A COGNITIVE RESOURCE MANAGEMENT FRAMEWORK FOR AUTONOMOUS GROUND VEHICLE SENSING

By

GREGORY A. GARCIA

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY:

UNIVERSITY OF FLORIDA

2010

To Rachael, Gregorio, Miriam, and Rosemarie

ACKNOWLEDGMENTS

I would like to start by thanking my family for their continued support over these last nine years as I have pursued both undergraduate and graduate degrees while attending the University of Florida. They have encouraged me to strive for the very best, and they have always led by example. I thank my wife Rachael for her patience and understanding, for putting everything on-hold so I could further my education. She was my source of motivation when the tasks seemed most daunting. She is my biggest fan and best friend.

I would also like to express my gratitude to Dr. Carl Crane for inspiring me to pursue a career in robotics and for affording me all the opportunities I have received while being a graduate student at the Center for Intelligent Machines and Robotics. His guidance has helped me navigate my way through my undergraduate and graduate education. Similarly, I would like to thank my committee members, Dr. Antonio Arroyo, Dr. Douglas Dankel, Dr. John Schueller, and Dr. Gloria Wiens for their valuable guidance over the years. This research would not have been possible without the support of the Air Force Research Laboratory at Tyndall Air Force Base in Panama City, Florida. I offer a sincere thank you to their staff for allowing me to work on-site in collaboration with them over the years.

Lastly, I would like to thank my co-workers whom I am proud to call my friends. As colleagues you have been a source of knowledge from which I have learned a great deal; more importantly, as friends we have shared experiences and created memories which I will never forget. Thank you for transforming the work days into memorable days.

TABLE OF CONTENTS

page

LIST OF TABLES

LIST OF FIGURES

LIST OF OBJECTS

LIST OF ABBREVIATIONS

AA	Application Analyst
AFRL	Air Force Research Laboratory
AGV	Autonomous Ground Vehicle
AIS	Application Interface Specification
APF	Adaptive Planning Framework
ARL	Army Research Laboratory
AS-4	Aerospace Standard Unmanned Systems committee
ATT	Advanced Teleoperator Technology
BS	Behavior Specialist
CA	Communicator Analyst
CAN	Controller Area Network
CIMAR	Center for Intelligent Machines and Robotics
CRMF	Cognitive Resource Management Framework
DARPA	Defense Advanced Research Projects Agency
DB	Decision Broker
DBMS	Data Base Management System
DOBPS	Distributed Object-based Programming Systems
DoD	Department of Defense
DoE	Department of Energy
DSA	Diagnostician/Systemizer Analyst
DSS	Decision Support System
DUROC	Dynamically Updated Resource Online Co-allocator
ES	Expert Systems
GARA	Globus Architecture for Reservation and Allocation

GIS	Geographical Information System
GPS	Global Positioning System
GRAM	Globus Resource Management Architecture
HPI	Hardware Platform Interface
IMU	Inertial Measurement Unit
JAUGS	Joint Architecture for Unmanned Ground Systems
JAUS	Joint Architecture for Unmanned Systems
JAUS RA	Joint Architecture for Unmanned Systems Reference Architecture
JDL	Job Description Language
JSIDL	JAUS Service Interface Definition Language
JSS	JAUS Service Set
LLA	Latitude Longitude Altitude
MBMS	Model Based Management System
MDARS	Mobile Detection Assessment and Response System
MDS	Metacomputing Directory Service
MS	Management Science
NIST	National Institute of Standards and Technology
NWS	Network Weather Service
OCU	Operator Control Unit
OGSA	Open Grid Services Architecture
OR	Operations Research
QoS	Quality of Service
PE	Plant Engineer
R2C2	Robotic Range Clearance Competition
RA	Resource Analyst

RAA	Resource Appraiser Analyst
RDL	Resource Description Language
REDCAR	Remote Detection Challenge and Response
REST	Relax and Enrich Strategy
RN	Roadway Navigation behavior
RSTA	Reconnaissance, Surveillance and Target Acquisition
SAE	Society of Automotive Engineers
SAS	Situation Assessment Specialist
SDOF	Single Degree of Freedom
SLAM	Simultaneous Localization and Mapping
SSC	JAUS Subsystem Commander Component
STANAG	Standard NATO Agreement
UAV	Unmanned Aerial Vehicle
UDP	User Datagram Protocol
UGV	Unmanned ground vehicle
UIMS	User-Interface Management System
WMKS	World Model Knowledge Store
WMS	Workload Management Systems

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

A COGNITIVE RESOURCE MANAGEMENT FRAMEWORK FOR AUTONOMOUS GROUND VEHICLE SENSING

By

Gregory A. Garcia

May 2010

Chair: Carl D. Crane, III
Major: Mechanical Engineering

The primary contribution of this research is the design, implementation, and evaluation of a Cognitive Resource Management Framework for Autonomous Ground Vehicle Sensing; a novel and formalized approach for modeling and managing a distributed collection of heterogeneous resources to provide new levels of autonomous capabilities management, currently unaddressed by unmanned vehicle systems standards. The thesis behind this research is that a well-formulated framework providing real-time modeling, appraisal, and brokering functionalities utilizing abstract representations of mission, jobs, and capabilities provides new levels of intelligent resource utilization to the autonomous ground vehicle community.

This research was performed using the resources of the University of Florida's Center for Intelligent Machines and Robotics. This environment provided a real world setting which enabled the creation of a resource management framework designed to address the challenges of resource utilization on high-capability autonomous ground vehicle systems. The experimental software and test plans, developed to validate the thesis of this work, would ultimately culminate into the framework Reference Implementation.

The Cognitive Resource Management Framework has been shown to be both a viable method for modeling and managing a distributed collection of heterogeneous resources and

resource-requests and a valuable contribution to the researchers whose algorithms depend on the reliable availability of resources to fulfill mission operating requirements. The framework's viability and value were demonstrated in both the Reference Implementation and during CIMAR's Environmental Mapping and Change Detection technology demonstration for the Air Force Research Laboratory.

This research presents a new approach to intelligent resource management onboard autonomous ground vehicles. The framework formalizes the discovery, modeling, monitoring and configuration of AGV resources. This work provides new levels of autonomy not currently supported by predominant unmanned systems technology standards. This research provides the foundation upon which new sensing, planning, and scheduling methodologies pertaining to intelligent resource utilization can be implemented and evaluated with the hope of promoting the advancement of highly autonomous platforms capable of sophisticated interactions with unknown, dynamic environments.

CHAPTER 1
INTRODUCTION

Robotics is an interdisciplinary field of study concerning the integration of mechanics, electronics, and software engineering expertise. Robots have continuously proven their utility as a partner in the workforce since the arrival of Unimate [1] on the assembly line at General Motors in 1961. Today a new generation of robots exists with capabilities so advanced they can assume many of the roles and responsibilities once entrusted to their human counterparts. This evolution in robotics is the direct result of continuing research advancements among its constituent disciplines. The roboticist is charged with applying approaches from these research fields to realize a solution to real world problem domains. The Unmanned Ground Vehicle (UGV) arena has and, continues to be, an active area of research within the robotics community.

Background

Any piece of mechanized equipment that traverses the ground and is capable of carrying or transporting some non-human cargo is, by definition, an unmanned ground vehicle. UGVs can be further classified based on the method of operation. In teleoperated vehicle systems, navigational guidance is transmitted to the vehicle via an externally situated human operator using perception information relayed from the UGV [2]. Autonomous vehicles determine their own course using onboard sensor and processing resources and can vary in size, method of locomotion, onboard capabilities, and levels of autonomy. The research presented here addresses high capability, highly autonomous ground vehicles.

Autonomous Ground Vehicle (AGV) research evolved from the discipline of artificial intelligence with the development of *Shakey*, in the 1960s, at Stanford Research Institute [3]. The wheeled platform, designed for indoor use, relied on an actuated TV camera and an ultrasonic range finder to perform navigation and exploration tasks. While *Shakey* never

achieved autonomous operation, research efforts proliferated during the 1970s, 1980s, and 1990s on the *Stanford Cart*, *CMU Rover*, *Navlab,* and DARPA-funded *ALV* projects to name a few. More recently, DARPA (Defense Advanced Research Projects Agency) funded the 2004 and 2005 Grand Challenges, followed in 2007 by the Urban Challenge. The Urban Challenge is reviewed in detail in Appendix A.

The last forty years of autonomous ground vehicle research were, in large part, devoted to addressing problems of path planning and navigation. A restatement of this problem is as follows: Given some goal to achieve and an optional set of operating bounds, a robot must localize itself within the world, sense and model the surroundings, construct a plan through the representation, and execute that plan, until the goal is achieved. A brief synopsis of the aforementioned problem decomposition areas follows.

Localization refers to the position of the robot with respect to some reference coordinate system. The choice of reference system is usually dictated by the world modeling representation chosen for the problem area to be addressed. A small scale mobile robot, designed to explore and map a building interior, might start off in a known location and use wheel encoders for dead reckoning to model the building relative to the *a priori* defined origin; whereas an autonomous resupply vehicle designed to travel hundreds of miles across rugged terrain, would likely use a geospatial world model whose coordinate system is globally referenced using latitude, longitude, and altitude (LLA). The research and discussion contained within the remainder of this document is discussed within the context of large scale outdoor AGVs, although it is directly applicable to all AGV platforms.

A common solution for georeferenced localization fuses global positioning data from satellite constellations (GPS), with precise inertial measurement unit (IMU) feedback to derive a

more accurate estimation position [4]. Simultaneous localization and mapping (SLAM) [5], is an alternative localization approach requiring active environmental sensing. A robot uses sensor information to incrementally build a model of its environment while synchronously determining its position on the map. This approach requires a high level of onboard cognitive ability both in terms of sensor resources and perception/processing capability. Ground vehicle sensor packages are composed of a heterogeneous mixture of resources, depending on the application. Typical sensor suites used for autonomous navigation consist of a compilation of monocular and stereo vision camera systems along with laser and radar based measurement systems, to name a few. This raw data is then processed and fused into a usable representation and stored within the system model. The successful extrapolation of this information is vital when generating viable motion plans.

The final elements of the navigation problem involve path planning and execution. The planner is tasked with finding a route that takes the vehicle from its current state to the goal state. Plan generation is a popular research area in the field of artificial intelligence due to the myriad candidate paths available. Common planning approaches involve some form of informed search through candidate paths where time, distance, or some other heuristic function is being optimized. Online path planning often occurs at various levels in an effort to manage plan optimality while maintaining real-time reactivity [6]. Reactivity in unmanned ground vehicles is critical in surmounting any discrepancies that may arise within the perceived local world. This reactivity is typically achieved via the local path planner which takes a high level goal and creates a series of intermediate sub goals that successfully avoid obstacles not previously accounted for during the higher plan generation stages [7]. In many instances the local path planner is also responsible for generating the appropriate actuation commands for steering,

throttle, and brake to realize the traversal of the planned path. This cycle of locate, sense, plan, and act continues in an iterative fashion until the AGV has reached the goal state.

On July 30, 2002 the DARPA Grand Challenge was announced to demonstrate the state-of-the-art in unmanned ground vehicle technology. The task, to design and create a vehicle that could successfully navigate 150 miles of dessert roads while detecting and avoiding static obstacles that might be present along the given path. The results of the first Grand Challenge, held in 2004, were telling. After more than 30 years of research in the field, only 7 teams possessed competition worthy vehicles. The most successful implementation only traveled 7.6 miles of the 150 mile course.

In 2005 DARPA doubled the prize money and held the race again. This time there were 23 finalist vehicles, 5 of which finished the entire course. The robotics community proved that AGV technology was mature enough to handle extended fully autonomous navigation along a roadway while successfully negotiating static obstacles. Subsequent demonstrations, including the 2007 DARPA Urban Challenge, evaluated the maturity of autonomous navigation capabilities with respect to planning in a highly dynamic environment. The field of dynamic path planning continues to be an active area of research within the AGV community.

These advances in autonomous navigation and related technologies have opened up new areas of research for mobile robotics application. A number of applications require machines that can move to a desired area and then perform a task. The Reconnaissance, Surveillance and Target Acquisition (RSTA) application domain has long drawn the attention of researchers. Started in the 1980s, the Advanced Teleoperator Technology (ATT) *TeleOperated Dune Buggy* demonstrated that humans, via multi-sensory human/machine interfaces, could remotely operate a vehicle and its weapons system at high speed [8]. In the 1990s, a joint effort between the

Army-Navy known as the Mobile Detection Assessment and Response System (MDARS) was developed to provide automated intrusion detection for DoD warehouses and storage sites [9]. The program continues to date and its success has spawned and aided in the development of other similar programs. The maturation of autonomous navigation capabilities has opened the door for automation of RSTA tasks and the development of systems which can effectively perform both operations simultaneously.

Focus

The design of an AGV platform/chassis is another area of engineering ingenuity. Drive-by-wire automated platforms are available in an assortment of configurations based on desired size, weight, and drive train. The selection process and underlying system design is highly constrained by a number of factors including but not limited to cost, operating environment, and mission objectives. An exhaustive study of design criteria is beyond the scope of this work but it is important to characterize how the composition of the platform constrains overall system capability.

Finite bandwidth is the principle factor constraining capabilities development. Bandwidth traditionally refers to the transmission capacity of a communications system, this work posits a less restrictive definition referring to the bandwidth capacity of the platform as a collective. Platform selection restricts payload capacity. Payload capacity and operating environment will constrain the size and type of power system. The bandwidth of said power system not only limits runtime, but further restricts the number of and type of sensors, effectors, and processing nodes. The location of hard points around the platform constrains the placement of effectors, sensors, and mission payloads. Engineers have found workable configurations despite these bandwidth constraints, but it is important to note that their approaches are platform and hardware specific as no best practices or platform design tools or standards presently exist.

Autonomous capabilities, like emergent behaviors, are the visible end product resulting from the interactions among multitier intelligent operations. The development of hierarchical control architectures, consisting of layers of specialized cooperative agents, has been a leading area of robotics research for over 30 years and is discussed in Chapter 2. Intelligent operations involve perception, reasoning, learning, communicating, and acting within complex environments. Perception entails the extrapolation of information, or knowledge, from raw data originating from a sensor. During reasoning, information is inferred about the present system state based on knowledge acquired during perception and that which is known a priori. Communication is vital to the dissemination of knowledge throughout the various system components to facilitate the development of intelligent behaviors. A plethora of approaches to machine learning exist, at its most primitive level, learning is the ability to acquire new knowledge and skills and use them in a manner that improves performance. The action period is the culmination of all preceding transactions, where after careful deliberation a control output is realized by an effector.

The proliferation of autonomous ground vehicle capabilities is often achieved via the application and integration of technologies developed during independent parallel research efforts. The roboticist/systems engineer is tasked with incorporating these disparate technologies onto a common platform to provide some capability that addresses a particular need. This task can be quite daunting as new integration challenges manifest themselves at the hardware and software levels. The focus of this research addresses capabilities integration issues that arise during design and run time pertaining to the utilization of sensing resources to achieve intelligent behaviors that augment autonomous ground vehicle capabilities.

Problem Statement

The preceding discussion described an intricate tapestry of resources and intelligent agents that work harmoniously to afford an unmanned system certain autonomous capabilities. Technological advances in machine intelligence, computing resources, sensing hardware, and perception algorithms have lead to the development of highly sophisticated AGVs capable of navigating long distances through highly dynamic environments. This breakthrough paves the way for the development of highly autonomous systems like MDARS, envisioned decades ago. As a result, mission plans require a higher level of participation/interaction from the ground vehicles and their operating environments often necessitating the need for a number of concurrent processes. To date, research efforts towards capabilities development have taken a "black box" approach, with a focus on addressing one unique problem area without considering the downstream loading effects on the system. This methodology increases integration cost by failing to account for the AGV bandwidth constraints. At the hardware level, existing autonomy packages for navigation have to be redeployed and supplemented where available with necessary RSTA hardware. World models and feature extraction algorithms once tuned for navigation must be retrofit so support the extrapolation and storage of new knowledge. Best effort policies and implicit resource scheduling can lead to resource starvation resulting in deadlock or livelock situations. The area of resource management within the context of the autonomous ground vehicle domain is a new area study in need of attention.

Autonomous ground vehicles are entrusted with performing highly complex tasks that necessitate the concurrent coordination of multiple system resources and increased capabilities management is needed to meet these requirements in an effective and efficient manner. The Cognitive Resource Management Framework (CRMF), discussed herein, provides a mechanism for the management and utilization of resources to sustain existing capabilities while facilitating

the development and integration of new ones. This research addresses sensor resource management onboard autonomous ground vehicles but is directly applicable and extensible to other unmanned systems operating domains.

