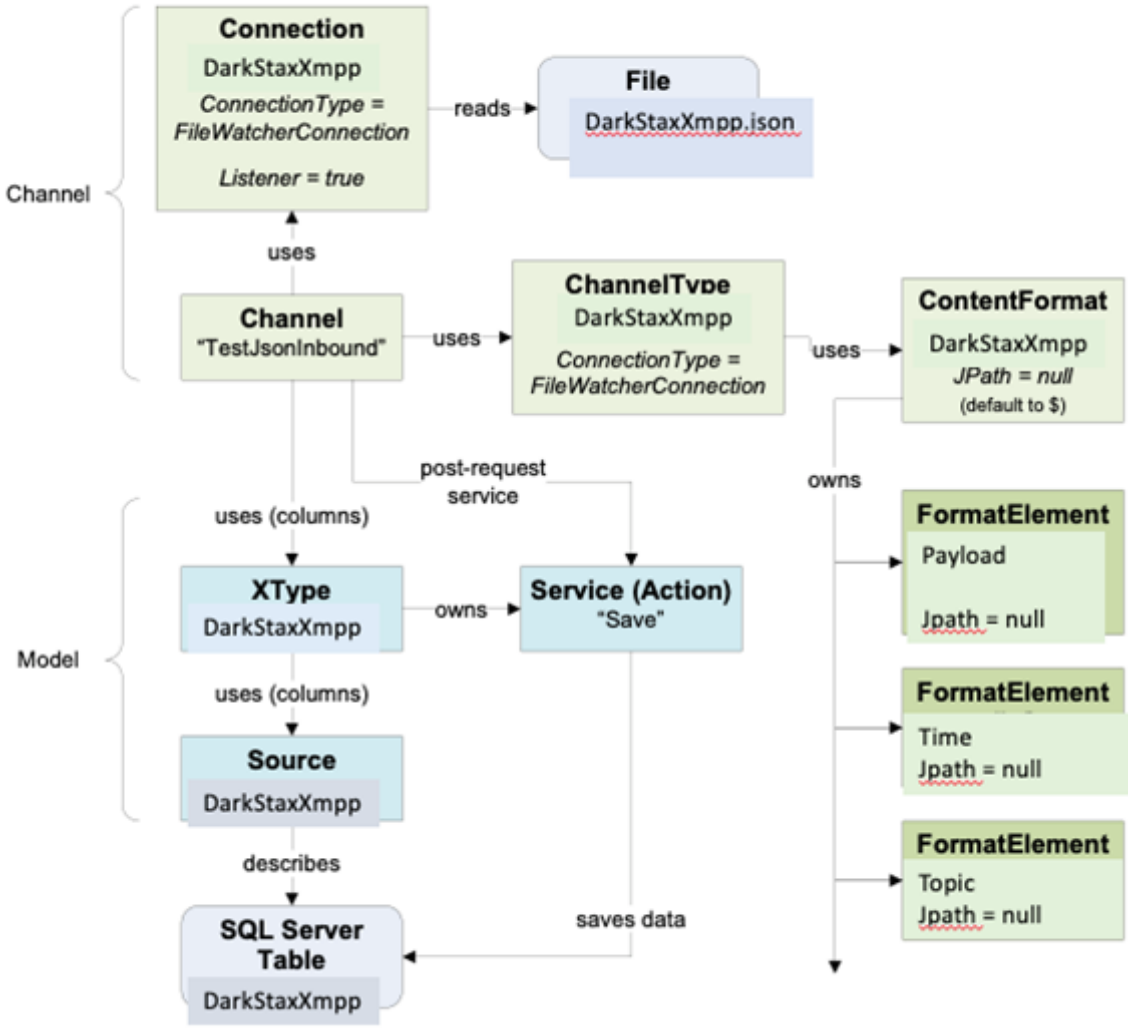


Figure 4.55. This model represents the components and channels implemented to enable the input of XMPP messages to the decision model.

4.3.1 Content Format

To configure the channel, the first component to create is the content format. Content formats describe the way data is laid out in specific types of messages or objects. Content formats

contain format elements that describe the data found in a particular message, specifying the underlying syntax of messages and individual pieces of data.

The content format configuration is found by selecting “development tools,” “channels,” “content format.” At the content format page, the most significant fields are the “content format” field, the “content format type” field, and the content “Jpath” field. The “content format” field is simply the name of the content format, in this case “DarkStaxXmppXml.” The “content format type” field is used to indicate the type of messages used, which for this exercise is XML. This is done to designate the format that the parser uses to analyze the message. The “JPath” is optional, but is used to point to the file or device from which the messages are generated. The content format page for this example can be seen in Image 4.56.

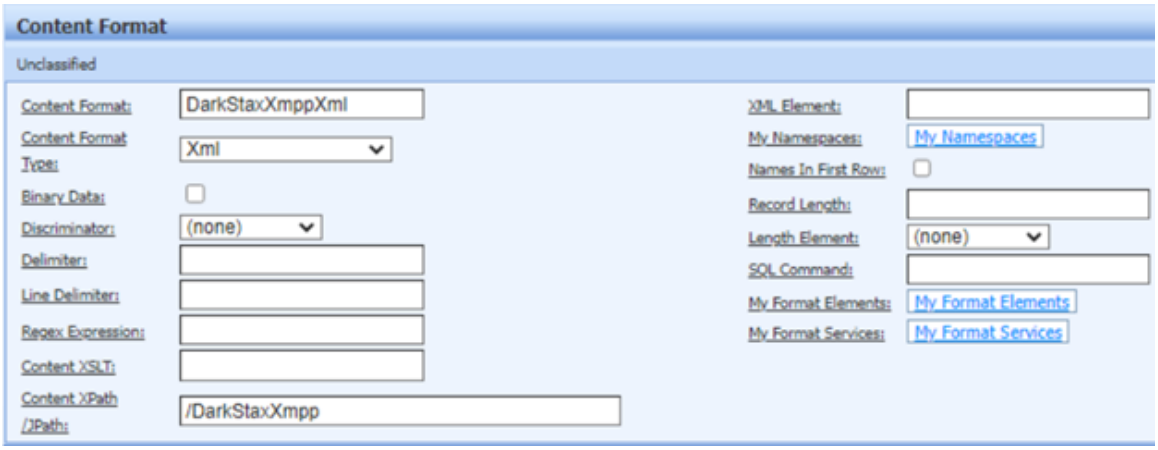


Figure 4.56. This image is an example of a complete content format page.

4.3.2 Format Element

The next component to configure is the Format Element. This section helps establish the format from which the messages are analyzed. A Format Element needs to be created for each part of the message. For this exercise, six format elements are created, to include: “creation date,” “id,” “jid,” “nodeID,” “Payload,” “serviced.” These six elements are the six designated components of OpenFire XMPP messages [19].

This section can be accessed by selecting the “My format Elements” option on the content configuration form. When creating a new format element, there are again three significant fields. “Content format” signifies the content format that the format element belongs to. “Format Element” is used to name the item being created. “XPath/Jpath” is again optional, but is used to point to the specific element in the XML message that contains the value of interest. Image 4.57 shows the creation of the “creationDate” format element.

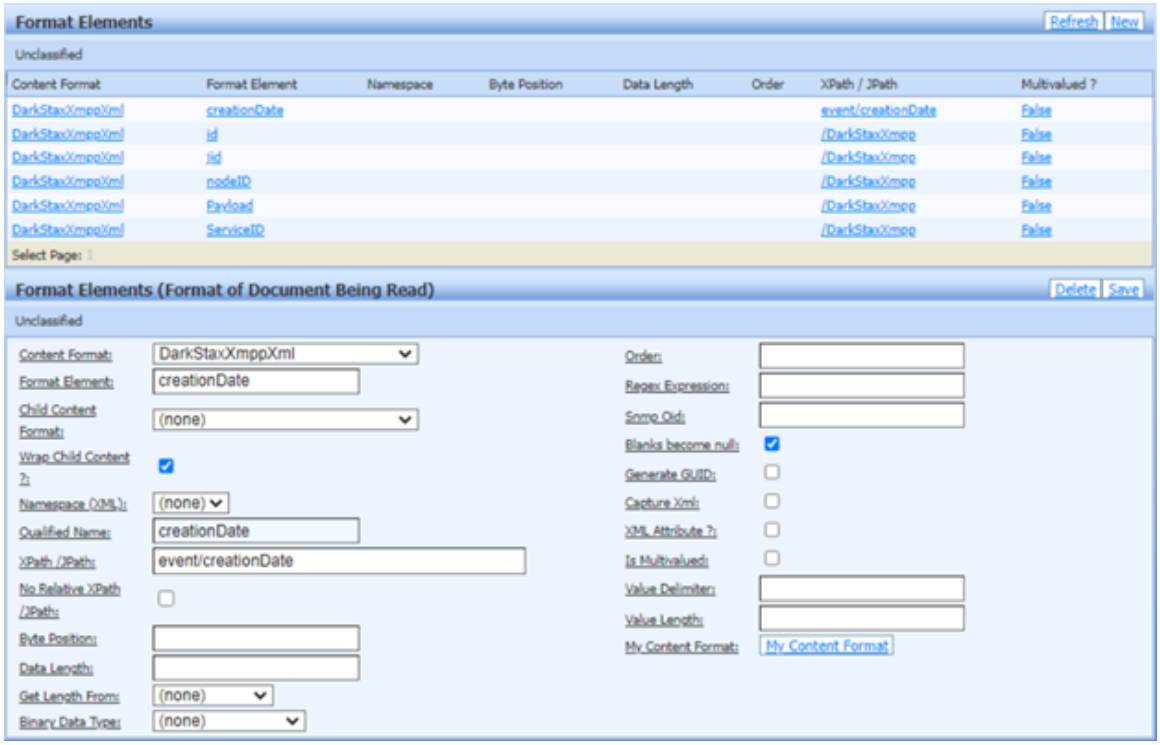


Figure 4.57. This image is an example of a complete Format Element page.

4.3.3 Channel Type

After content Format and Format Elements, the next item to configure is the Channel Type. Channel Types are used to describe the particular combination of data format and connection type, which in this scenario is data from a file. In general, these are created so that they can be reused by other channels when needed.

To complete this implementation, navigate to the Channel Type page via “development tools,” “channels,” “channel type.” The creation of the channel type requires five information fields. “Model” designates the association between the channel type and a particular model, usually the model that the channel’s Concept belongs to. The “channel type” simply states the name of the channel type. The “channel direction” specifies the direction of traffic flow with inbound meaning the messages are coming from an external source and outbound meaning the messages are being sent out. “Connection type” denotes the type of connection that must be used. The “content format” field identifies the format of the data that is expected over the connection.

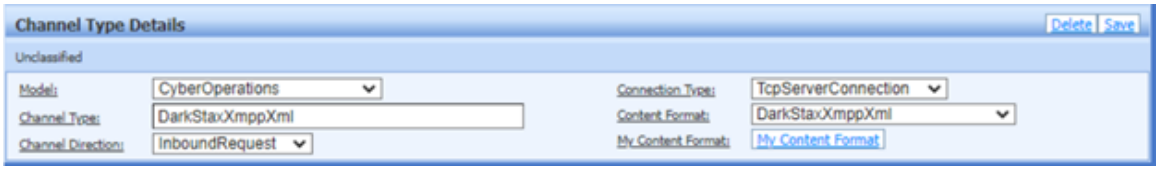


Figure 4.58. This image is an example of a complete channel type page.

4.3.4 Channel

After the channel type is configured, the channel itself can be created. For channel completion, the “Xtype” that will be populated from the channel needs to already exist. In this scenario, the “Xtype” that is populated is the “DarkStaxXmpp” Concept, which does exist. Channels uses a set of components that provide the desired behavior. The key components of channels are content format, channel type, and connections.

To configure the channel, navigate to the “maintain channel” page via “development tools,” “channels,” and “channels.” To create the channel, four fields must be populated. “Channel” refers simply to the desired name of the channel. “Channel Type” refers to the type of the channel, which specifies the format of the incoming data and the allowed connection type. “Request Content Type” refers to the existing Concept that will receive the incoming data. “Post-Request Service” designates the desired action that will be requested of the Concept once the incoming data has been parsed.

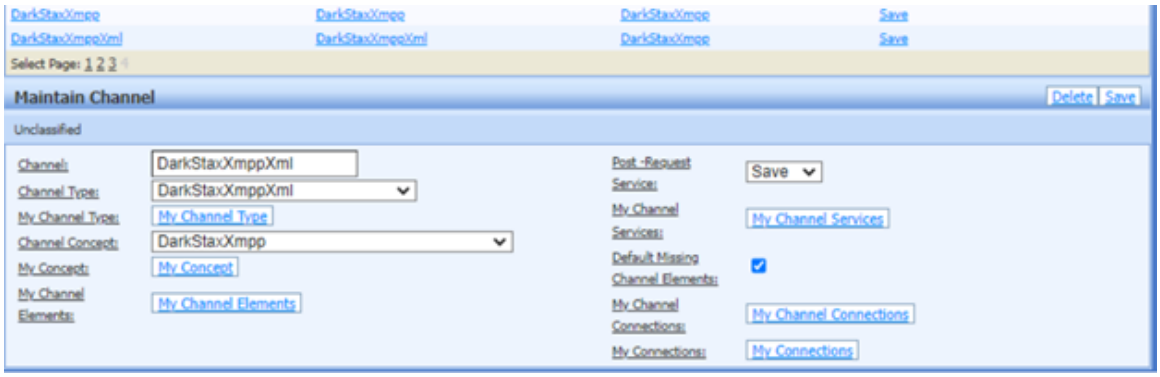


Figure 4.59. This image is an example of a complete channel page.

4.3.5 Connection

Next to implement is the connection, which specifies the protocol to be used to communicate with the external source and describes requirements to connect. In this scenario, the data is being read from files. For that purpose, the "FileWatcherConnection" and "Directory-Watcher" are used to read data any time the operating system indicates a change in the file directory.

To create the connection, navigate to the connection page via "development tools," "channels," "connection." To create a new connection, four fields need to be populated. The "Connection" field refers to the name of the connection. The "ConnectionType" designates the desired protocol, such as TCP or UDP. The "Is Startup" field indicates where the connection needs to listen for incoming messages. Since the "FileWatcherConnection" is used, the connection does need to the listen for changes to the source file. "File Path" describes the path to the location of the source file on the local file system.