Motivation

This research supports the development of advanced autonomous capabilities by contributing to ongoing mobile robotics research endeavors by the Center for Intelligent Machines and Robotics (CIMAR) at the University of Florida and the Air Force Research Laboratory (AFRL) at Tyndall Air Force Base. Since the 1980s, CIMAR has been at the forefront of mobile robotics research developing robotic systems for both nuclear industry and defense applications under the Department of Energy and Department of Defense research programs with the common goal of minimizing the human presence required in high risk situations. Together CIMAR and AFRL have collaborated to develop automated solutions for rapid runway repair, explosive ordinance disposal, and perimeter security for military installations. Intelligent decision-making, capabilities integration, and resource utilization have been critical to the success of these programs.

The early success of the Mobile Detection Assessment and Response Systems (MDARS) [10] program spawned the Remote Detection Challenge and Response (REDCAR) project to expand on their original concept. The Remote REDCAR project is an Air Force Battle Lab concept evaluation initiative, motivated by the Air Force Integrated Base Defense requirement, to provide a robotic element to existing base defense and security. The REDCAR concept involves three different robotic platforms designed to meet requirements of mobile long range detection and assessment, agile lethal and non-lethal challenge and response, and small scale search in areas not accessible by other larger platforms. The large scale Landtamer platform integrates autonomous navigation technologies such as motion planning and environmental

sensing and modeling components developed by CIMAR with a proprietary Operator Control Unit (OCU) to achieve platform control.

Viable platform control solutions must provide robotic control at various levels of autonomy. The Landtamer navigation system supports two levels of control: fully autonomous, and a hybrid of teleoperation and autonomous approaches. TeleOp-assist navigation mode allows a remote operator to direct the platform, but ultimately gives the vehicle sufficient authority to avoid collisions caused by the operator's chosen path. Collaborative control is achieved by mapping the operator's joystick inputs as instantaneous waypoints to the motion planner; the planner, using sensor information stored in the world model, decides the appropriate actuator outputs that direct the vehicle along the operator specified path. This example of intelligent machine and operator collaboration illustrates an important paradigm shift in robotics research which this research addresses via a centralized control strategy.

Despite the success and maturity of fully autonomous navigation technologies, program managers require deployed platforms to support specific level of human-in-the-loop interaction known as supervisory control. Fielded platforms must possess the capabilities and intelligence required for fully autonomous operation, but must also provide mechanisms for operator intervention. As the previous example illustrated, there are a number of benefits to this approach. Supervisory control reduces the user's cognitive load allowing an operator to supervise multiple vehicles or activities from one command station with greater effectiveness. The level of autonomy granted to the vehicle allows for optimal onboard decision-making through local state monitoring and reasoning mechanisms. This abstraction of the operator's intent enables the system to intelligently select the optimal combination of effector actions among all available

candidate solution sets. This approach results in a more robust system capable of appropriate behavior adaptations in response to the dynamics of the mission and operating environment.

Recent autonomous vehicle research at CIMAR has emphasized developing advanced navigation, sensing, and modeling capabilities for large scale vehicle platforms. Its most recent platform, the Urban NaviGATOR seen in Figure 1-1, was designed to compete in the 2007 DARPA Urban Challenge. The Urban Challenge, as the name aptly indicates, presented roboticists with numerous technical challenges at both the hardware and software levels. The task necessitated a vehicle capable of driving through an urban environment while safely interacting with other moving vehicles. While the list of behaviors is quite extensive, some key tasks included: driving within a lane, obeying speed limits, maintaining safe following distance, passing slow moving or stopped vehicles, merging into traffic, negotiating an intersection, navigating an obstacle field, detecting a road blockage, and performing any subsequent course corrections [11]. These requirements demanded a vehicle with perception, reasoning, modeling capabilities, and response times akin to those of a human driver.

The platform, built off a 2006 Toyota Highlander Hybrid sport utility vehicle, underwent extensive modification to provide the capabilities mandated by the challenge. The chassis supported drive by wire throttle and braking capabilities from the factory, although a proprietary interface board was constructed for commanding throttle and brake efforts. Motors were added to automate steering and gear shifting operations. A power distribution system was designed and integrated with the existing hybrid vehicle power system to satisfy all motor, sensor, and processing system needs. The sensor network, designed to meet the perception requirements of the challenge, was composed of four fixed laser range scanners, four single degree of freedom actuated laser range finders, four cameras, three GPS receivers, wheel encoders, and an inertial

navigation system. A custom built computer rack populated with twelve dual-core computing nodes was added to handle the processing, modeling and reasoning requirements of the challenge. After eight months of design and fabrication, thousands of man hours, and a cost of approximately two-hundred-fifty-thousand dollars in hardware alone, the Urban Navigator was complete.

The Urban Challenge, along with collaborative research efforts with AFRL, illustrates the critical need for the integrative resource management framework described herein. The Urban Navigator now functions as a test-bed for autonomous capability development. Researchers at the Center for Intelligent Machines and Robotics continue to supplement existing navigation and localization capabilities. Some areas of study include enhanced localization within a lane using vision and laser range finders, laser and vision based object classification, extrapolation and storage of ground plane representations from laser topography data, and advanced environmental mapping and monitoring.

The evolution of capabilities places new requirements on the utilization of the sensor resources. Platform bandwidth constraints limit the quantity and type of physical sensors that a platform can handle. From a cost and time feasibility standpoint, software architectures and platforms must be modular enough to support the addition and removal of sensors and capabilities to extend the life of the platform. Systems integration costs should be minimized while providing a mechanism by which to achieve robust performance. Given the limited availability of resources onboard vehicle platforms and the growing number of capabilities that consume them, it is inevitable that resource conflicts will ensue. Furthermore, system capabilities development should continue with a dual focus on full autonomy and supervised autonomy. The

purpose of this work is to address these issues through the development of a lightweight sensor resource management framework for autonomous ground vehicle systems.

Statement of Purpose

This dissertation documents the design, implementation, and evaluation of a Cognitive Resource Management Framework for Autonomous Ground Vehicle Sensing; a formalized approach for modeling and managing a distributed collection of heterogeneous resources to provide new levels of autonomous capabilities management, currently unaddressed by unmanned vehicle systems standards. The thesis behind this research is that a well-formulated framework providing real-time modeling, appraisal, and brokering functionalities utilizing abstract representations of mission, jobs, and capabilities provides new levels of intelligent resource utilization to the autonomous ground vehicle community.

Research Solution

A Cognitive Resource Management Framework (CRMF) was designed to address the challenges of resource utilization on high-capability autonomous ground vehicle systems operating in dynamic environments. These complications manifest themselves in all phases of the AGV life cycle, from design and integration through testing and deployment.

The framework conceptualization evolved through the careful analysis of resource management approaches across numerous academic disciplines, a detailed review of the literature is presented in Chapter 2. At its core, the framework embodies a Plant Engineering motif, evidenced by the Plant Engineer, the centralized resource broker. The framework espouses to a hybrid of Gupta's *Give them information to decide* and *Let the computers tell us* scheduling theory paradigms [12]; asserting that a well informed manager, using analytical information derived through artificial intelligence methods, will make optimal or near optimal decisions through the application of managerial and technical experience and judgment. The Cognitive

Resource Management Framework is now summarized; a detailed discussion is contained in Chapter 3.

To facilitate implementation and integration with existing vehicle systems and standards, the framework defines a Plant Engineer element along with a distributed collection of cooperative "analysts" which are utilized to construct a representative model of an autonomous vehicle's resource capabilities:

- Plant Engineer (PE)─ a single centralized authority given sole discretion over assigning value to incoming jobs, and to allocate and provision the resources necessary to fulfill the job based on all acquired analyst knowledge
- Diagnostician/Systemizer Analyst (DSA)─ charged with creating and maintaining a real-time representation of system state information, including resource capabilities and configuration
- Communicator Analyst (CA)─ plug-in interface acting as gateway between the CRMF and vehicle control and planning architectures for importing vehicle mission state information
- Resource Appraiser Analyst (RAA)─ oversees performance management through the collection of framework data for real-time or offline processing
- Resource Analysts (RA)─ each devoted to a particular resource, relays abstract resource attribute representation to DSA for discovery and monitoring
- Application Analysts (AA)─ each devoted to a particular software application requiring the utilization of a resource, converts application specific resource need into generalized framework Job Request for submission to the Plant Engineer

Together, these framework elements provide mechanisms for the discovery, monitoring, and modeling of resources; guidelines for the submission of job requests; and strategies for resource allocation and scheduling. In addition, the framework defines generalized abstract representations for application resource needs, termed Job Requests, and an extensible class structure of Resource Objects. The concept of operation is as follows. RAs and AAs comprise the lowest level of the framework hierarchy; they provide the intermediate level analysts with distilled resource and job representations. At the intermediate level, the CA, RAA, and DSA

compile lower-level distilled knowledge into higher level system state information. The data representation used at the intermediate level can vary depending on the brokering algorithms implemented at the PE level. While executing its brokering duties, the Plant Engineer transforms the intermediate level analysts assimilated knowledge into executed management policy.

Figure 1-1 The Urban NaviGATOR: CIMAR's Urban Challenge Platform

CHAPTER TWO
REVIEW OF LITERATURE

While the topic of resource management within the context of autonomous ground vehicles is an emerging field for researchers, the problem is well studied by other application domains. A multi-discipline review of the literature detailing the intricacies of and approaches to the resource management problem was conducted to provide a baseline for comparing the work produced here. Furthermore, competing robotics architectures were reviewed to ensure interoperability with emerging standards and how the Cognitive Resource Management Framework extends capabilities not yet addressed by standards bodies.

Standards Compatibility

Standards bodies are set up to stem the proliferation stove-pipe unmanned vehicle systems, to promote interoperability, and to ease integration between multi-vendor systems. The following sections describe four predominant unmanned vehicle systems standards, an agricultural vehicle communications standard, along with a framework developed at the Center for Intelligent Machines and Robotics. An analysis of resource management and compatibility with the produced work is included.

JAUS Reference Architecture

The Joint Architecture for Unmanned Systems (JAUS) Reference Architecture is a DoD-mandated messaging standard among the elements that comprise an unmanned system and was intended to enhance modularity, reduce system costs, accelerate development, facilitate code reuse, and provide a framework for technology insertion. While the JAUS working group is now de-chartered and work has transitioned to the Society of Automotive Engineering (SAE) AS-4, there continues to exist a number of military programs [13], platform providers [14], and open

source development [15] using the most current JAUS reference architecture (RA) necessitating the continued support and evolution of this standard.

Standard development predates 1998 when the working group drafted RA Version 0.2 under the name Joint Architecture for Unmanned Ground Systems (JAUGS), after which the scope was expanded to include all unmanned systems. The final RA, now at Version 3.3 [16] defines a collection of reusable components and their interfaces that are independent of technology, computer hardware, operator use, vehicle platforms, and isolated from mission.

The standard contains a predefined set of components subdivided into categories pertaining to:

- Command and Control
- Communications
- Platform
- Manipulator
- Environmental Sensing

These components, once integrated, provide an assortment of capabilities that form an unmanned system. Furthermore, JAUS specifies the necessary accompanying message vocabulary, protocol and sequencing, and transport guidelines for passing JAUS messages over a particular medium, such as UDP, TCP, or Serial.

At the command and control level, JAUS employs a centralized uninterruptable control model. Access to a primitive driver or manipulator is granted to one component at a time by the Subsystem Commander. While the messaging architecture provides a framework for the request and transfer of control, conflict resolution within the existing model is achieved via predefined component priority levels designated through an Authority Code.

The Cognitive Resource Management Framework is intended to supplement existing resource allocation mechanisms defined by the JAUS architecture with an additional layer of

intelligent decision-making and modeling capabilities. The Reference Implementation maintains backwards compatibility with existing systems through the reuse RA v3.3 component control and manipulator control message transfer protocols. Additional functionality is provided and permitted through the use of "User-defined Components" and "Experimental Messages." It is important to note that the framework does not prescribe a JAUS compliant platform, or any particular architecture; its abstract and modular design enables its implementation within the architecture of choice.

SAE AS-4

In 2005, the JAUS began a 4 year transition to the Society of Automotive Engineers Aerospace Standard 4 (SAE AS-4) Unmanned Systems committee. AS-4 represents a transition from a component and message based architecture towards a service oriented architecture. The standard defines a XML based JAUS Service Interface Definition Language (JSIDL) [17] through which service interfaces are specified, along with an extensible list of services within the domain of unmanned systems known as the JAUS Service Set (JSS) [18].

The standards community is actively developing standards documents to address the growing complexity of missions and system design. Subcommittee task groups are currently drafting standards documents for environmental sensing and modeling and manipulators [19]. These service sets propose the message sets and interfaces for discovery, and control of manipulator and sensor resources along with the management capability to manipulate a list of commands to be executed by the resource. While the proposed manipulators standard provides abundant messaging for feedback control, additional resource specific capabilities such as health and task monitoring that support advanced monitoring and modeling are beyond the scope of current efforts.

At present, only the JSS Core Service standard has been published. The services form a foundation upon which other domain specific services can be developed. As published, JSS Core services include access control, events, management, discovery, time, and liveness. These services are merely an evolution of the JAUS RA 3.3 towards service oriented architecture and as such, still employ the same authority level based, exclusive control model. Like the JAUS RA, AS-4 provides a discovery mechanism and message sets which can be utilized by the Cognitive Resource Management Framework. Furthermore, the standard allows for the development of experimental services, such as a resource management service, to provide advanced autonomous resource provisioning and allocation capabilities which may be a candidate for adoption into the standard.

STANAG 4586

Standard NATO Agreement (STANAG) 4586 [20] provides definitions of architecture and messages required for interoperability of unmanned aerial vehicles (UAVs) in the NATO Combined/Joint Services Operational Environment. This standard enables an operator to possess a defined level of control over a UAV. While the standard has been mandated by the United States Congress [21] as a requirement for a number of systems, other nations have expressed reservations regarding its applicability to future UAVs[22].

The last few years has seen evolutionary advances in unmanned systems autonomy thanks to continued research and development successes among various engineering disciplines. Several groups have attempted to define metrics for describing platform autonomy. Figure 2-1 depicts the NIST's approach to specifying a vehicle's level of autonomy based on the 3 axes of Environmental Difficulty, Human Independence, and Mission Complexity. Platts asserts that a vehicle's autonomy is characterized in terms of 4 dimensions: Context of vehicle operation, Level of Human Interaction with the system, System Knowledge acquisition, and System

Reasoning Capability [23]. Although their terminology differs slightly, it is evident that as higher autonomy is sought, the expectations regarding system capabilities increases and the role of the human transitions from that of platform operator towards capabilities supervisor.

The aforementioned standards were not developed with these imperatives in mind. They were intended to promote interoperability by supporting unmanned vehicle teleoperation and autonomous navigation through route planning by waypoint selection. To accommodate today's state-of-the-art autonomous capabilities, the standards require a paradigm shift from a time multiplex approach, where an operator explicitly addresses the needs of a particular resource in turn, towards a higher level of command abstraction, where the operator identifies a series of capabilities and goals, and the system decides which resources are best suited to meet the requirements.

Commanding at the capability level requires that allocation of tasks be mainly carried out by the system, rather than the operator. What remains to be developed is the "transfer function" to map an input of a capability requested, to the desired behaviors that each resource will execute to achieve it. This function will need to understand concepts of operation, such as mission goals, to select which resources can most appropriately provide the capabilities requested. These issues are at the forefront of the research presented here. Given the short evolutionary cycle of technological advances in the field of UGVs, it is important to design frameworks that augment the capabilities of current standards, rather than attempt to replace or rewrite them altogether.

NIST 4D/RCS

In the late 1990s NIST, in collaboration with the Army Research Lab (ARL), developed the 4-dimensional Real-Time Control System (4D/RCS) reference model architecture [6] as an extension of their existing Real-Time Control System. Although originally developed for the manufacturing industry, it is now used for intelligent control of unmanned vehicle systems.