Figure 4.60. This image is an example of a complete connection page.

4.3.6 Channel Connection

The final configuration step is to connect the channel with the connection created. Channels can use more than one connection and connections can have more than one channel. Due to this multifunctionality, a final configuration step is required to specifically connect the connection and channel created above. While in the channel configuration form, select the "My Channel Connections" option. From there, a simple channel connection form is displayed. The channel connection is created by filling in the channel and connection desired to be linked. "Channel" refers to the channel being connected, while "Connection" refers to the connection providing data to the channel. In this case, the channel and the connection is both "DarkStaxXMPPXML."

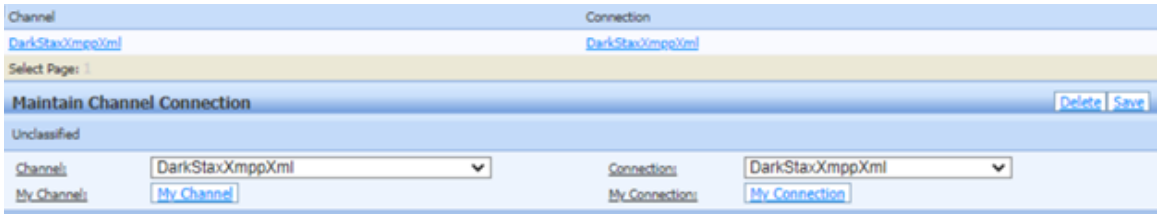


Figure 4.61. This image is an example of a complete channel connection page.

4.3.7 Node Type Listener

After the channel and connection are created, the Node Type Listener needs to be configured in order to activate the capability of the created channel. Configuration of the node type listen is done via "development tools," "background processes," "node type listeners."

Once at the node type listener page, the listener for both XML and JSON messages can be created. For this exercise, the XML messages input from COTS are analyzed. However, the JSON messages obtained through the file reader can be analyzed as well. The fields that need to be filled in are the node type, source, and connection. For the JSON connection, the connection field is the "DarkStaxXMPP," the source is "none" and the node type is "web server." For the XML connection, the connection is "DarkStaxXMPPXml," the source is "none," and the node type is "windows service." The XML connection configuration is visible in Figure 4.62.

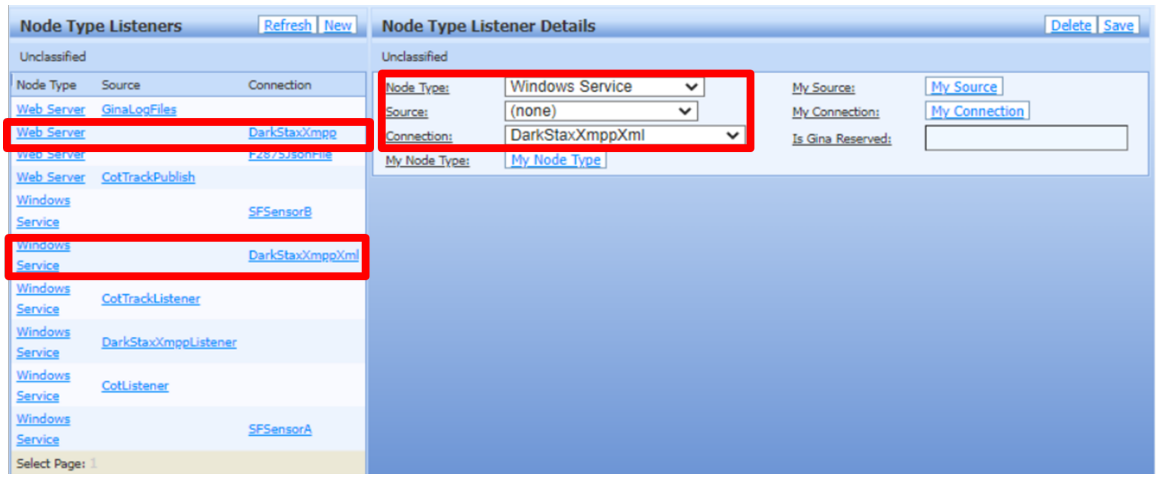


Figure 4.62. This image is an example of a complete node type listener connection.

4.3.8 COTS Configuration

The final component of implementation, though not part of the model itself, is used in this research to conduct testing and simulation. The COTS simulator is configured to read

XML messages from a file and output them as publish/subscribe messages to a designated Internet Protocol (IP) and port. The GINA service listens to the port to input them into the decision model. This simulates the behavior of a sensing technology scraping PCAP from large network data for XMPP messages and forwarding them to GINA. The input messages use the same form as potential filtered XMPP messages.

To enable this capability, the COTS machine is set up as described and depicted in Image 4.63.

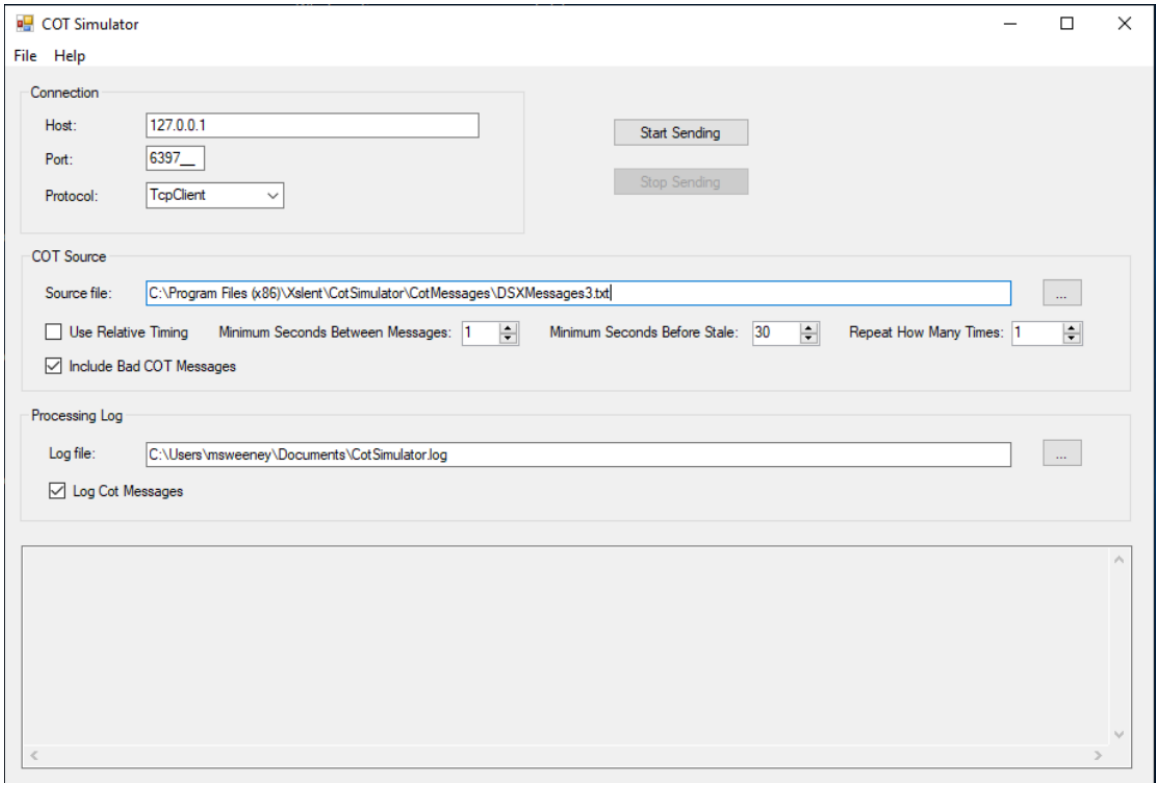


Figure 4.63. COTS simulator is used to facilitate testing of the model. The simulator is configured accordingly in order to read the specified files containing the messages to send and to the specified IP and port of listening GINA server.

The connection information needs to match the information configured for the GINA server

and ingest channel. The host address should read “127.0.0.1,” the port reads “6397,” and the protocol is “TcpClient.” The source file must state the path to the file that contains the messages to be read. For this exercise, a directory was created in Program Files (x86)\Xslent\CotSimulator\CotMessages\FileName. The “FileName” used in this test is “DSXMessages3.txt” which contains the four UML messages to be input. Additionally, the option to “Include Bad COT Messages” should be enabled, as the messages do not follow the exact desired format of the program. Enabling this option allows for the messages to still be uploaded. Successful transmission of messages should show the total number of messages sent with a matching number of errors due to the programs perceived “Bad Cot Messages.” This does not affect the functionality of the decision model.

CHAPTER 5:
Testing and Results

The section itself is broken down into two main subsections: testing and results. The testing section will discuss the overall technical process of the tests. It will then breakdown the details of each of the four tests conducted, display the contents of the test messages, restate the criteria introduced in chapter three, and state the expected results based on analyzing the contents of the message against the criteria.

The results section will show if the expected results discussed in the testing sections were the achieved results. It will show the real-time output of the alert status based on the message content. It will also show the real time output of the recommended actions that are based on the real-time alert status. The machine takes the message as input, compares it to the criteria, signals the proper alert state based on the criteria and message, then recommends actions associated with the alert status.

5.1 Testing

The purpose of the testing is to verify the model was designed and implemented correctly. The test will utilize the COTS simulator application to generate sensor data for real time testing. The COTS simulator is configured according to 4.3.8.

5.1.1 Test Procedure

While the purpose of the test is to verify the functionality, the tests were also designed to replicate the classification of large scale tactical networks. The COTS configuration simulates the sensor capability of scraping and filtering of XMPP messages from PCAP. It reads the contents of a text file and sends the contents to the IP and port that the GINA services is listening on. For the proof of concept, the file contains four XML messages, written in form of OpenFire chat services. Each message is intended to exactly match a particular instance criteria of the use case.

The COTS simulator runs one time with the file that contains all four messages. Initiate

the transmission by selecting the “start sending” option, depicted in Figure 5.1. Once the entire file is sent, the machine will automatically stop transmitting and display the status of the messages sent. The text box should state that five messages were sent correctly, as seen in Figure 5.1. There should also be five corresponding errors based on the difference in format, but that does not affect the decision model. This is attributed to the fact that COT message format is more complex and unique than pure JSON and XML, although similar. Regardless, the simulator will process and send the plain XML or JSON and simply trigger the flag that the message is not COT.

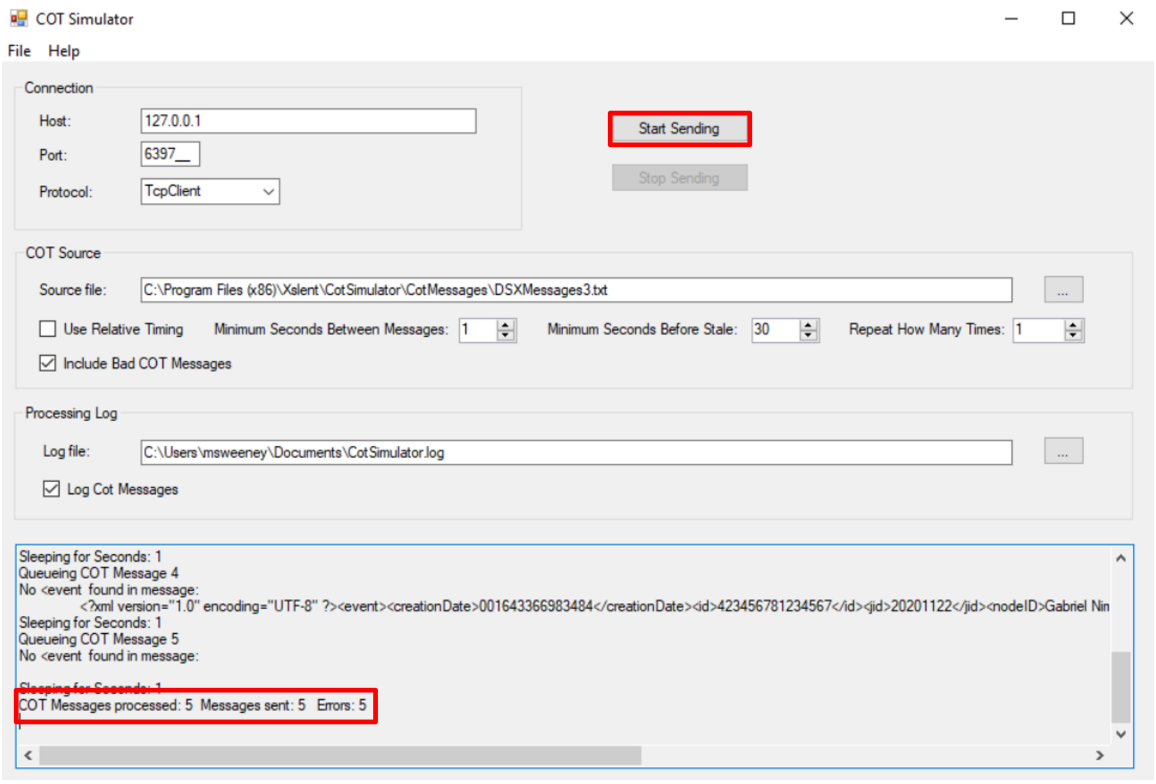


Figure 5.1. COTS transmission of messages to GINA via the GINA service and ingress channel.

At this point, the messages have been sent to the decision model, which will interpret each UML message individually.

5.1.2 Final Criteria

The criteria remains the same as was presented in section 3.3.3.

5.1.3 Messages and Expected Output

Message 1

The content of message one is as follows:

```
<?xml version="1.0" encoding="UTF-8"
↪ ?><event><creationDate>001643366983181</creationDate><id>123456
↪ 781234567</id><jid>20201122</jid><nodeID>Gabriel Nimbus, Army
↪ Big Data
↪ Platform</nodeID><serviceID>pubsub</serviceID><Payload>Sensors
↪ detect traffic is none</Payload></event>
```

Of most importance, the Payload section states that "sensors detect traffic is none." The "traffic is none" text matches a section of the defined criteria. Based on that match, it is expected that the model identifies the green-None bit as true, which should result in green objective state.