4D/RCS provides a multi-layered, multi-resolutional hierarchy of computing nodes each possessing elements of sensory processing, world modeling, value judgment, behavior generation, and a knowledge database as illustrated in Figure 2-2. Higher levels within the hierarchy enable goal defining deliberative behaviors, while at lower echelons, the nodes generate goal-seeking reactive behaviors. The architecture prescribes both horizontal and/or vertical communication pathways among elements within nodes and hierarchy levels. Each node within a hierarchy is responsible for sensory processing, world modeling and knowledge database maintenance at corresponding resolution and map size while computing behavior generation plans for the specified time slice. As a direct result of its layered processing and reasoning approach, 4D/RCS requires provisions for extensive symbolic data structures resulting in high network traffic, redundant computations, and high system integration costs. In Figure 2-2, for example, nodes at the primitive level are responsible for processing, modeling and storing information within 5 meters at a resolution of 5 centimeters while generating behavior plans valid for .5 seconds. The temporal layering of planning and execution operations within the hierarchy illustrates the criticality of an unmanned system's reactivity.

Mission Planning, Behavior Generation, Value Judgment, and World Model responsibilities within a single 4D/RCS node are analogous to the roles of Plant Engineer and Appraisal, Systemizer/Diagnostician analysts in the presented work providing immediate opportunities for integration at 4D/RCS VEHICLE echelon. The roles of resource analyst would be integrated into the SERVO layer while the application analysts roles would be distributed among the Behavior Generation components throughout the architecture. Furthermore, the 4D/RCS methodology of task decomposition is akin to task abstractions of the produced framework.

While 4D/RCS was demonstrated at DEMO III, only the first three echelons in the locomotion subsystem were implemented [24]. Complications arising from sensor resource scheduling conflicts, collocation issues, or hardware failures were not addressed.

ISO 11783

Initiated by manufacturers in the early 1990s, ISO 11783 is a Controller Area Network (CAN) standard that governs electronic control for electrical systems in agricultural, forestry and construction equipment over a serial network. Today it is commonly referred to as ISOBUS. A stated goal of ISOBUS was to reduce the growing number of "black boxes" in the tractor cab. These proprietary control boxes not only impair the operator's visibility while inside the cab but also increase the user's cognitive load, which can lead to distractions that result in job-site accidents. Figure 2-3 provides a visualization of how ISOBUS can link common agricultural equipment multi-vendor components. ISOBUS promotes greater compatibility among multi-vendor agricultural products, interchangeability between sensors and controllers, and improved diagnostics and performance records.

The rationale behind ISOBUS closely parallels that of the Cognitive Resource Management Framework thereby reinforcing its need for development and it operational significance. Perhaps of greater importance are the notable differences in the scope and applicability of these two bodies of work. ISOBUS places emphasis on the interconnectivity of tractors and implements at the hardware and software level, the associated high implementation costs virtually eliminates its portability to other platform types with incompatible hardware. The ISOBUS open network model relies heavily on CAN-bus to achieve communication between elements. A number of sensors and actuators deployed on today's unmanned ground vehicles do not offer CAN-bus interfaces. The CRMF employs a more relaxed approach; it does not prescribe a transport protocol and outlines a Communicator Analyst with plug-in type

functionally to achieve similar open network communication among other AGV architectures and standards.

The scope of ISO 11783 will address the tasking and sequencing of operations, however, much of this functionality is not yet standardized. Furthermore, ISOBUS is not intended for autonomous operation in dynamic environments and relies heavily on human interaction for preplanning of activities. Unlike the CRMF, which autonomously schedules tasks at runtime using onboard artificial intelligence and vehicle self-awareness, the ISOBUS task controller requires fixed offline task planning using a desktop application and executes one mission at a time. This approach of scripting tasks and resources diminishes overall robustness of a system, a significant contribution of the Cognitive Resource Management Framework over existing standards. It should be noted that ISO 11783 does allow for the automation of functions involving sequence control, but is restricted to "record" and "playback" functionality where operator inputs commands are recorded, save and later played back (see Figure 2-4).

APF

Researchers at the Center for Intelligent Machines and Robotics proposed an Adaptive Planning Framework for Situation Assessment and Decision-Making on an Autonomous Ground Vehicle [25] to address the issues with autonomous operation in complex, unstructured environments. The framework manages a collection of virtual Situation Assessment Specialists, Behavior Specialists, and a Decision Broker. The specialists render findings, known as metadata, which can be used by other specialists for decision-making. A conceptualization of this model is depicted in Figure 2-5.

The knowledge representation scheme identifies three domains of specialists, functioning in a cooperative capacity, which oversee the propagation of facts into findings into recommendations and finally into actionable behaviors. Situation Assessment Specialists (SAS),

continually update metadata findings regarding a set of previously identified conditions, states, and events that are of importance to other specialists. The Behavior Specialist (BS) monitors findings from the SAS and renders a recommendation regarding the suitability of its respective behavior for controlling the vehicle. The Decision Specialist, or Decision Broker (DB), must weigh the findings and recommendations of all SAS and BS and select the most appropriate vehicle operating behavior.

The stated goal of the framework is to use the elements of the Knowledge Representation Scheme derived during the design phase to produce actionable, high-level decisions at run-time. In an effort to standardize the process, a collection of knowledge engineering tools were developed for defining behaviors, protocols, and findings. The Behaviors Use Template, see Figure 2-6, is used to examine the actions and contingency actions associated with any given behavior transitions while specifying the specialist findings that are used as inputs. By specifying required inputs, the framework assumes that the SAS has full utilization of resources necessary to render findings, an assumption that is not always valid. Accordingly, a Findings Worksheet, see Figure 2-7, should be populated for each Situation Assessment Specialist. The sheet is intended to describe the rules or algorithms necessary to render a finding. Any sensor dependencies are listed in the "Comments" field, which is intended to facilitate understanding, not for verification or validation purposes. The framework, as written, states that finding worksheets should be reviewed for cohesiveness, completeness, and ambiguity; in other words, a human should review the sheets to ensure that values are used by another specialist (cohesiveness), ensure that every possible value has some means by which to be determined (completeness), and that the values produced are repeatable (no ambiguity). These tools are intended as design time aids to facilitate understanding and are not suitable for model validation, and cannot support dynamic reconfiguration.

Team Gator Nation implemented the Adaptive Planning Framework on its 2007 DARPA Urban Challenge autonomous vehicle, the Urban NaviGATOR. CIMAR's approach decoupled the problem of autonomous navigation into six operating behavior modes shown in Figure 2-8. Each behavior is comprised of one or more sub-behaviors through which it transitions. The combination of behaviors and sub-behaviors act as system triggers for the execution of behavior specific algorithms and protocols that affect the manner in which the vehicle operates. Appendix A specifically addresses certain implementation details, but an in-depth review of the implementation is available to interested parties in the literature [26].

The Cognitive Resource Management Framework is intended to enhance capabilities of the current system. One potential integration model could involve the embedding of APF specialists within the Cognitive Resource Management Framework and the development of a new Loitering behavior. A foreseeable application might include a Situation Assessment Specialist operating within the Resource Appraiser Analyst; if a number of resource requests are denied due to high resource utilization, the specialist may render a finding such as "resources are overtaxed." This information could be consumed by a new Loitering Behavior Specialist which in turn would recommend its application. Ultimately, the Decision Broker would determine the viability of the recommended behavior and choose whether or not to execute.

The Adaptive Planning Framework in many ways was the catalyst for the produced research. The APF, as implemented, abstracts specific deliberate vehicle functionality into high level operating behaviors and sub-behaviors. It empowered the vehicle to autonomously evaluate all possible alternative behaviors and select the most appropriate behavior given the strictly defined mission goal of "finish the race." Likewise, this "derived" work abstracts mission goals along with resource capabilities and requests to autonomously determine the most appropriate

allocation and scheduling of resources to fulfill the mission. In addition, this framework provides mechanisms for conflict resolution, resource monitoring and modeling.

Resource Management in Computer Science

Resource management within computer systems occurs at both the micro level, a single operating system overseeing the utilization of resources on the host computer, or at the macro level, such as mainframe computers or server systems consisting of a collection of similar resources. Traditional resource management approaches within computer science were designed to address a target environment that was largely homogenous, reliable, and secure. Large scale processing activities required the use of highly integrated enterprise computing centers, an approach termed "host-centric." However, the internet and the emergence of e-business have altered the information technology infrastructure to encompass external resources, networks and services, requiring a "network-centric" approach. This continuing decentralization and distribution of software, hardware, and even human resources necessitates the achievement of qualities of service (QoS) among the distributed ensemble of resources.

GRIDS

Grid technologies enable the sharing and coordinated use of a distributed and heterogeneous network of computing resources. In *The Grid: Blueprint for a New Computing Infrastructure* [27], Grid architects Forster and Kesselman provide a thorough discourse on all matters pertaining to Grid computing including areas of application, programming tools, services, and infrastructures. Grids manage a logical collection of dynamic distributed resource sets termed Virtual Organization [28]. The Globus Toolkit, an open source set of software libraries and services that support Grids, is the *de facto* standard for building Grid systems. The Open Grid Services Architecture [29], or OGSA, is an evolution of the Globus Toolkit signaling the convergence of Grid computing and Web Services. OGSA is a service oriented approach that

addresses the creation, maintenance, and application of Virtual Organizations. Over the past decade researchers have attempted to define the problem of resource management within a Grid computing system and derive solutions to address it.

The Globus Resource Management Architecture was designed to provide access to a collection of computers in distributed heterogeneous systems, a scenario closely resembling the composition of an autonomous ground vehicle. The architecture, depicted on the left side of Figure 2-9, consists of 3 principle components: Information Service, Local Resource Managers, and Co-allocation Agents. The Information Service, closely resembling the Diagnostician/Systemizer Analyst described in this work *sans* the spatiotemporal modeling capabilities, characterizes a resource by type, architecture, structure, and state. Discovery and publication methods are automated to ensure information is kept up to date. A query mechanism is used by applications to locate resources. The Co-allocation Agent, using the Information Service, matches the QoS requirements to resource requirements and then directs the allocation request to a local resource manager. This architecture does not address advanced reservation and heterogeneous resource types, key areas of interest for unmanned ground systems sensor networks.

The application of Grids to performance sensitive applications often requires the utilization of multiple resources simultaneously. It should be noted that while simultaneous allocation can be achieved manually, it is prudent that any resource management entity support the capability of coordinating the management of multiple resources. In *Resource Co-Allocation in Computational Grids* the authors propose a layered co-allocation architecture that addresses the allocation, configuration, and monitoring/control function of a resource ensemble [30]. A three level architecture is proposed consisting of a resource management layer, which assumes the use

of Globus Resource Management Architecture (GRAM), a co-allocation mechanism atop a single-resource management component, and Co-allocation agents using the co-allocation mechanisms to implement the application-specific strategies. By allowing application-level guidance of resource selection, as opposed to the broker level approach advocated in this research, sub-optimal operation may ensue due to the application's limited knowledge regarding the overall system state. Two co-allocation mechanisms are described; Atomic Transaction Mechanism and Interactive Transaction Mechanism. Both require explicit specification of required resources beforehand, whereas the Cognitive Resource Management Framework involves the submission of a capability request thereby allowing the system to allocate the best available resources. The Dynamically Updated Resource Online Co-allocator (DUROC), was created to handle the interactive co-allocation requests for processors on a parallel computing system. An evaluation of the system revealed an allocation time of 2 seconds for a single sub-job, most of which stems from latencies associated with GRAM. Not only are such latencies unacceptable given the highly dynamic environments in which autonomous ground vehicle operate, but these strategies do nothing to ensure that a given co-allocation request will succeed.

The absence of a resource reservation mechanism results either in a cost prohibitive system, due to over provisioning, or one which often suffers from degraded service for critical traffic; these phenomena hold true in both the field of ground vehicle robotics and Grid computing. Foster and Kesselman proposed the Globus Architecture for Reservation and Allocation (GARA) to support dynamic discovery, advanced and immediate reservation of heterogeneous resources [31]. GARA, depicted in Figure 2-9, extends the Globus Resource Management Architecture by introducing a generic resource object and a Co-reservation agent. The Co-reservation agent is responsible for discovering a resource set that satisfies the

application of QoS requirements. However, due to the large overhead cost incurred while creating a generic resource object, reservation and allocation are split into distinct operations. The Co-reservation agent merely returns reservation handles that can be passed to a co-allocation agent. The authors question the feasibility and usefulness of generically treating all objects uniformly due to its high overhead. By prescribing an extensible set of resource objects which define specific resource attributes, the framework presented herein attempts to efficiently merge the allocation and reservation processes within a single Plant Engineer component and extend the resource model to include a variety of sensors and actuators whose capabilities qualities are not as trivially qualified as those of networks, disks, processors, and memory that are currently supported by Grid computing.

Recent research efforts have focused on generalizing the management process by defining Agreement semantics. In *Agreement-Based Workload and Resource Management* [32], the authors present an architecture consisting of a resource-independent set of services for discovering, monitoring, and signaling agreements. This approach requires conformity to the Web Services Agreement Specification [33] and Workload Management System (WMS), a set of Grid middleware components which handle task distribution and management. Their proposed architecture, depicted in Figure 2-10, only addresses the problem of single resource reservation and management with a stated goal of minimizing individual task execution time. The WMS employs a bottom-up discovery approach with respect to the layered architecture due to the absence of a system wide modeling component advocated in this work.

A treatise regarding resource management within a Grid is provided in *Agreement-Based Resource Management* [34]. The paper sites some 60 sources capturing the evolution and application of Grid computing solutions. The authors utilize “agreements” to explicitly state the

terms between a resource user and resource provider. The encapsulation of policy terms within an agreement facilitates the abstraction of a resource. However, Czajkowski and Foster acknowledge that agreement notation and protocols will often be domain dependant. Current resource management systems focus on management of compute resources, particularly addressing processing and storage. This research embodies a similar generalized resource management approach geared towards addressing unmanned ground vehicle domain specific requirements.

Resource management systems within Grids are predominantly focused on high-throughput computing, where the goal is to minimize task execution time [35], [36]. These systems assume all tasks are of equal importance; their goal is to realize optimum scheduling and load balancing, without taking into consideration the criticality of a certain task over another. As a result, these approaches require only limited systems model monitoring capabilities. Hao's model, depicted in Figure 2-11, bears a resemblance to the Cognitive Resource Management Framework model with a few key distinctions. The Information Collection Module consists of a grid information management system that collects and issues state information. Unlike the Diagnostician/Systemizer Analyst, MDS only stores static configuration information such as the type and speed of processors, while sufficient for Grid computing application, is unfit for deployable sensor modeling onboard an AGV. Like the NWS elements within Hao's model, the resource analysts form a distributed monitoring system to feedback information to the Appraiser Analyst. However, these framework analysts carry the added responsibility of transmitting task assignments to the resource themselves without the overhead associated with Grid Toolkits.

While Grid Computing architectures have proven their utility with respect to distributed, heterogeneous compute resource management, they are not as well suited to address the domain

specific needs of unmanned ground vehicles. However, their characterization of the resource management problem and their approaches towards the virtualization of resources and task abstraction provide valuable insight that has aided in the design and implementation of this body of work.

Service Availability Forum and Distributed Object Based Programming Systems

While computational Grids receive a great deal of focus within the literature, similar efforts have been attempted and some even developed in parallel. The Service Availability Forum (SA Forum) is a consortium of industry-leading computing and communication companies working towards the development of high-availability and management software interfaces [37]. Hardware Platform Interface (HPI) and Application Interface Specification (AIS) service sets have already been published. The SA Forum approach to standardization is through the development of middleware, see Figure 2-12, targeted specifically for computer systems and networking/telecommunications resources and does not readily support integration with existing autonomous ground vehicle architectures or standards.

Distributed Object-Based Programming Systems (DOBPS) predate the SA Forum and Grid approaches. The authors present a discussion of design and implementation of such systems in [38]. DOPBS are intended to simplify the programming language interface through the use of a generic object which encapsulates state information, data, or a set of operations or procedures to manipulate the data. These systems utilize object abstraction to create a common primitive that reduces the complexity of the human and machine programming. These systems address issues pertaining to object management, object interaction management, and resource management. Again, resource management is limited within the context of memory, secondary storage, and processors. This approach assumes a high-level of coordination among programmers at the software engineering phase and is not intended as an abstract method to

integrate existing systems from multiple vendors. Furthermore, DOBPS object interaction management and system level invocation handling requires the explicit object identification at the time of invocation, rather the utilizing a capabilities based model of objects to autonomously select the most appropriate object for utilization as the produced research does.

Resource Management within Business/ Operations Research

Long before the advent of the computer resource management systems within the preceding discussion, humans bore the responsibilities of formulating and executing business strategies. The academic disciplines of Operations Research (OR) and Management Science (MS) are devoted to the application of analytical methods to study complex problems affecting businesses and industry. Operational situations typically necessitate the processing of jobs on certain facilities, a situation analogous to the distribution of tasks among a collection of sensor resources on board an autonomous ground vehicle. The objective of this section is to analyze various proven approaches within the literature that address the topic of resource management within business systems to develop the strategies and framework components necessary for creating the Cognitive Resource Management Framework for Autonomous Ground Vehicle Sensing.

Scheduling Theory

An exhaustive review of Operations Management literature revealed that the topic of Scheduling was among the top two areas of study spanning the last two decades of the Twentieth Century [39]. Scheduling conflicts often have serious ramifications on an organization's performance. Resource starvation is a multitasking related problem which occurs when a process is perpetually denied access to a resource. Deadlock and livelock are two specific starvation related problems. Deadlocks occur when two or more programs wait for a resource that is occupied by another program in the same set, none of the involved programs are able to change

their internal states due to lack of resource availability. Similarly, livelock occurs when the programs involved are alternating states, but no overall progress is achieved. The "Dining Philosophers Problem" is an illustrative example of these concurrency problems. The developed framework advocates a centralized "waiter" solution rather than utilizing resource hierarchy models or a fully distributed solution proposed by Chandy and Misra [40].

Gupta provides a summary of various scheduling problems arising in manufacturing along with possible approaches to solve them [12]. While he reviews a number of scheduling research paradigms, three are of particular importance and are discussed herein. Existing sensor resource managers [41] employ *Might is right*, implicit strategy. The *Give them information to decide* paradigm, using similar rational as decision support systems, affirms that if enough information is provided to managers for making decisions, they will be able to make optimal or near optimal decisions. As an added benefit, this approach enables the utilization of managerial and technical experience and judgment in the schedule decision-making process. The final paradigm discussed in the literature, *Let the computers tell us*, requires the use of artificial intelligence such as expert systems or neural networks to solve a constraint satisfaction problem. This concept envisions the utilization of learning mechanisms within computer systems to solve practical problems; however, the practice of scheduling has not been much affected by this paradigm.

The Cognitive Resource Management Framework, detailed in Chapter 3, utilizes a hybrid of the *Give them information to decide* and *Let the computers tell us* paradigms wherein the Plant Engineer plays the role of manager, and a team of cooperative intelligent analysts provide the information necessary for establishing scheduling policy. The role of analysts in resource conceptualization is of paramount importance. Managers' mental models affect what they see, and two managers with different models can conceptualize resources differently and also suggest

different relevant resources [42]. This concept of a manager's mental model is defined in strategy literature as "dominant logic" [43]. Bettis and Prahalad suggest that dominant logic acts as an information filter, focusing the efforts on certain information that is deemed relevant. In designing the model representation for the proposed DSA it is important to characterize critical knowledge elements needed by the Plant Engineer for effective decision-making.

Human Oriented Solutions

Plant engineering is a branch of engineering concerned with the installation, operation, maintenance, modification, modernization, and protection of physical facilities and equipment used to produce a product or provide a service [44]. Thusly, the plant engineering function is multidisciplinary in nature, as is robotics. The position requires technical expertise extending beyond engineering disciplines to incorporate business and management skills such as supervision, project management, and contracting; in short, the plant engineer is a specialist in anything. Among the activities a plant engineer oversees are design and operation of systems, the contracting for equipment and materials, administration of the organization and personnel, and coordination of activities with all other functions and departments within the organization. These activities require a high level of experience in applied knowledge.

The preceding illustration of the responsibilities of a plant engineer depict the parallels between the requirements of managing an organization of humans and machine tools with that of managing a collection of resources onboard an autonomous vehicle that form a virtual plant. These descriptions serve as the inspiration for this research. The framework for autonomous ground vehicle sensor resource management was formulated using a plant engineering analogy which has proven successful in real world industrial applications in organizations of all sizes. Figure 2-13 highlights the importance of the plant engineer within the organizational structure of a typical small size plant. To be most effective, the plant engineer should report directly to top

plant or facility management. If implemented using the JAUS RA or SAE AS-4 standards, the plant manager depicted in the figure would likely report to the Subsystem Commander as specified in the published work. In the context of the Adaptive Planning Framework, the plant manager operates below the Decision Broker/Decision Specialist; this plant engineering analogy lends itself to integration within a number of existing robotics standards and architecture further demonstrating its suitability.

The Cognitive Resource Management Framework, utilizes a network of cooperative analysts that aid the Plant Engineer entity in decision-making. The conventional analyst is a man who knows the job, a wizard who is requested if the problem gets out of hand, where this expert acts as a neutral party. Unfortunately, members in industry began doubting the effectiveness of practicing analysts since many proposed solutions were never adopted by management because the analyst failed to appreciate the significance of the situation and the factors that enter into the manager's decision. Experience shows that for a successful OR/MS implementation the analyst must work harmoniously with the manager to evolve a problem solving process that convinces the manager of the utility of the approach [45]. In *The changing role of analysts in effective implementation of operations research and management science* [46], the author identifies nine typical analyst roles that increase the interplay between the analyst and the client system which leads to a research solution better suited towards addressing the dynamics of implementation.

Hildebrandt describes a collection of roles that an analyst must assume, whereas the developed framework is composed of a team of analysts, each performing a specific role. A brief discussion of the roles relevant to the framework discussion in Chapter 3 is now presented. The role of the communicator is to create a dialogue between the organization and the environment. This entails the mutual exchange of information with the goal of acquiring knowledge regarding

the environment and applying it within the organization to provide a suitable solution. An expert role is described where an analyst possessing special insight and expertise in a given subject matter renders its expert opinion to a decision maker. Two other closely related roles are that of systemizer and diagnostician. The diagnostician is responsible for collecting characteristic aspects of the organization, from the experts, for the purposes of diagnosing an organization's ability to fulfill its strategic goals. The systemizer is tasked with finding a description of a complicated system that is both clear and insightful. The paradigm just described, in which the analysts enable others to solve problems more effectively is known as participative model building. This model incorporates a political or social welfare element into the plant engineering decision-making process by characterizing goals based on the collective judgments of the analyst network. This approach allows the framework to seek solutions that optimize some abstract goal condition, rather than traditional approaches which seek to minimize execution time.

Computer Based Solutions and Decision Support Systems

Xia and Wei proposed a generalized resource management framework for adapting business service capabilities at runtime built upon Web Services technology [47]. The framework, depicted in Figure 2-14, utilizes a resource model consisting of a hierarchical collection of resources. Each resource possesses a list of associated capability attributes and roles. Runtime adaptability is achieved by monitoring the status of tasks, and reallocating them among a new set of resources. This approach is suitable if idle or underutilized service sites exist whose roles match the business task but is not as effective or beneficial in an unmanned ground vehicle which, due to bandwidth constraints, does not have access to idle redundant sensor resources. While the authors tout the novelty and utility of a dynamic resource model, the capabilities representation within the model still remains static. In short, the model representation is not suitable for monitoring spatiotemporal data, nor is it capable of supporting graceful

degradation of a resource with capabilities, such as a serial manipulator; two key issues of paramount importance pertaining to autonomous ground vehicle technologies.

Decision Support Systems (DSS) are computer based information systems which enable a decision maker, the user, to interact with the system to a conduct a "What if" analysis based on some given scenario. Traditional DSS systems are composed of three main entities depicted in Figure 2-15, a Data Base Management System (DBMS), a Model Based Management System (MBMS), and a User-Interface Management System (UIMS). The UIMS allows the user to interact with the database to acquire all necessary data via the DBMS. Using the MBMS, the user selects the appropriate model to run the scenario. Decision Support Systems are well suited to analysis and synthesis of complex scenarios in which there are too many aspects to handle.

Utilizing a traditional Decision Support System for scheduling problems poses some technical challenges. Ecker cites a number of applications of decision support systems to scheduling problems and summarizes the problems encountered [48]. He proposes a Problem Base Management System, depicted in Figure 2-15, to address issues arising from problem variety and complexity. The system utilizes constraint satisfaction where hard and soft constraints have to be taken into account. Hard constraints are requirements, whereas soft constraints can be thought of as preferences. The question of whether there exists a schedule that meets all hard and soft constraints is known to be NP-complete [49]. Furthermore, if no feasible solution obeying all constraints exists, heuristic conflict resolution techniques are often applied where all hard constraints are first satisfied and as many of the soft constraints are maintained. This conflict resolution approach can be employed using the relax and enrich strategy (REST). While the framework does not prescribe a specific conflict resolution approach to implement, it is clear that it must support conflict detection and facilitate a resolution mechanism.

More recently, Human Resource Decision Support Systems have been developed to extend existing Human Resource Information Systems capabilities to include decisions on how, when, why, and who to select, evaluate, and reward [50]. A prototype framework for a human resource management system is depicted in Figure 2-16. Employee appraisal serves as the hub of the framework because an organization's capacity to evaluate employee performance is critical to improving organization performance. The Cognitive Resource Management Framework's Resource Appraiser Analyst is inspired by Kelemenis' model. Initial conceptualizations of the framework portray the Resource Appraiser Analyst role as one of which maintains critical data pertaining to resources and resource requests for post processing and offline analysis. Several systems have been developed which integrate Artificial Intelligence and Operations Research [51]. The possibility for implementing machine learning algorithms and using them during resource allocation is a fruitful area for future researchers, but is beyond the scope of this work.

The Cognitive Resource Management Framework is well suited towards analysis of the complexities behind sensor resource management onboard an unmanned ground vehicle. The framework, through its use of cooperating analysts provides the necessary data management and model management capabilities. The Plant Engineer component, intended for autonomous control of resources, can be modified to provide a user interface allowing the system to function as an operator training tool. Users have the capability to adjust the placement of sensors and simulate the same scenario to quantify if a more favorable outcome occurs. Likewise, entire missions can be simulated to determine which combination of activities, if any, suffer from resource conflicts. The analytical contributions added by this component alone are significant for system design and mission planning.

Robotics Related Resource Management Approaches

The Mobile Detection Assessment and Response System discussed in Chapter 1 is an example of a highly capable unmanned ground vehicle platform. It exemplifies the status of current UGV design and control of high capability vehicles. A prototype MDARS Patrol Unit Vehicle (PUV) is depicted in Figure 2-17. This PUV boasts both semi-autonomous navigation and RSTA capabilities; however, as the labels in the image indicate, each capability requires a dedicated set of resources each with a statically defined purpose at design time. These systems have limited fault tolerance as a result of design time resource constraints. This prototype underscores the infancy of robotic technologies that support high capability systems. This research facilitates the maturation of high capability technology by providing tools which aid the design, evaluation, and implementation of integrated high autonomy vehicles.

The preceding sections presented a survey of Computer Science and Business solutions addressing the resource management problem within their respective domains, along the way, parallels were drawn between those systems and a sensor network onboard an unmanned ground vehicle. It is now time to examine robotic approaches developed to address autonomous operation in dynamic environments. First, a discussion of existing architectures and their suitability is presented followed by a discussion of two different control/management approaches. The section concludes with a review of sensor modeling capabilities.

Architectures

At their core, robotic control architectures provide a mechanism for determining the response generated by a robot to a perceived stimulus. The “response” in this context, is a very generalized abstract concept not to be confused with Nilsson’s stimulus-response agents [52] which are purely reactive in nature. A better characterization of the control process involving autonomous vehicles illustrated through the sense/plan/act architecture [52] in which

uncertainties and other anomalous behavior are addressed through continuous feedback from the environment. This iterative approach begins with an initial sensing/perception stage, after which the system plans a sequence of actions. Note that only the first action within the sequence is executed. The system then re-senses the environment and compares the current state to the predicted state. A new plan is generated correcting for the error and the first step is executed. This approach, commonly used for autonomous navigation and obstacle avoidance, was demonstrated by Team CIMAR in the 2005 DARPA Grand Challenge by its receding horizon planner [7].

Reactive systems involve a perception and action cycle where each stimulus has a prescribed behavioral action. The planning cycle, if any, is trivial and typically involves some form of conflict resolution/behavior selection process. Rodney Brooks developed a layered hierarchy of behaviors in a control scheme known as a subsumption architecture [53]. At design time each behavior is assigned a priority, or placement within the layers of behaviors. During runtime, the mobile robot assesses the current circumstances and the suitable behavior at the highest control layer then subsumes the behaviors and lower levels. While reactive systems can find solutions where more complex approaches fail, Brooks' architecture is limited by the foresight of engineers at design time, i.e. their ability to predict which behaviors will be of importance. As a system's capabilities and mission complexity increases so does the task of layering the behaviors.

Ronald Arkin proposed a reactive architecture for mobile robot navigation using Motor Schemas [54] based on schema concepts originating in psychology and neurology. Each motor schema corresponds to a particular behavior that when combined with other schemas can form complex behaviors. However, each motor schema requires the specification of an associated

perceptual schema which is statically determined at design time. Unlike Brooks' approach, which utilized layering of behaviors, schemas are active individual computing agents that exist in a dynamic, soup-like network. This approach allows for the modular construction of schemas that aids in development and integration of new capabilities.

Deliberative systems, in contrast, rely heavily on the planning phase. These systems seek to formulate new plans incorporating newly sensed information regarding the environment with any system wide state changes detected. Often the planning stage involves deep reasoning along with the simulation of various alternative steps that can affect the overall utility of a plan. The inherent complexity and overhead added by the planning process can excessively tax computational resources thereby reducing the frequency of iterative cycles. As a result, roboticists have sought to develop hybrid systems capable of balancing the level of reactivity required to overcome operating environment dynamics with the generative planning capabilities necessary for operation in complex environments.

The autonomous ground vehicle navigation problem naturally lends itself to a hybrid system approach. Typical hybrid architectures like 4D/RCS and those remaining to be discussed are portioned into tiers based on time constants for reactivity. Bonasso describes a three-layer cognitive architecture [55], depicted in Figure 2-18, resembling typical three-level intelligent control architectures from AI texts [52]. The highest layer, termed the deliberative layer, is where planning and scheduling occur and is typically event-driven and characterized by greater time constants. Reactivity is achieved at the bottom layer, which is primarily concerned with executing primitive tasks it receives from the executive or middle layer. In *Plan Execution Monitoring and Control Architecture for Mobile Robots*, Noreils presents a detailed discussion of a three level control architecture for mobile robot navigation [56]. Figure 2-19 portrays

Payton's hierarchical hybrid system architecture for real-time autonomous vehicle control [57], a modification of the three-level architecture with an added layer of granularity. This seminal work addresses design constraints and strategies specific to autonomous mobile robot navigation. The highest level is tasked with translating abstract mission goals into a set of geographic goals to be used by the next module. The map-based planning layer translates geographic goals into route plans while the local planning layer oversees any reflexive actions needed while executing the route plan. The reflexive planning layer has the task of maintaining continuous vehicle control. It is important to note that Payton's architecture, like the APF and Arkin's Motor Schemas, requires a dedicated set of perception inputs for each layer and operates under the assumption of continuous data input. A contribution of the proposed research is the development of a framework to manage the sensor resources onboard a mobile robot platform in order to promote resource sharing across agents in a manner that ensures that the mission critical agents receive the data they need.

From the review of existing robotic architecture systems it is an apparent that a variation of the hybrid system approach, utilizing the Sense/Plan/Act paradigm, will be employed by the Cognitive Resource Management Framework. Furthermore, unlike the hybrid systems of today, the architecture of the framework components is driven by functionality and goal/task abstractions rather than time constants.

Control/Management Approaches

A mobile robot can be thought of as a collection of sensors, processing nodes, and effectors on a mobility platform. The autonomous capabilities of a mobile robot are achieved via controlling/managing (management and control can be considered synonymous in this section and used interchangeably) the interactions between the constituent elements of the platform. The remainder of this section discusses two competing control methods. Regardless of the

management approach taken, both reactive and deliberative systems alike must produce outputs that exhibit realistic and consistent behavior.

In general, distributed approaches are preferred when computation requirements necessitate a distribution of planning effort or when a single point of failure is unacceptable. Each entity within a distributed system operates largely independently, utilizing information gathered from locally available resources. This approach promotes rapid response to dynamic conditions and reduces communication requirements. Distributed approaches work best for problems that can be decomposed into mostly unrelated problems. The drawback of this approach is that highly sub-optimal plans often result because only local information is utilized.

Distributed or decentralized control in mobile robots was first introduced by Brooks' subsumption architecture [53]. Recall the architecture consisted of a collection of independent behaviors vying for control of the platform. Control was governed by the dominant layer in the architecture which was given authority to subsume all lower tiered behaviors. Designers layered behaviors and engineered them in such a way that emergent behaviors, the resultant of a blending of possible behaviors, were consistent. Arkin's Motor Schema architecture, yet another decentralized control example, utilized potential fields and fusion algorithms [54] for maintaining consistency. The output of each schema is a single velocity vector derived from a potential field formulation utilizing vector summation.