Message 2

The content of message two is as follows:

```
<?xml version="1.0" encoding="UTF-8"
↪ ?><event><creationDate>001643366983282</creationDate><id>223456
↪ 781234567</id><jid>20201122</jid><nodeID>Gabriel Nimbus, Army
↪ Big Data
↪ Platform</nodeID><serviceID>pubsub</serviceID><Payload>Sensors
↪ detect traffic is low</Payload></event>
```

Again, the most important aspect of the message is found in the Payload section, which contains the text "traffic is low." Based on the criteria, this should cause the model to signal the green bit, which should result in a green objective status.

Message 3
The content of message three is as follows:

```
<?xml version="1.0" encoding="UTF-8"
↪ ?><event><creationDate>001643366983383</creationDate><id>323456
↪ 781234567</id><jid>20201122</jid><nodeID>Gabriel Nimbus, Army
↪ Big Data
↪ Platform</nodeID><serviceID>pubsub</serviceID><Payload>Sensors
↪ detect traffic is medium</Payload></event>
```

This payload section indicates that "traffic is medium." This condition should create a true yellow bit, which should then result in a yellow objective state.

Message 4
The content of message four is as follows:

```
<?xml version="1.0" encoding="UTF-8"
↪ ?><event><creationDate>001643366983484</creationDate><id>423456
↪ 781234567</id><jid>20201122</jid><nodeID>Gabriel Nimbus, Army
↪ Big Data
↪ Platform</nodeID><serviceID>pubsub</serviceID><Payload>Sensors
↪ detect traffic is high</Payload></event>
```

The "traffic is high" text in this message indicates that the red bit and red objective state should be associated with this message.

5.2 Results

After the messages were transmitted by the COTS machine, the GINA model displayed all four messages in the XMPP and Status Concept Forms. The results are all in accordance with expectations and are discussed below.

5.2.1 Message1 Results

Message 1 contained the words “traffic is none” which means a status of green objective state was expected. This result was achieved by both the “DarkStaxXmpp” Concept and “missionObjectiveStatus” Concept. Depicted below in images 5.2 and 5.3. are the output on the XMPP Concept and the output of the status Concept, respectively.



Figure 5.2. The Dark Stax XMPP Concept associates the message with a green-none bit, as per the criteria. The associated objective state is green, according to the truth table.

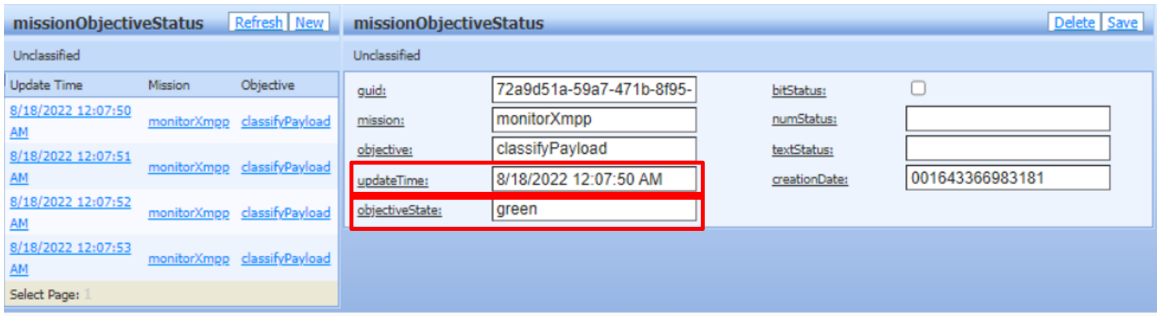


Figure 5.3. The status Concept saves the appropriate green objective state for this message.

Image 5.2, which displays the XMPP Concept, highlights the Payload message, the appropriate true value of the green-none bit, and the appropriate objective state which was retrieved from the state Concept.

The status Concept highlights that it too displays the proper objective state determined by the truth table. Both images also highlight the date time to prove they are referring to the same message.

5.2.2 Test 2 Results

Message 2 resulted in a green bit status and green objective state, as expected with the content of "traffic is low." The results can be seen below in Images 5.4 and 5.5. Both images highlight the same fields as message 1.

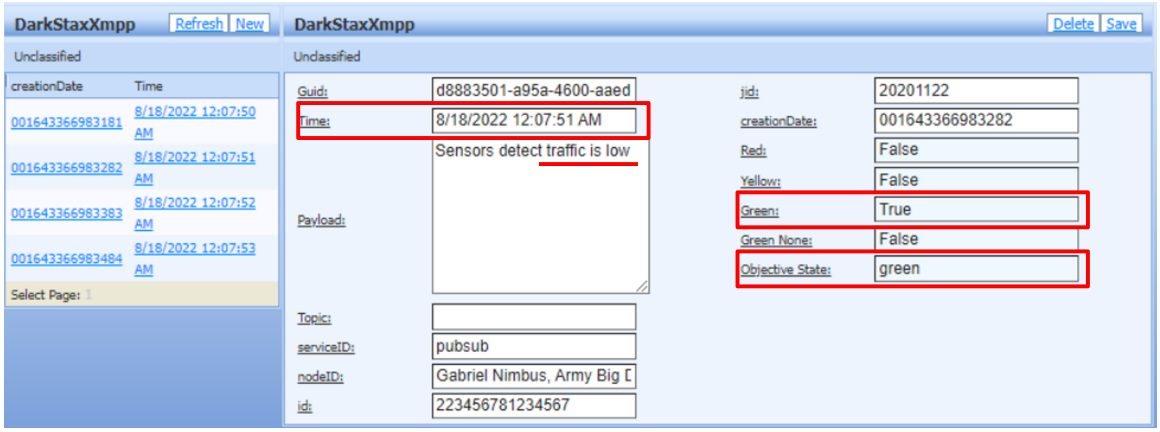


Figure 5.4. The Dark Stax XMPP Concept associates the message with a green bit, as per the criteria. The associated objective state is green, according to the truth table

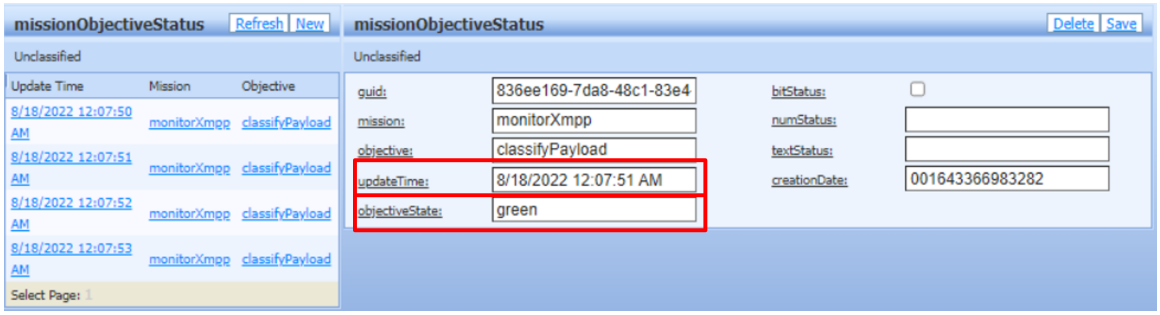


Figure 5.5. The status Concept saves the appropriate green objective state for this message.