Perhaps the most popular example of decentralized control in robotics involves multi-robot coordination. *TraderBots* presents an economy-based approach to multi-robot coordination [58]. Zlot and Stentz extend Diaz's work by utilizing task trees to further subdivide tasks for trading in a market setting [59]. Multi-robot approaches to distributed control typically involve an auction mechanism. Tasks are submitted to a collection of self-interested parties in a virtual economy.

Each robot can negotiate for a task it wants or subcontract a task it already has. This approach requires the development of sophisticated bidding and auction clearing algorithms to ensure that tasks are assigned in a timely manner.

The principle benefit of centralized approaches is that they can be used to generate optimal plans by exploiting information within a centralized knowledge store. One area of concern is the system response to environmental changes. Any remotely detected changes must filter up to the knowledge store so the centralized broker can analyze it and devise a response; the response must then filter back down to the subsystems. If network bandwidth is sufficiently large enough to support the associated communication throughput the latency problems can be overcome. Another potential drawback of centralized approaches involves the computational requirements of optimal coordination; however, this may not be an issue given the nature of the problem or approximation methods used. The final area of concern involves the introduction of a single point of failure. However, the state of autonomous ground vehicle technology to date relies heavily on a number of mission critical systems which have no backups. In short, there are distinct benefits and drawbacks to be considered when designing a system; many of the deficiencies can be overcome, but ultimately the design parameters will dictate the approach.

Centralized management approaches are common among autonomous/semi-autonomous mobile robot standards for reasons previously discussed. JAUS and AS-4 architectures, for example, call for a “Subsystem Commander” component charged with controlling the resources onboard an AGV. The standards, primarily designed for teleoperation, do not address the appropriateness of one controlling action versus another. This is where designers must interject intelligent decision-making techniques such as a Fuzzy Logic based approach [60]. These architectures do include provisions for an Operator Control Unit (OCU) facilitating human in the

loop control of missions. Centralized management policies facilitate the implementation of human in-the-loop control, be it in a supervisory capacity, or for high level mission planning.

Expert Systems (ES) have successfully exploited centralized management approaches by replacing a human operator with a software subject matter expert, often with minimal integration cost. Expert Systems have been used in the fields of medicine, the food industry, and more recently, to optimize the operations of a natural gas pipeline [51]. An added benefit of the ES involves their ability to provide an explanation system, through retracting the steps involved in the decision-making process. The explanation mechanism can be of great use during systems design and debugging. The Resource Appraiser Analyst in the framework, inspired in part by expert systems, is intended to provide similar debugging capabilities to provide system designers with behind-the-scenes insight.

Blackboard systems are another example of centralized management policy utilized to integrate multiple activities. These systems use distributed expert modules with their own inference mechanisms and local knowledge to extract information which is stored in a central database, a blackboard. Liscano developed an architecture in which a blackboard's rule set and knowledge determine which activity controls a robot's actuators [61]. In this work he highlights the need for centralized global reasoning through a high-level decision-making module needed for a robot to understand the overall situation. Such a reasoning agent can act as an arbiter among activities to improve a system's performance. This same rational is embodied in Touchton's Adaptive Planning Framework, in which a Decision Broker, utilizing a collection of specialist data, selects the most favorable Operating Behavior [25]. The Plant Engineer operates in a similar fashion, utilizing information from the Resource Appraiser Analyst, Communicator Analyst, and Diagnostician/Systemizer Analyst to allocate resources.

Resource Modeling

The ability for a mobile robot to effectively interact with its environment is largely dependent on its ability to perceive the situation, derive a representative model, and apply reasoning techniques to the model to generate a commensurate response. Researchers have developed a myriad of models for the representation of knowledge including spatial, temporal, graphs, and other abstract models. The knowledge representation or type of model chosen is driven by the nature and application of the information.

A World Model Knowledge Store (WMKS) is a common model used by autonomous vehicle systems. The knowledge store serves as a repository for perceived, inferred, *a priori*, or transmitted data, information, or knowledge, that comprise a "world model." Typically, the knowledge stores provide mechanisms which allow the storing, querying, and analyzing of model data; however, the degree of sophistication is application and architecture specific. Highly autonomous vehicles and architectures such as 4D/RCS utilize sophisticated world models with simulation and prediction capabilities [6], while other models, need only store an *a priori* representation of the world such as a rasterized grid [4]. The structure of the knowledge store can be centralized, like the aforementioned blackboard approach, accessible by a collection of cooperating agents. Alternatively, each agent can maintain a local copy, receiving information on a need to know basis. Often, bandwidth, security, and remote processing capabilities dictate which architecture is implemented.

WMKSs within the AGV domain are commonly used to store geo-special information. These knowledge stores include methods for representing polygonal objects, topographical information, and other Geographical Information System (GIS) data. Recently, researchers have extended the use of knowledge stores to include storage and prediction of dynamic data [62].

This branch of modeling which addresses geographic and dynamic/time based data is referred to as spatiotemporal modeling.

In *Spinning Sensors*, the authors developed spatiotemporal models as part of a robotic sensor node middleware to promote coordination among actuated sensors [63]. Their work focused on achieving maximum effectiveness with minimal consumption of resources. The obvious solution is to outfit a vehicle with an extensive sensor network that provides large coverage and high granularity. However, this approach is only feasible if sensor nodes are cheap and if the platform has suitable network, I/O, processing, and power bandwidth to support such an array. The alternative approach, adopted by the authors and TEAM GatorNation on the Urban NaviGATOR, is to construct a network of static and articulated sensors. This sensor actuator pairing increases coverage while decreasing the number of sensors required.

Aoki and Nakazawa developed a sensor and actuator classification scheme in an attempt to characterize the constituent elements of a sensor node in an abstract fashion. Their efforts, while too rudimentary for many sophisticated mobile robot applications, provide a foundation upon which to build this research. The existing sensor classification scheme is based solely on two criteria: sensing range and direction. Sensors which cover a few meters are considered "long range," while sensors which that cover only a few centimeters are dubbed "short range." Sensors such as thermometers and hygrometers, which cover 360 degrees, are termed "non-directional," whereas microphones, cameras, and illuminometers possessing limited coverage angles are termed "directional." Furthermore, the middleware assumes a primitive set of single degree of freedom actuators and the models developed do not support the serialization of actuators into multiple degree of freedom manipulators.

The Spinning Sensors Middleware supports spatial, temporal, and real-time models for sensor nodes. Figure 2-20 depicts the parameters for both spatial and temporal modeling. The existing models assume sensor coverage over a two dimension space. The existing spatial models are used to estimate the coverage area of a sensor network system, but are insufficient for determining if a particular sensor can provide meaningful coverage of an object at specific location in 3-dimesnional space, as many RSTA and AGV navigation applications require.

Temporal models, also two dimensional in nature, consist of specialized case based equations for computing the maximum sensor coverage time of an obstacle. These models, depicted in Figures 2-21 and 2-22, require predicting the behavior of the target in order to appropriately estimate coverage time; a capability which is not addressed by the current middleware. Temporal information could be used within a resource allocation algorithm to select among a number of candidate resources. An important development of Aoki and Nakazawa's work is the notion of a real time model. This model parameterizes real-world system latency in terms of time to sense and communicate data, processing time, and time to communicate and control the actuator. These parameters are incorporated to form an inequality comparing temporal information with real world constraints, which is used to estimate the fitness of a sensor choice. This technique may be incorporated into an informed search used by the Plant Engineer to prune possible resource solution sets.

The Cognitive Resource Management Framework seeks to optimize resource allocation with respect to varying mission goals, rather than just minimizing consumption. The Diagnostician/Systemizer Analyst incorporates 3-dimensional spatial and temporal models to satisfy AGV perception requirements while supporting serialization of actuators. Furthermore, the framework extends the classifications of sensors to incorporate an abstraction of capabilities.

The Cognitive Resource Management Framework extends spatiotemporal modeling capabilities outlined in *Spinning Sensors* to realize the autonomous allocation of sensor resources onboard an AGV to support advanced mission capabilities.

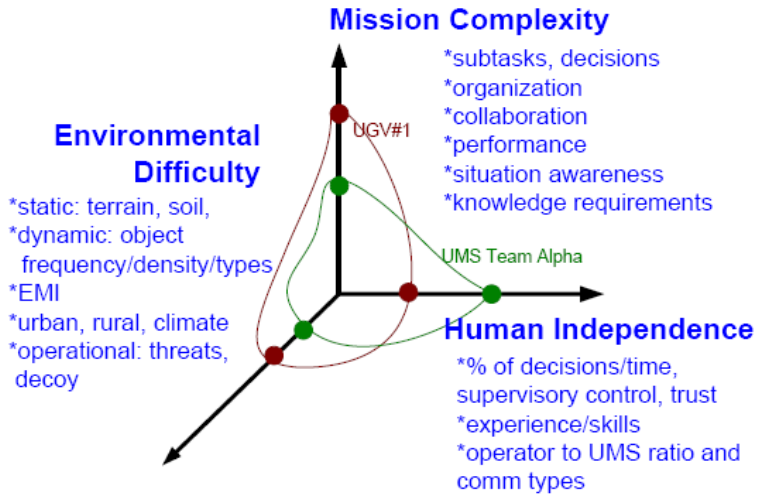


Figure 2-1 ALFUS Axes for Specifying Autonomy Level [2]

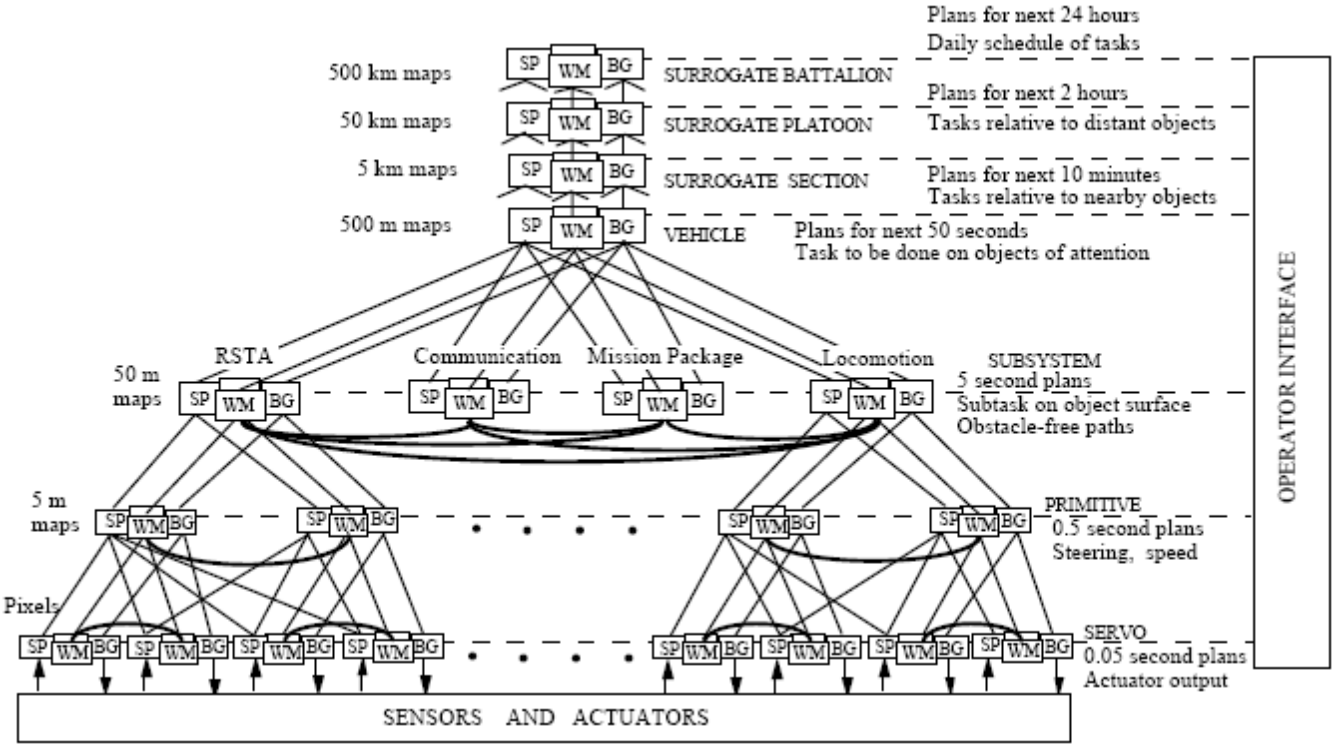


Figure 2-2 NIST 4D-RCS Reference Model Architecture [6]

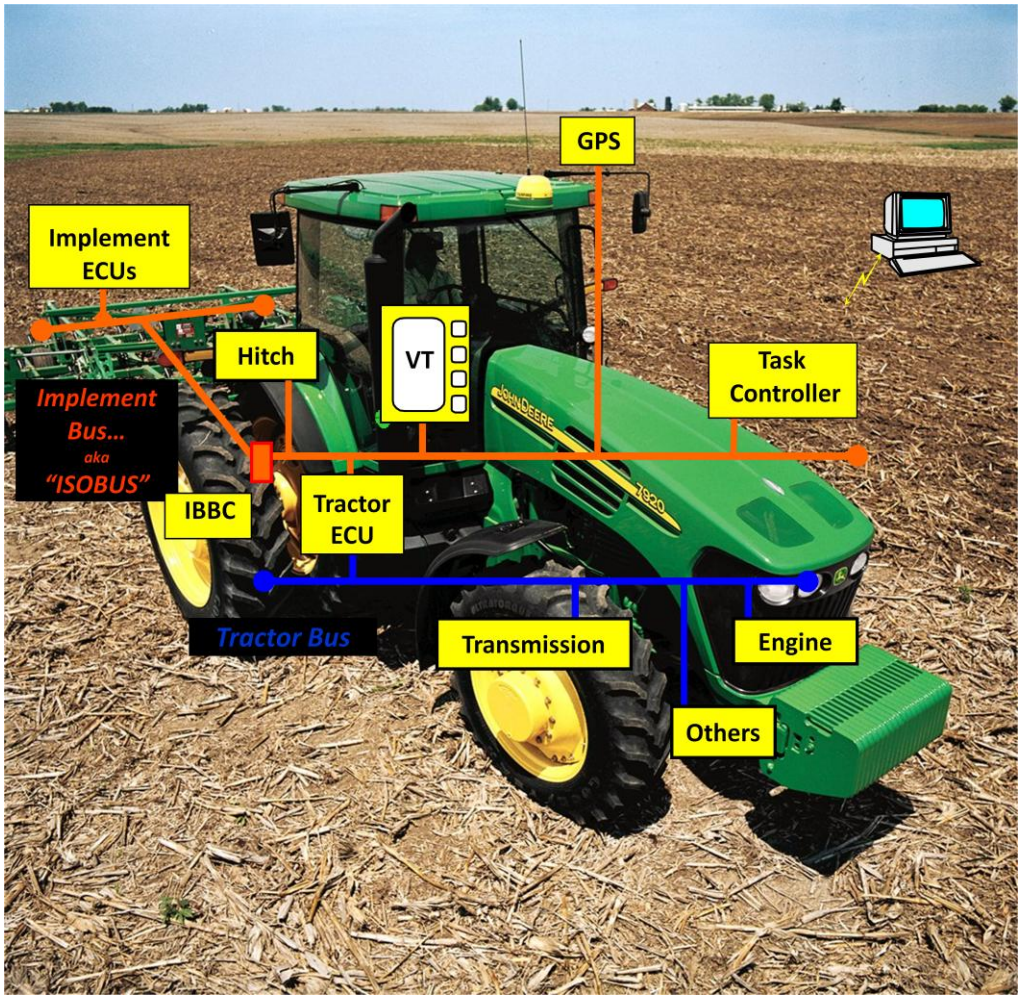


Figure 2-3 ISO 11783, ISOBUS, Implementation Visualization [64]

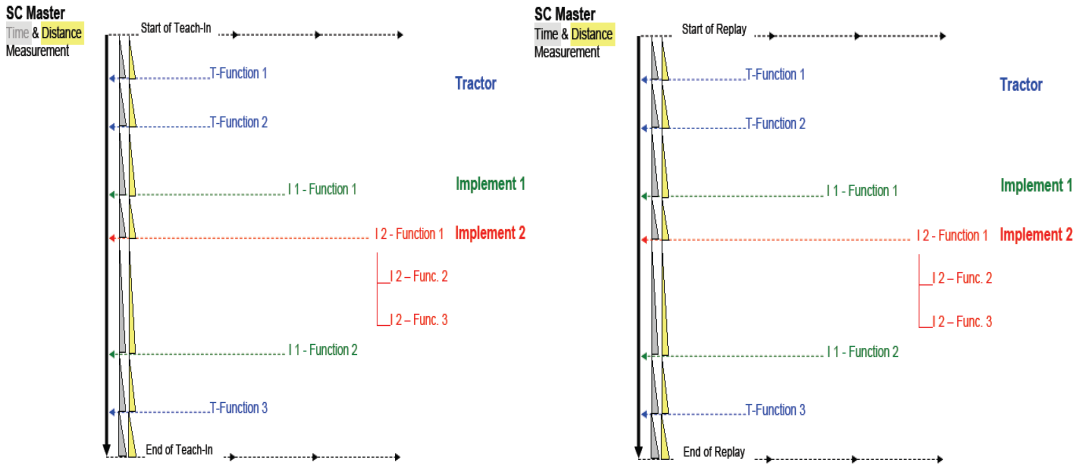


Figure 2-4 ISO 11783, ISOBUS, Sequence Automation [64]

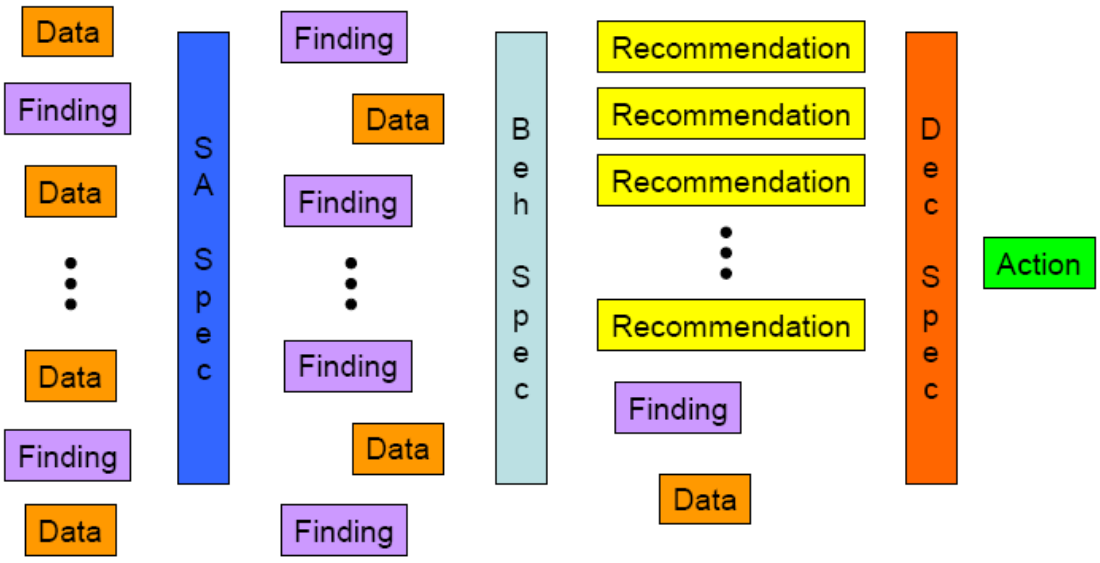


Figure 2-5 Adaptive Planning Framework Conceptual Model [25]

Behavior Use Case Template
(Deliberative)

Scenario Description:

Assumptions:

Constraints:

Entry Conditions:

Exit Conditions:

Inputs Consumed:

Outputs Produced:

Steps for Deliberative Behavior:

Step #	Action	Contingency Action
1	Action to take for 1st Step	Action to take if 1st Step fails
2	Action to take for 2nd Step	Action to take if 2nd Step fails
3	…	…

Figure 2-6 APF Knowledge Engineering Tool: Behavior Use Case Template [25]

Findings Worksheet

Specialist:

Finding: **Type:**

Possible Values:

Rule(s)/Algorithm(s):

Element	Comments
Rule/Algorithm for finding 1st Possible Value	
[optional Rule/Algorithm for alternate ways of finding 1st Possible Value]	
Rule/Algorithm for finding 2nd Possible Value	
…	

Figure 2-7 APF Knowledge Engineering Tool: Findings Worksheet [25]

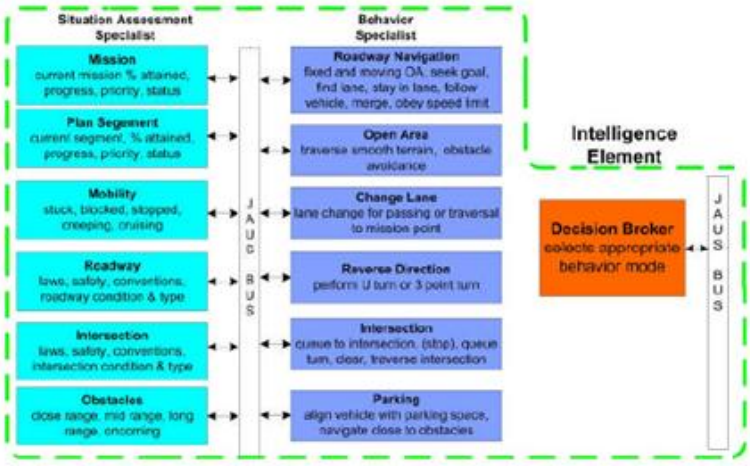


Figure 2-8 Gator Nation 2007 DARPA Urban Challenge Adaptive Planning Framework Implementation [26]

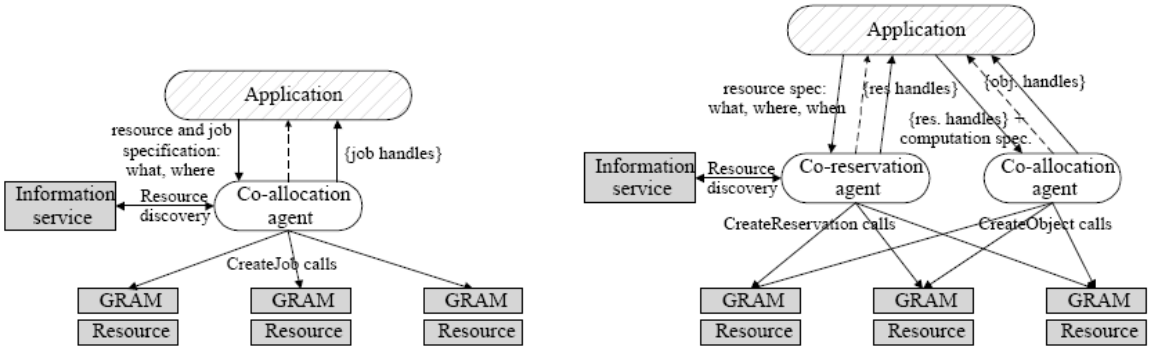


Figure 2-9 Globus and GARA Resource Management Architectures [31]

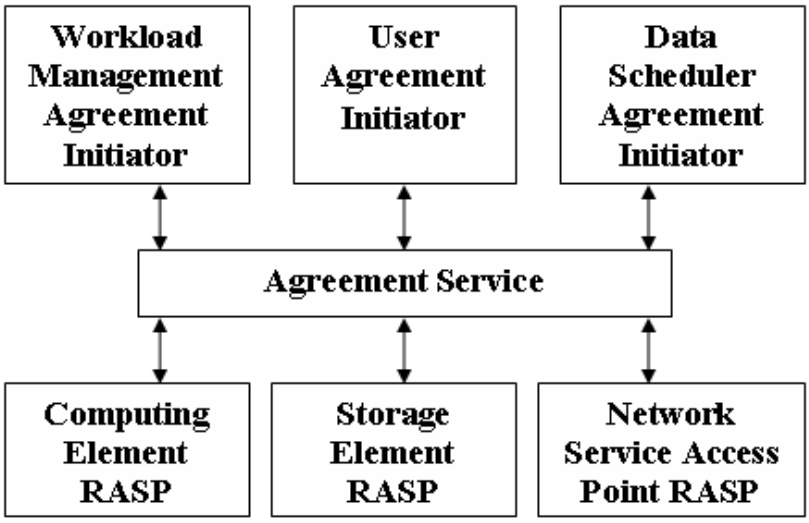


Figure 2-10 Agreement Service and Related Architecture Components [32]

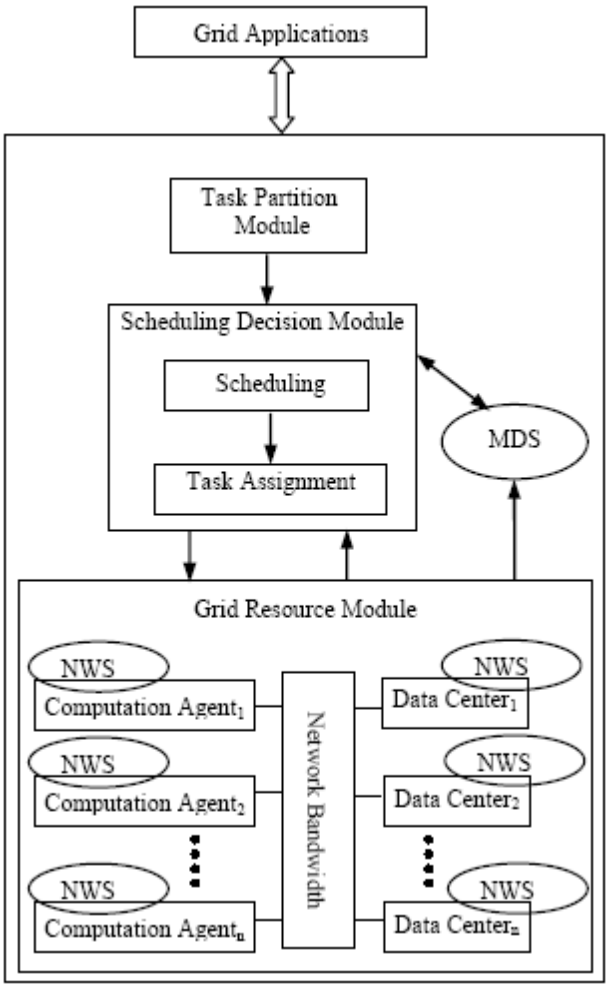


Figure 2-11 Model of Grid Resource Management and Scheduling [35]

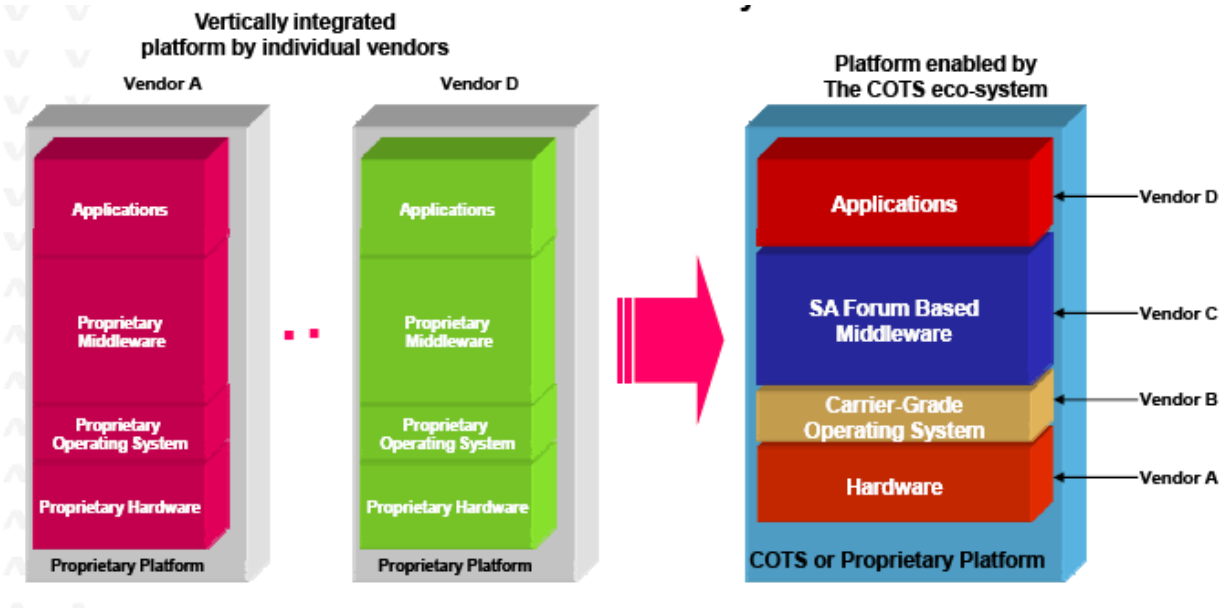


Figure 2-12 Service Availability Forum Model [37]

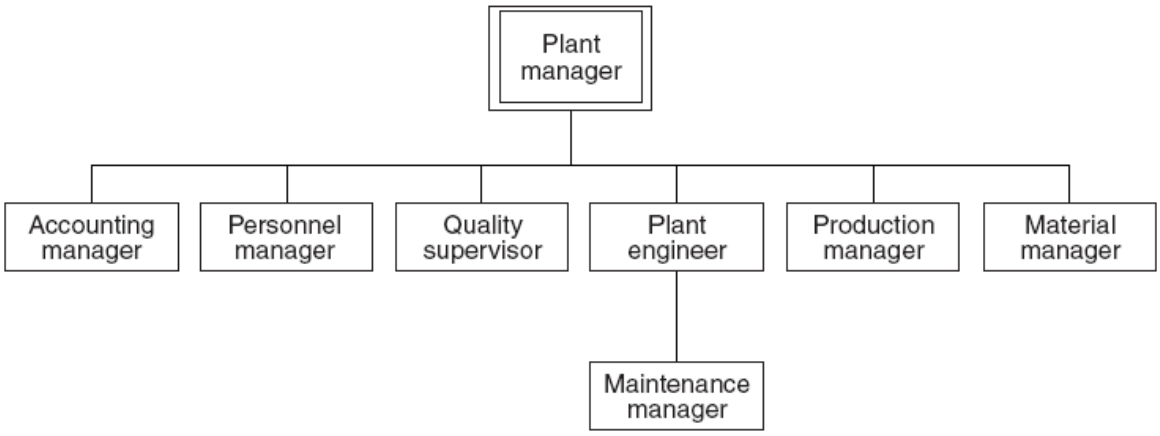


Figure 2-13 Typical Organization of a Small Plant [44]

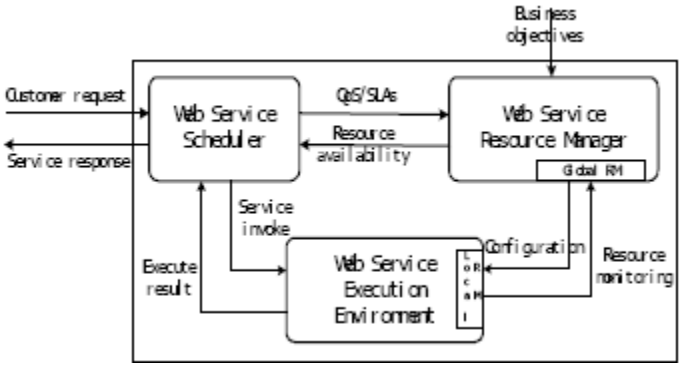


Figure 2-14 Adapting Business Service Capability Framework [47]

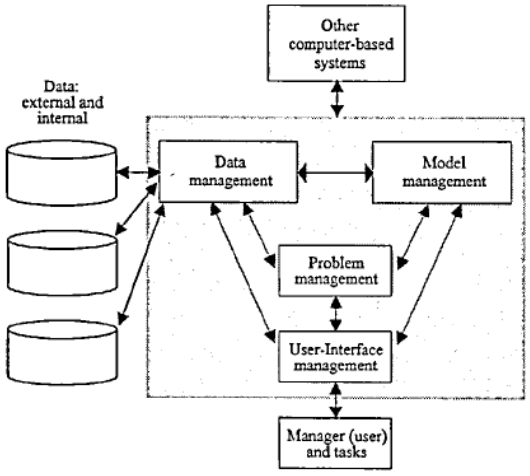


Figure 2-15 Decision Support Systems Model Augmented with Problem Management [48]