5.2.3 Test 3 Results

Message 3 resulted in a yellow bit status and yellow objective state. This is in accordance with the criteria and expectations. Results can be seen in images 5.6 and 5.7.

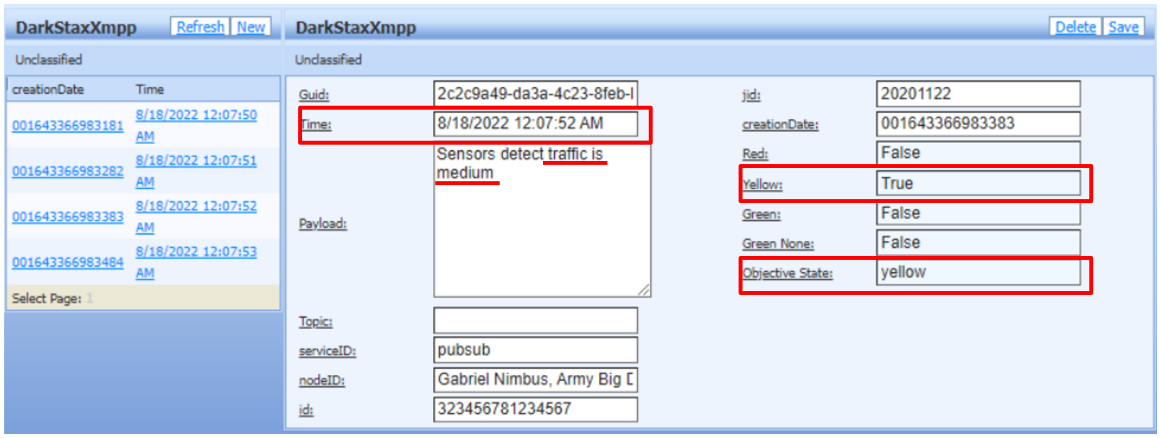


Figure 5.6. The Dark Stax XMPP Concept associates the message with a yellow bit, as per the criteria. The associated objective state is yellow, according to the truth table

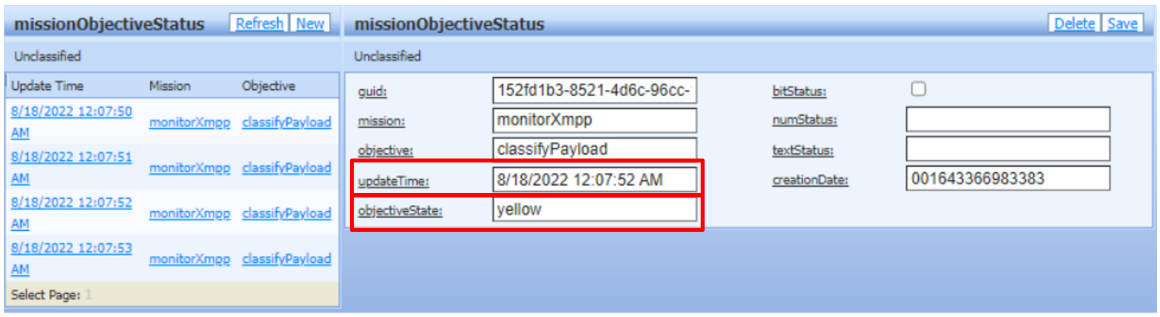


Figure 5.7. The status Concept saves the appropriate yellow objective state for this message.

5.2.4 Test 4 Results

Message 4 produced results of a red bit status and red objective state, as expected considering the "traffic is high" payload. Results for the "DarkStaxXmpp" and "missionObjectiveStatus" Concepts can be seen in figures 5.8 and 5.9.

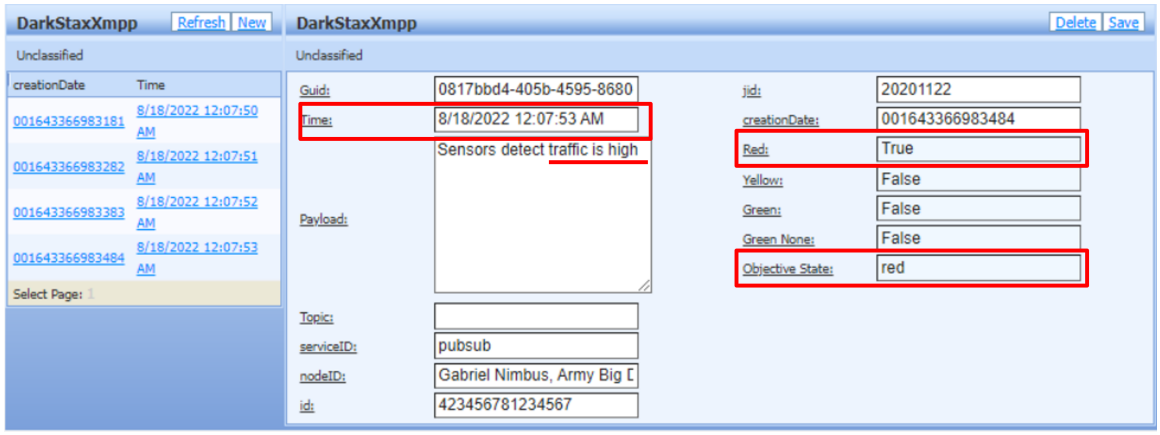


Figure 5.8. The Dark Stax XMPP Concept associates the message with a red bit, as per the criteria. The associated objective state is red, according to the truth table

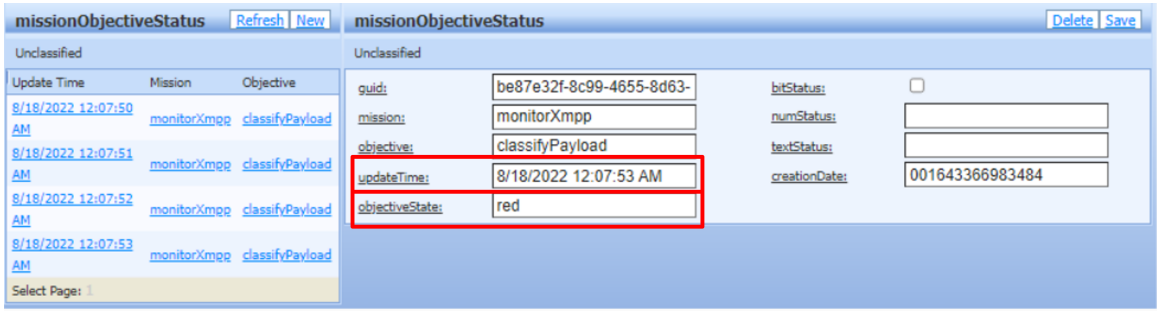


Figure 5.9. The status Concept saves the appropriate red objective state for this message.

THIS PAGE INTENTIONALLY LEFT BLANK

CHAPTER 6:
Conclusion and Future Work

This chapter summarizes and concludes the experiment. The experiment outcome is viewed positively as the main objective of designing and implementing the decision model was achieved. The model was implemented and tested to prove functionality. It must be emphasized that this network application was entirely configured and at no time compiled. Further, the developer can access any component to modify or change according to mission needs. This chapter further discusses possible use cases and how a VRDM based decision model could be applied to those scenarios. The examples provided are USMC focused and range from tactical scenarios to diagnostics of large scale networks. Finally, the chapter presents a portion of the vast opportunities for potential future work. The applicability of this model and GINA to the DOD is extensive. The scenarios presented here are options for the immediate next steps.