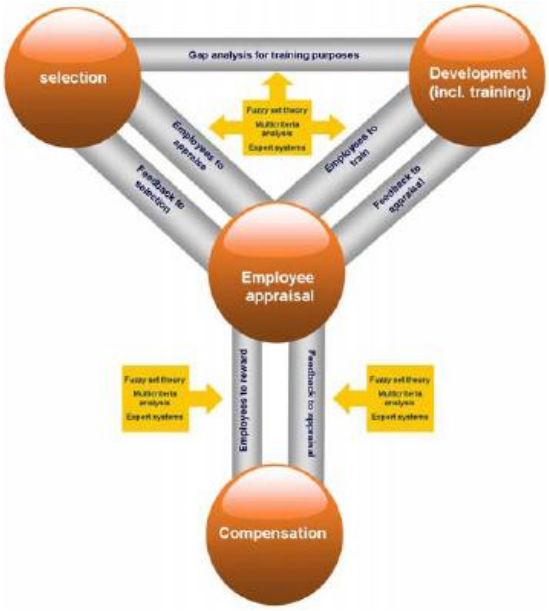


Figure 2-16 Human Resource Management Framework [50]

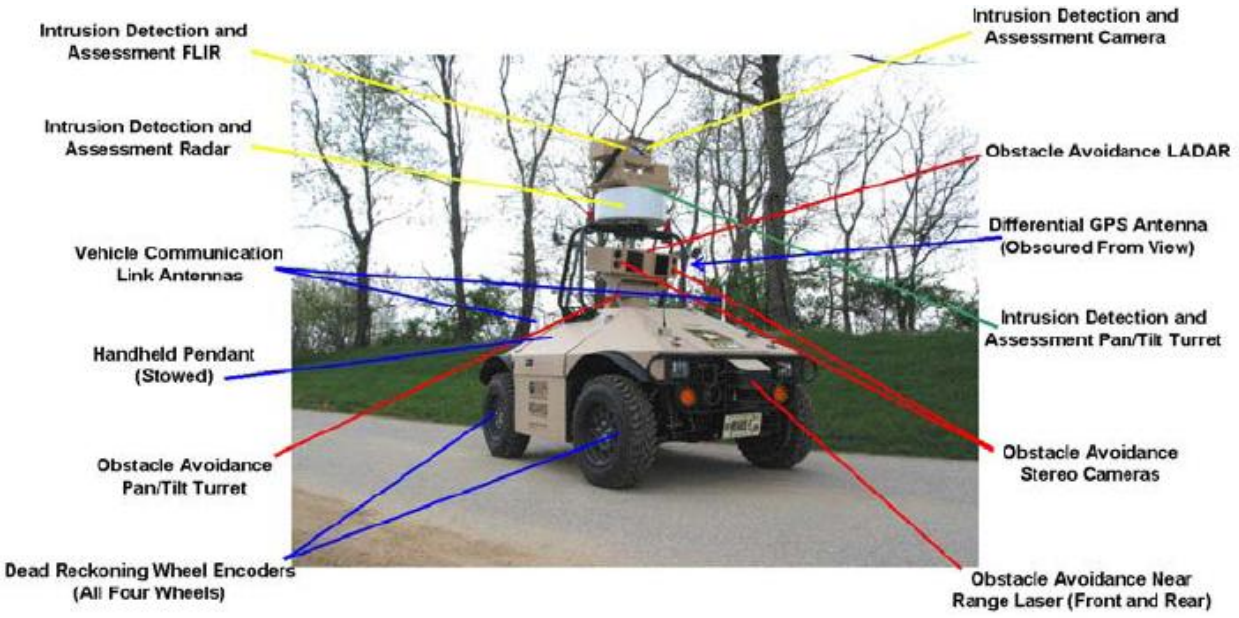


Figure 2-17 MDARS Patrol Unit Vehicle [10]

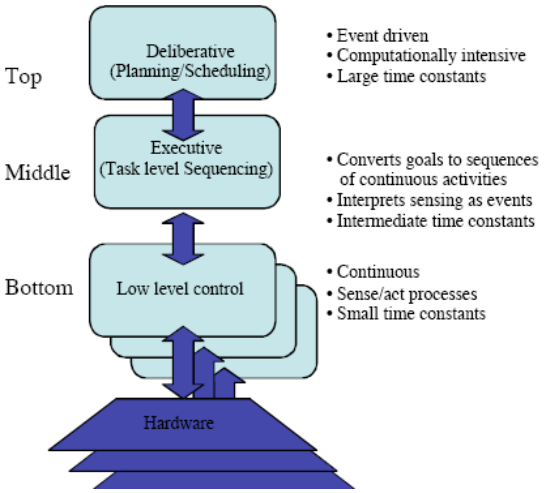


Figure 2-18 Three-layer Cognitive Architecture [55]

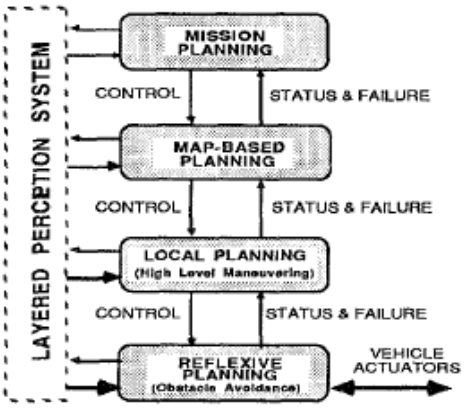


Figure 2-19 Hierarchical Architecture for Real-time Autonomous Vehicle Control [57]

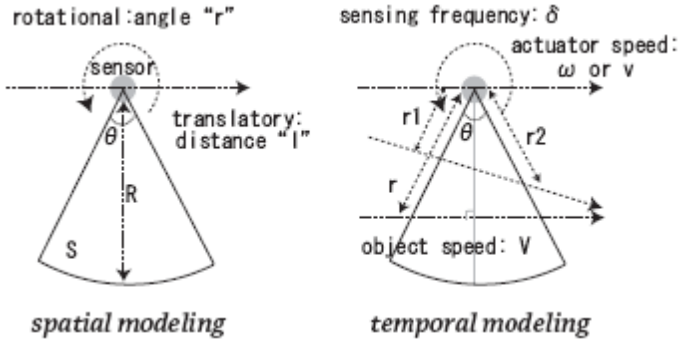


Figure 2-20 Spinning Sensors Spatial and Temporal Models [63]

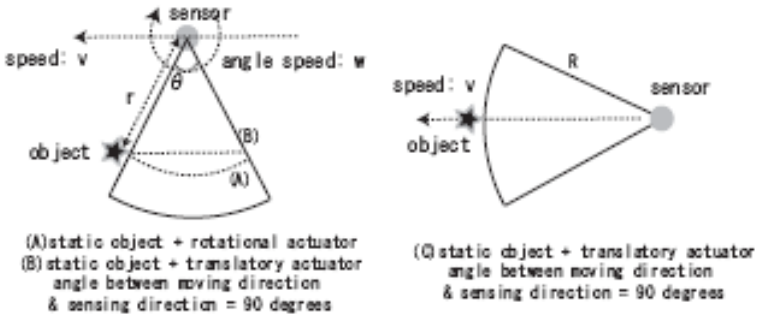


Figure 2-21 Spinning Sensor Static Object Temporal Models [63]

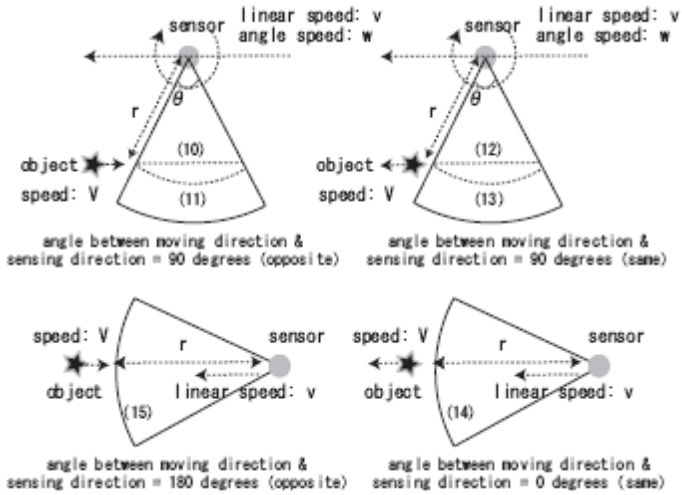


Figure 2-22 Spinning Sensors Mobile Object Temporal Model [63]

CHAPTER THREE
THE COGNITIVE RESOURCE MANAGEMENT FRAMEWORK

This chapter presents the Cognitive Resource Management Framework, the result and principle deliverable of the research performed. To effectively manage resources the framework must provide mechanisms which support:

- Resource Discovery
- Job Submission
- Resource Allocation/Scheduling
- Resource Modeling and Monitoring
- Goal Abstraction

To provide cognitive resource management, the framework performs world modeling, value judgment, and behavior generation based on acquired abstract representations of a system's capabilities, operating guidelines, and specific jobs to be performed. The subsequent discussion addresses the design, reasoning mechanism, and tools developed for the Cognitive Resource Management Framework.

Framework Architecture and Representation Scheme

To facilitate implementation and integration with existing vehicle systems and standards, the framework defines a knowledge representation scheme along with an organizational architecture for interpreting and managing system-wide resource information. A Plant Engineer element along with a distributed collection of cooperative Analysts are utilized to construct a representative model of an AGV's capabilities:

- Plant Engineer (PE) ─ a single centralized authority given sole discretion over assigning a value to incoming jobs, and to allocate and provision the resources necessary to fulfill the job based on all acquired analyst knowledge
- Diagnostician/Systemizer Analyst (DSA) ─ charged with creating and maintaining a real-time representation of system state information, including resource capabilities and configuration

- Communicator Analyst (CA) ─ plug-in interface acting as gateway between CRMF and vehicle control and planning architectures for importing vehicle mission and state information

- Resource Appraiser Analyst (RAA) ─ oversees performance management through the collection of framework data for real-time or offline processing

- Resource Analysts (RA) ─ each devoted to a particular resource, relays abstract resource attribute representation to DSA for discovery and monitoring

- Application Analysts (AA) ─ each devoted to a particular software application requiring the utilization of a resource, converts application specific resource need into generalized framework Job Request for submission to the Plant Engineer

Successful framework operation requires a high degree of interaction among these elements involving the production and consumption of framework specified knowledge. There are three principle CRMF knowledge representation types:

- Resource Object ─ contains a collection of resource attribute information utilized for resource discovery, modeling, and performance monitoring

- Job Request ─ contains a collection of job attributes which provides a representation of the resource need at job submission for resource allocation, scheduling, modeling, and performance monitoring

- Meta-Knowledge ─ container of system state information conveying representations of the mission and vehicle state information for use during allocation, scheduling, modeling, monitoring, and for goal abstraction

The following sections address the logical organization of resources, framework architecture, knowledge representation and individual framework components.

Resource Virtualization

The first step towards abstract resource management requires each ground vehicle be logically organized into a "Virtual Plant," a representative example is shown in Figure 3-1. Stacked resources represent a tightly coupled resource set, analogous to Aoki and Nakazawa's sensor node concept consisting of a sensor/actuator pair. The Virtual Plant visualization is used as a design and analysis tool when organizing vehicle software and hardware architecture

components and for modeling their dependencies. Additionally, it provides a roadmap for implementation, indicating the number of Application and Resource Analysts needed along with their respective "host" (both hardware and software) locations.

Framework Architecture

Autonomous operation in a dynamic environment presents engineers with the problem of designing a system which combines the ability to react quickly and the ability to plan actions to achieve complex goals. Common approaches such as a Nillson's three-level architecture [65] and Albus' perceptual tower architecture [66] call for a fusion of reactive and deliberative strategies. These architectures utilize different deliberation paths from sensory signals to motor commands. To maintain reactivity they include a "short circuit" path from sensors to effectors thereby bypassing all deliberation. In Albus' approach and other layered-architecture approaches such as Bonasso's Cognitive Architecture [55], components are organized based on their required level of reactivity. A component's planning horizon determines its location within the architecture with lower layer elements possessing the smallest time constants. These approaches are quite cumbersome, requiring redundant calculations and high levels of data transport among all the different elements. In contrast to the approaches just discussed, the CRMF calls for a deliberative approach using a hierarchical architecture based on knowledge abstraction and functionality to achieve real-time asynchronous operation.

The functional elements of the Cognitive Resource Management Framework are organized in hierarchical fashion as depicted in Figure 3-2. At the lowest and most primitive level, "sensory" signal information is distilled and assembled by Resource Analysts and Application Analysts into specific knowledge attributes which are then utilized to meet the information needs of the intermediate and executive level elements. Information flows vertically between the primitive, intermediate, and executive levels of the framework, hence the

hierarchical architecture. At the intermediate architecture layer, the Diagnostician/Systemizer Analyst, Communicator Analyst and Resource Appraiser Analyst assimilate all accumulated knowledge into a higher level, more abstract representation, to be used by the Plant Engineer when fulfilling its duties. At the intermediate layer, knowledge pathways are both horizontal and vertical, i.e. horizontally between the intra-layer analysts and vertically between inter-layer elements. Figure 3-2 illustrates these architecture information pathways. Knowledge representations between the Plant Engineer and the intermediate level analysts can vary greatly depending on the framework application domain, implementation details of the DSA model, and brokering and allocation strategies applied at the executive layer. The reference implementation, described in Chapter 4, details the knowledge representations used by a spatiotemporal DSA sensor system model and a value based Plant Engineer.

Through the judicious application of information fusion and knowledge abstraction, this layered, hierarchical architecture facilitates the real-time management of resources at the macro or capability level. Each layer of this architecture uses more abstract perceptual predicates than the lower ones. The significance and appropriateness of knowledge abstraction throughout the framework cannot be understated. Research has shown that managers' mental models affect what they see; two managers with different models can conceptualize resources differently and also suggest different relevant resources [42]. This dominant logic [43] acts as an information filter, focusing the efforts on certain information deemed relevant. The prudent use of abstraction throughout the architecture hierarchy underpins the effectiveness of the Cognitive Resource Management Framework.

Resource Object

Resource-specific knowledge is represented within the CRMF as a Resource Object, which is populated by a Resource Analyst and then disseminated throughout the framework. The

Resource Object serves as a container for specific resource attribute information in the form of "Capability Attributes" and "Performance Attributes". Each resource attribute must convey a unique piece of information possessing an unambiguous interpretation. This information is used to construct an abstract resource representation within the Diagnostician/Systemizer Analyst for resource discovery and modeling. Additional uses include performance management and monitoring, which occurs within the Resource Appraiser Analyst. The role of the system engineer is to determine which resource attributes need to be included to construct and sustain an accurate real-time system model. The final section of this chapter presents knowledge engineering tools designed to assist engineers in this process.

Capability Attributes are intended to identify a resource and describe its function. These attributes might convey capability classification knowledge in the form of a predefined resource type using a byte or character data type. Alternatively, textual information could also be contained within a string-type capability attribute whose semantics are determined at run-time by some intelligent entity operating within the framework. Each attribute is designed to covey a new piece of knowledge utilized during the resource management deliberation process which leads to actionable results. Figure 3-3 illustrates examples of Capability Attributes used within the Resource abstract base class. The "resourceType" attribute is an example of an *a priori* classification system. This particular field denotes whether the resource can be classified as either a sensor or actuator. The DSA would then maintain the mapping of which job types a resource of such a type is capable of performing. A more sophisticated DSA implementation may utilize natural language processing to interpret the "textDescription" attribute and construct a viable capability-to-resource mapping. Whichever approach is taken, these attributes are responsible for resource identification and capability assessment.

Performance Attributes are used to describe the performance of a resource through some numerically quantifiable metric. The parameter is described in terms of a known unit of measurement specified in the attribute definition worksheet. Possible Performance Attribute data types include int, float, and double. These attributes convey vital information describing a resource's operating criteria. The DSA utilizes this information when constructing the system model to determine the extent and degree of a resource's capabilities. The nature of this information can vary greatly from implementation to implementation depending on the type of system model selected. Sample performance attributes shown in Figure 3-3 describe the position and orientation of a particular resource with respect to a base reference coordinate system, information necessary to assess the coverage provided by a sensing resource. The "minVelocity" attribute, an actuator-specific performance attribute, might be used to compute a worst-case estimate of servo time which may render an otherwise capable resource as an unviable candidate.