6.1 Conclusion

The main contribution of this research is the design and implementation of a ubiquitous decision model. This should have a huge impact because it can serve as the base of any decision model implemented by MDOC5i. The model implemented here provides a structure that can be used in all environments for a wide range of purposes. The established starting point will enable GINA to be configured quickly in response or preparation for any scenario, thus equipping the Marine Corps with technology capable of achieving information overmatch.

A secondary objective of this research was to show that the VRDM platform could do what normal programming can do. For future work, the use cases should be expanded to include more complexity, which would highlight the benefits of GINA and VRDM. Because of the simplistic use case, the benefits of GINA were not on full display as this use case could have been programmed using normal programming languages.

The hypothesis was that a ubiquitous decision model that executes process could be designed and implemented using VRDM and GINA. That was proven correct, as the model was designed to take advantage of VRDM constructs and implemented in the GINA framework.

Secondary questions of this research dealt with the expansion of the model to broader use cases and the direct comparison of VRDM techniques to traditional modeling techniques. Though these questions were not definitely answered during this thesis, the model should enable future related work dedicated to these questions that provides definitive answers.

6.2 Extensive Military Use Cases

For practical purposes, this section discusses potential scenarios by which the decision model could make improvements to current practices. The intent of the ubiquitous model is to fit multiple types of scenarios ranging from tactical decisions to complex cyber networks.

6.2.1 Full Network Scenario

The first scenario to visualize is that of a network health diagnostic as depicted in the hypergraph portrayed by Image 3.3. The intent of this project originally was live network implementation with a focus on XMPP, that should be the first step, but full implementation should be the goal. It would be relatively easy to model all aspects of the hypergraph when employed by a functional unit with a live network. The unit would need to document all physical aspects of their network so they could be added to the input portion of the decision model. The unit could also document all significant, real-world events that affect the unit, to include holidays, distinguished visits, scheduled maintenance, weather phenomenon, and confirmed attacks. These criteria could be given weights based on significance and impact and could be combined to make an aggregate score across all influential areas. This aggregate score could then be applied to a Likert scale that would determine the health of the network and would recommend an alert status and associated course of action. This is the full version of the simplistic test implemented in this research.

The preparatory work for this model would not be significant, as physical network components are already accounted for and real-world factors are also accounted for and could be added as necessary. The resulting model would be able to track and diagnose all factors of a network in real time.

The network scenario depicts the cyclic, self-sustaining nature of the GINA model. Each hierarchy is the result of different input factors. The factors not only influence what follow-

on actions should be made, but also influence the hierarchy state, signifying that information will be treated differently based on the current state.

This XMPP would be combined with other inputs to evaluate the system and kick off a new state. It is important to understand the incoming information changes the state of the mission and therefore information will be treated differently. The ubiquitous model's behavior is narrative based. Incoming information changes the mission and therefore changes the way the model looks at and treats new incoming information.

While the machine operates in a hierarchy, it evaluates the state based on inputs and moves accordingly. For example, the machine could start in the normal alert status, conducting the actions associated with "see network." Based on PCAP input, the machine could detect malicious activity that qualifies for the heightened alert status, and would thus transition to "secure network" actions. Once indications confirm the situation is resolved, the machine would revert back to the normal alert status and conduct "see network" actions. This cycle depicts the continuous and self-adapting capabilities of the system.

6.2.2 Tactical Scenario - Convoy Takes Fire

Consider a scenario where a convoy is engaged by enemy fire. The commander must make a quick an effective decision on how to neutralize the situation while having little definitive information. The ubiquitous decision model could be fit to this situation and aid the decision maker.

This model takes in two inputs: vehicle diagnostics and enemy location. To achieve the enemy location, this model would use the Laser Imaging, Detection, and Ranging (LIDAR) abilities. There are four possibilities based on the combination of inputs. Based on the inputs, the decision model could recommend a course of action immediately after taking fire based on inputs of enemy location and vehicle diagnostics.

1. *Engage while driving*. This would be the recommended action if the enemy location is known and the vehicle is not damaged. In this scenario, the unit could engage the enemy using truck mounted weapons while continuing to drive through the area and evade the danger zone.
2. *Drive to evade*. This would be the recommended action if the enemy location is

unknown and the vehicle is not damaged. In this scenario, the primary objective is to exit the danger zone.

3. *Engage from stationary vehicle*. This would be the recommended action if the enemy location is known and the vehicle is damaged. In this scenario, the vehicle is unable to continue but the enemy location is known. The primary objective is to engage the enemy so that repairs can be made.
4. *Provide security*. This would be the recommended action if the enemy location is unknown and the vehicle is damaged. In this scenario, the vehicle cannot continue, and the enemy location is unknown. The primary objective is to provide security effectively so repairs can be made.

This recommended course of action would be provided immediately after taking fire, providing the decision maker with the objectively best course of action immediately. No time would be wasted trying to figure out if the enemy location is known or if the vehicle is damaged. Further, the initial shock of taking fire would be partially mitigated by the recommended action. The recommender would help mitigate the friction and fog of war that is common in this scenario, thus leading to quicker and better decisions regarding course of action.

Another benefit to point out with this scenario is the interoperability that MDOC5i emphasizes and enables. MDOC5i is not a "black box" system that limits the user to its own organically provided capabilities. It can incorporate emerging capabilities, such as this LIDAR capability, seamlessly without needing to design their own or configure compatibility. The benefit is not a single standalone package as it often is with other groundbreaking tactical systems. The benefit is the base system that can incorporate all capabilities into its decision model. That way the system can improve as capabilities do and can be tailored to specific needs cases.

This is a simple case with only two inputs and a handful of decision to recommend. Even with the small number of inputs and outputs, it should be evident how useful this recommendation would be if it was immediately provided at the moment of contact. However, the system can incorporate much more and can drastically increase internal complexity while still providing the decision maker with a clear cut, objectively sound decision based on identified inputs. This can be imagined by expanding this scenario. The decision model and system could take

geographic conditions, map information and destination information as additional inputs. More importantly, it could incorporate information and intelligence from other units. It would be critical to know if the initial contact was just a precursor for heavier enemy resistance ahead. The model could know that based on friendly intelligence and factor that into the recommended decision. The machine not only makes quicker objective decisions but knows more information due to the interconnected information system that enables the CIP and COP. This enables the decision maker to think clearly and focus on the situation.

6.2.3 Tactical Scenario - Defensive Area Operations

Another tactical situation this model could apply to is in the scenario of a defensive position preparing to engage incoming hostile forces. The scenario depicts a large number of various enemy vehicles approaching a position with limited weapons and ammunition. The decision model could be built with all known information about enemy vehicles, to include safety features, armor capability, and physical appearance. It could also be programmed with all statistics and ballistic information of friendly firing capabilities and updated to know the exact weapons and ammunition counts at the location. Inputs to the model would be a live terrain map and a live video or satellite stream of the area. Based on the loaded descriptions, the model could identify the enemy vehicles and compare the individual vehicle statistics and total number of vehicles with the capabilities and quantities of fires capabilities preparing to engage. The model could then produce an engagement plan to include which weapons should engage which vehicles based on effectiveness and even when the weapons should engage certain vehicles based on distance and terrain. This plan would objectively provide the best target match for each weapon system in combination with the best firing times based on vehicle distance and effective weapons distance. This effectively integrate all available weapons and provides each weapon operator with their identified targets, order to engage targets, and time at which to engage each target.

Additionally, the model could provide the operator with feedback on the proper aiming point to effectively engage the target, as demonstrated during the live fire exercise at Twenty-Nine Palms. Thus, in addition to providing a specific weapons engagement plan and time of engagement, the model would ensure the operator manipulates the weapon correctly to ensure effective results. This would ensure no rounds of ammunition are wasted and would most efficiently engage the enemy.

This example also shows that the model inherently improves communication and fires coordination. Because optimal targets are provided to all weapon systems, no two weapon systems should engage the same target, unless specifically designed to do so. This would prevent all personnel from focusing on one vehicle and wasting limited resources.

6.3 Future Work

Future work associated with this research should be explored in order to maximize the utility of the decision model. Future topics should focus on increasing the complexity of the use case, as well as directly comparing results from a VRDM model to traditional data manipulation techniques.

Future work associated with this model should focus on expanding the use case to include more complexity. The most direct way to do this would be to use fabricated data that replicates full XMPP traffic in conjunction with complex diagnostic criteria that aggregates all incoming traffic. The same model created could then be applied to captured or live USMC traffic. These tests would prove functionality in more realistic and complex scenarios and would also show the benefits of integrating GINA/VRDM models with DarkStax technology.

In addition to work done directly related to USMC traffic, future work should be done to compare GINA model's capabilities with that of tradition machine learning. This work should be done directly with other NETCOM related theses undertaken at NPS. Machine learning techniques are used for large scale network analysis. Future work should plan for the creation of a GINA model that seeks to achieve the same analysis on a large-scale network as employed machine learning techniques. This would provide a direct comparison between the VRDM model and traditional programming methods. This would prove or disprove that VRDM models are able to provide the same functionally as machine learning. During implementation and design, it could also provide context as to the ease in which the model can be constructed. The feasibility and ease of constructing an effective model that replicates machine learning results would be a great starting point to support the claim that VRDM models can provide the same capabilities as machine learning with more flexibility.

The final phase of future work would be full integration with the Marine Corps. The first step of this would be for the MDOC5i team to embed with Marine Corps units and manipulate

the decision model to the needs of that unit and show the utility of the model. The next part would be to teach Marine Corps units how to use the models and configure the models organically. Another part of this phase would be to implement a more complex and complete decision model. This would still have to be a generic and ubiquitous model that could be manipulated and configured based on specific scenarios. However, a more robust baseline would be beneficial if the goal is to enable Marines to use it organically. The baseline could include more preconfigured options that need to only be selected and modified, rather than created. Multiple versions of this baseline could exist for different units. These could serve as templates to choose from given different situations.

Overall, the potential for VRDM/GINA models is unlimited if the claims are proven to be correct. The future work for this project should work towards proving those claims and then spreading this methodology to all areas of the military.

THIS PAGE INTENTIONALLY LEFT BLANK

List of References

[1] *Summary of the 2018 national defense strategy of the United States of America : sharpening the American military's competitive edge.* Washington, DC: Department of Defense, 2018.

[2] *Commandant's planning guidance*. Washington, DC: United States Marine Corps, 2019.

[3] Interaction Design Foundation, "Human-Computer Interaction (HCI)." Accessed July 10, 2022 [Online]. Available: https://www.interaction-design.org/literature/topics/human-computer-interaction#:~:text=Human%2Dcomputer%20interaction%20(HCI)%20is%20a%20multidisciplinary%20field%20of,(the%20users)%20and%20computers.

[4] *Vision and Strategy for 21st Century Learning*, TECOM Vision and Strategy v13, Training and Education Command, Washington, DC, USA, 2020 [Online]. Available: https://www.tecom.marines.mil/Portals/90/3284_TECOM%20Vision%20%26%20Strategy_v%2013_1.pdf

[5] R. F. Kelly, "Automated cyber threat analysis and specified process using vector relational data modeling," Gallup, Shelley, September, 2014.

[6] M. AC/S Communications Directorate (G-6) MAGTF TC, "Joint MDOC5i interim authority to test (IATT) with interim authority to connect (IATC)," MCSC SCA Determination, Twenty-Nine Palms, CA: USA, 2022.

[7] D. T. M. Piscotty, M. Zepeda and T. Anderson, "Multi-domain operations command, control, computers, communication, combat systems, and intelligence," Lawrence Livermore National Laboratory., Livermore, CA, USA, Tech. Rep. Version 0.1, 2021.

[8] T. Anderson, "Global information network architecture vector relational data modeling: Game implementation," VRDM Application Level Brief, Monterey, CA, USA, 2018.

[9] D. Dolk, T. Anderson, F. Busalacchi, and D. Tinsley, "GINA: System interoperability for enabling smart mobile system services in network decision support systems," in *2012 45th Hawaii International Conference on System Sciences*. IEEE, 2012, pp. 1472–1481.

[10] T. J. S. I. Group, “Global information network architecture: Multi domain operations c5i (MDOC5i) service level training exercise (SLTE) objectives,” March 28, 2022, unpublished.

[11] F. Martinig, “Staruml – open source UML tool,” Methods and Tools, 2011 [Online]. Available: https://www.methodsandtools.com/tools/staruml.php

[12] MKLabs Co. Ltd., “StarUML Documentation.” Accessed Jun. 10, 2022 [Online]. Available: https://docs.staruml.io/

[13] G. Stevenson, M. Naiman, A. Boyd, and A. L. Valenta, “Cursor on target: Research for a sensor network,” *Online Journal of Public Health Informatics*, vol. 4, no. 3, Dec. 2012. Available: https://journals.uic.edu/ojs/index.php/ojphi/article/view/4275

[14] L. Eckart, S. Eckart, and M. Enke, “A brief comparative study of the potentialities and limitations of machine-learning algorithms and statistical techniques,” *E3S Web of Conferences*, vol. 266, pp. 2001–, 2021.

[15] B. K. Technologies, “BKT memo to team GINA to assist in differentiation of GINA efforts,” unpublished.

[16] J. M. Seng, “Behavior-based network management: a unique model-based approach to implementing cyber superiority.” SPIE, 2016, vol. 9826, pp. 98 260H–98 260H–15.

[17] K. Keigwin, “Sensor fusion and target identification: Implementation details,” Draft, Big Kahuna Technologies, LLC, July 6, 2020.

[18] L. Almonte, E. Guerra, I. Cantador, and J. De Lara, “Recommender systems in model-driven engineering,” *Software and Systems Modeling*, vol. 21, no. 1, pp. 249–280, 2022.

[19] Ignite Realtime, “OpenFire Database Schema Guide,” Accessed June 5, 2022 [Online]. Available: https://download.igniterealtime.org/openfire/docs/latest/documentation/database-guide.html#ofPubsubItem

Initial Distribution List

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California