Resource Objects act as containers for a collection of Capability and Performance Attributes. While resource technologies vary greatly and are continuously being refined, there exist commonalties between them. The Cognitive Resource Management Framework prescribes a modular and extensible class structure when defining each resource object to exploit commonalities and to mitigate the impact of technological advances. Each Resource Object inherits from an abstract base resource class, illustrated in Figure 3-4. This approach not only prevents the duplication of information, which minimizes communication bandwidth, but also minimizes the integration cost of incorporating new resource technologies. Each class hosts a collection of Capability and Performance Attributes selected in order to distill resource information down to the essentials needed by the Intermediate Layer CRMF Analysts. The

reference implementation, discussed in Chapter Four, presents the Resource Object syntax developed for representing the sensors and actuators deployed on CIMAR's Urban NaviGATOR.

Job Request

The framework Knowledge Representation Scheme defines a Job Request which houses a collection of Capability Request Attributes and Target Attributes. This collection of attributes sufficiently describes the nature of the job to be performed and the object which may be the subject of the job. The Application Analyst is tasked with populating and submitting a new Job Request whenever a software routine requires the utilization of a resource which is currently managed by the CRMF. Knowledge conveyed within each Job Request is utilized to perform the critical operations of value judgment, capability-to-resource mapping, allocation and scheduling of resources, and performance management and monitoring. In this fashion, the framework is able to manage requests for complex operations, often consisting of a number of sub-operations, by abstracting those needs to the capability level. For example, rather than request for a series of individual manipulation commands for positioning a specific articulated sensor, a software program need only request for range information of an object at a given georeferenced point and the framework will return a suitable resource capable of providing and fulfilling the requested capability. At design time the roboticist must determine which resource attributes are needed to represent a resources capabilities requirements of the software application. A series of knowledge engineering tools are included at the end of this chapter to assist in the Job Request engineering design process.

Capability Request Attributes encapsulate key knowledge elements pertaining to the task to be performed for incorporation and digestion by the Cognitive Resource Management Framework. New attributes are specified using the framework tools developed as part of this research. Like their Capability Attribute counterparts, contained within a Resource Object,

Capability Request Attributes must contain a representation of the type of resource job the application requires. This representation can exist in the form of a predefined enumeration scheme, as a natural language textual description, or can take on some other form. There are tradeoffs to each approach such as transmission bandwidth, implementation complexity, and robustness to new capability requests. Ultimately the method chosen is a design choice and depends on the capability-to-resource mapping technique implemented within the Diagnostician/Systemizer Analyst.

In addition to task abstraction information, the Capability Request Attributes are used to convey information needed to support more explicit resource requirements. Examples of these requirements include advanced reservation of resources, explicit resource assignments, or even privileged user information. Advanced reservation support requires the inclusion of a viable time window for scheduling along with an estimate of the job's duration. It is foreseeable that system architects may equip a vehicle with certain resources intended for a specific and predetermined purpose. The framework can support this level of explicit tasking by including the unique resource name or unique resource ID as Capability Request Attributes. A slightly less restrictive approach advocates that certain resources be reserved for a "privileged" user-group. An implementation of the CRMF could easily support this through the inclusion of a user ID field, which discloses the identity of the requesting Application Analyst. Furthermore, designers may choose to craft Capability Request Attributes which denotes a desired quality of service level for a job. Through the use of Capability Request Attributes, system designers can tailor an implementation to the desired level of complexity commensurate with the operating guidelines.

The second attribute category contained within the Job Request knowledge representation, known as Target Attributes, is dedicated to providing information regarding the subject on which

the task is being performed. As with all other framework attributes, new Target Attributes are defined and documented in accordance with framework specified worksheets. The data representations of these attributes can vary greatly depending on system model choice, value judgment parameters, and other proprietary object characteristics. A spatial model might require target information in the form of a PostGIS geometric polygon representation. Temporal models dealing with dynamic objects might require knowledge regarding the heading or velocity of the target. Target attributes may also include *a priori* classification information in the form of enumerations if the application, through some external means, is able to categorize the object with an appropriate degree of certainty. Because this framework presents a generalized approach towards resource management, it provides generic methods and tools that designers can utilize to implement custom solutions.

The syntax developed for representing the resource capability requests for the Environmental Mapping and Change Detection demonstration and Reference Implementation deployed on CIMAR's Urban NaviGATOR are presented in Chapter 4 and Appendix C.

Meta-Knowledge

Information derived within the intermediate framework architecture layer is represented as Meta-Knowledge. This information is utilized by consumers to support resource management activities at the intermediate and executive layers as illustrated in Figure 3-2. The Meta-Knowledge container hosts system state information conveying mission, platform-state, and proprietary DSA model-specific data representations needed to facilitate resource allocation, scheduling, modeling, or monitoring. The use of this framework specified container provides a vehicle through which information fusion can take place. Meta-Knowledge elements originating within the Communicator Analyst encapsulate critical system information which would otherwise remain foreign to the framework. The knowledge is derived by external means, but is

converted into a format that the framework can interpret. Examples include system pose and velocity information, which are made readily available by common unmanned systems architectures such as JAUS, information critical in supporting a real time spatial system model. Other more abstract representations, such as system state information, may be used to interpret the operational objectives of the platform, allowing the Plant Engineer to alter its allocation strategies accordingly. Meta-Knowledge may also result from performing computational operations within the Resource Appraiser Analyst and the Diagnostician/Systemizer Analyst. This new insight is derived directly from its Resource Object and Job Request knowledge representation counterparts. These elements may vary greatly in structure and content depending on design time implementation details, such as desired model and allocation strategy. For example, the DSA publishes a Candidate Resource List Meta-Knowledge element which the Plane Engineer utilizes when selecting among possible resource alternatives. The contents of this list are generated as the end product of behavior generation activities occurring within the DSA. Framework tools formalizing the design and documentation of these new elements are discussed at the conclusion of this chapter.

Resource Analyst

The Resource Analysts provide a generalized abstraction of AGV hardware resources and serve as the bridge between the framework and the hardware. This approach facilitates implementation by giving roboticists the freedom to architect system layout and resource connectivity in a fashion best suited to the application. Furthermore, it minimizes integration costs on existing platforms because it requires no changes to existing hardware connectivity. The framework supports both centralized hardware implementations, in which all resources are connected to a central I/O node, and distributed ones where resource connectivity is spread among a number of computation nodes. Resource Analysts are intended to symbiotically coexist

one layer of abstraction above any existing low level resource specific I/O control. Each resource to be managed by the framework must have a corresponding Resource Analyst attached to it as depicted by the yellow boxes in the resource virtualization illustration of Figure 3-1. These analysts are subject matter experts in the resource they represent and are responsible for disclosing capabilities for discovery, allowing for configuration and reconfiguration of the underlying resource, and generating resource status reports.

Resource Analysts communicate with the Resource Appraiser and other upper-layer analysts using framework specified syntax. One communication mode is via a Resource Object, which provides resource parameters and specifics needed for discovery and modeling. Figure 3-5 shows a draft Resource Object schema for common single degree of freedom linear or rotary actuators. The framework does not prescribe a specific discovery protocol nor does it mandate that the Resource Analyst transmit a full resource description. The extent of a model's *a priori* resource knowledge is an implementation specific design choice. An example of a Resource Object schema describing an actuator whose parameters are known *a priori* is presented in Figure 3-6. In such cases, the RA need only disclose Capability Attributes for identification purposes. The framework merely requires that a Resource Analyst provide the Resource Appraiser Analyst enough information needed by the particular framework implementation to model the sensor and its parameters.

In addition to enabling discovery and control of a resource, the analyst must support the real-time modeling of capabilities. At a minimum, the Resource Object Capability Attributes should contain basic status information regarding health such as "operational" or "failure." The information is utilized to update the resource model to provide a graceful degradation of capabilities or to signify to a supervisor that the vehicle can no longer effectively meet the

mission objectives. The contents and semantics of these status Capability Attributes are to be predetermined at framework implementation time, defined on an as-needed basis and cataloged within the framework glossary.

Application Analyst

Application Analysts can be thought of as a Resource Analyst's software counterpart. These analysts form a link between the framework and the AGV software applications providing an abstraction of desired tasks which require the utilization of managed resources. Existing technologies can be incorporated into the CRMF by supplementing the implementation with an Application Analyst. The analysts specify resource requests through the use of a Job Request. Instead of explicitly requesting a particular resource; applications can request a capability allowing the framework to select the most suitable candidate among all available resources. This approach leverages a vehicle's autonomous capabilities and self awareness to provide robust and optimal runtime solutions rather than following a preset script created at design time under assumed conditions whose relevance is unproven.

For example, an unmanned vehicle equipped with multiple cameras may request an *image capture* capability to document the detection of an unknown object. Typical implementations either require the application have a dedicated resource it can monitor and task to provide the requested capability, or the application must possess sufficient knowledge regarding the resource's utilization by other applications for other tasks. Under the proposed framework, an Application Analyst is responsible for submitting a job to the Plant Engineer, which deliberates over the importance of the job and allocates the most suitable resource set to the application.

The CRMF both generalizes and centralizes the resource management functionality so that individual applications are spared the additional overhead associated with distributed management systems. Furthermore it has been shown in [67] that a centralized management

policy simplifies management protocols, eases implementation, and facilitates data consistency, all with less message traffic than partially decentralized or fully decentralized approaches.

Resource Appraiser Analyst

The Resource Appraiser Analyst is responsible for overseeing the collection of framework data for real-time and offline analysis. This analyst's primary role concerns system monitoring and performance management; as such is involved with every facet of framework operations. The most critical aspect of performance management is the ability of the RAA to perform value judgment regarding the fitness or suitability of a resource to handle a particular job based on previous performance appraisal. In order to provide performance management the Resource Appraiser Analyst must monitor resource consumption and utilization time, log requests for capabilities, and collect resource update data and job completion data. It must then compile and evaluate these performance statistics before any actionable measures can be undertaken, either autonomously or through human intervention.

During discovery, the RAA catalogs the resource check-in, forwards the information to the Diagnostician/Systemizer Analyst (DSA) which adds the resource to the model and assigns the resource a unique system identifier. Upon the successful completion of check-in protocols, the RAA sends a confirmation notice to the RA. Resource updates, in the form of Resource Objects, are received and logged by the RAA, which subsequently alerts the DSA if any changes in the model are required. Job requests, which are submitted to the Plant Engineer, are forwarded to the Resource Appraiser Analyst for record keeping purposes.

Under the current iteration of the framework, presented in the following chapter, the RAA functions as a framework knowledge repository operating in data logging and mining capacities. This information, once analyzed, can yield invaluable system diagnostic material. This functionality will provide developers with tremendous insight regarding the inner workings of a

highly autonomous system. Other more advanced implementations may harness the appraiser's performance monitoring capabilities to augment the resource brokering process with neural network learning or even rule based expert system resource recommendations. An explanation mechanism describing the rationale behind provisioning decisions could be implemented and should be of great use when field testing new technologies. These are just a few fruitful areas of research that build upon the existing framework to enhance unmanned system robustness and autonomy.

Diagnostician/Systemizer Analyst

The Diagnostician/Systemizer is tasked with maintaining a suitable system model that the Plant Engineer can query during periods of deliberation. The nature and capabilities of the system representation are largely implementation driven, however, at a minimum, the model should provide resource state information.

The DSA acts like a world model knowledge database. It can contain *a priori* information available to the system before operations begin, as well as *a posteriori* knowledge gained through situational awareness. This component could contain information pertaining to space, time, events, and states of framework elements and the external world. It may also contain knowledge about motives, goals, and priorities of tasks. Knowledge about the physics and resource operations, rules, and logic must be represented in appropriate forms such as function parameters which then generate predictions on the viability of a resource choice. The correctness and consistency of the DSA system model is dependent on the validity of the knowledge it receives from Resource Objects, Job Requests, and Meta-Knowledge.

Additional analyst responsibilities include updating the models to account for changes detected by the Resource Appraiser Analyst and answering Plant Engineer queries with appropriate Meta-Knowledge. The DSA registers resources upon check-in assigning them a

unique identifier. Furthermore, the model must account for resource dependencies such as sensor actuator pairings, for example. Should the RAA report an actuator failure in the camera panning mechanism, it is the job of the DSA to adjust the model to a static camera configuration.

The Diagnostician Systemizer Analyst is charged with matching capabilities to resources. The framework grants system designers the freedom to provide this functionality through the implementation of their choosing. Some implementations may adopt a rule-based approach to match capabilities to specific resource types. Other implementations may choose to employ a more complex constraint satisfaction approach such as REST [48], using a combination of hard and soft constraints. In this manner the framework remains general and applicable to a greater range of problem areas while providing opportunities for future technology development.

The specific capabilities and implementation details are largely left up to the designer; however, the concept and generalized role of the Diagnostician/Systemizer are clear: to provide the Plant Engineer with an accurate and adequate model of the system from which optimal decisions can be made. Prescribing whether the DSA is a standalone component or implemented on top of an existing AGV World Model Knowledge Store is an unnecessary constraint that violates the generality and flexibility of the framework.

Communicator Analyst

The Communicator Analyst (CA) acts as the gateway between the CRMF Virtual Plant and other vehicle control and planning architectures such as JAUS or STANAG 4586. This analyst is responsible for importing mission specific parameters from which abstract goal states are defined. The reference implementation CA distills mission information from a set of Operating Behaviors to form a goal vocabulary, whose semantics are known beforehand. Furthermore, since current robotic architectures do not support highly autonomous resource management capabilities, it is beneficial that the analyst also support an interface through which external

architectures could query resource health information or other resource specific knowledge within the framework's purview.

System designers are tasked with drafting interface control documents defining implementation specific protocols depending on the nature of information required. Upon maturation of technology these protocols may be incorporated into framework communicator plug-ins, further alleviating integration pains. Plug-ins have been developed for interpreting some of the more common JAUS messages along with select Adaptive Planning Framework Metadata elements. The current emphasis within the robotics community to develop technology standards means that only a relatively small number of CRMF Communicator Analyst implementations are needed to achieve compatibility with a number of fielded platforms.

Lastly, the CA is responsible for maintaining the Human-Machine-Interface providing much desired operator supervisory control. The communicator would need to encapsulate relevant resource utilization statistics along with other data into the desired format for the operator console. Again these formats are typically specified by the standards bodies and could be incorporated in a plug-in fashion. The framework facilitates this functionality by granting the communicator analyst sufficient authority to access and share framework component knowledge and by providing a virtual tunnel through which the Plant Engineer can be accessed.

Plant Engineer

The Plant Engineer functions as the centralized resource broker for a given Virtual Plant. By utilizing the framework knowledge representation scheme, which assimilates knowledge from distributed disparate resources and applications, the Plant Engineer is able to craft allocation strategies that optimize mission performance. Intelligent operations pertaining to behavior generation, value judgment, and system modeling are made possible under the auspices of the PE.

Behavior generation is the end product or the culmination of a number of framework operations perceived as a sequence of effector operations that achieve an overall system goal. The Plant Engineer's role as resource broker plays a critical role in the behavior generation process. It's responsibilities as job arbiter and resource provisioner will dictate if, when, and how job requirements will be fulfilled. While other framework elements such as the DSA play auxiliary behavior generation roles, the principal burden lies with the Plant Engineer.

Value judgment is performed on multiple occasions during Plant Engineer operations. Due to platform bandwidth constraints limiting resource availability, it stands to reason that periods will exist in which the number of resource requests outnumber the resources themselves. The first level of value judgment might occur upon job submission. The Plant Engineer must deliberate over which operations are more mission-critical then others and then attempt to satisfy them accordingly. A second level of value judgment is needed when selecting from among a list of suitable candidate resources. In instances where "ties" or other anomalies arise, proper conflict resolution strategies should be formulated and applied. Algorithms for assigning value and conflict resolution measures must be defined according to framework implementation specifics and is tightly coupled to the attribute information encapsulated in the Knowledge Representation Scheme.

While the DSA is the principle entity tasked with maintaining the system model, the Plant Engineer, in its capacity as centralized framework decision broker, has influence over certain representative models. Designers may choose to implement a "job queue" structure to represent current resource requests. The PE must regularly access and manipulate this model structure when assigning value to new jobs and selecting the jobs to process. Another useful model parameter is a representation of the resource utilization schedule. This information is

needed to support advanced reservation, soft-state management, and to assess resource availability. As new assignments are made the PE accesses and updates this information accordingly.

Principal analyst interactions occur between the PE and the Resource Appraiser, Diagnostician/Systemizer, and the Communicator analysts. The communicator provides the PE with mission goal Meta-Knowledge that allocation algorithms can be tailored towards. The acquired goal state information allows the PE to prioritize inbound jobs through the application of reasoning methods. The Diagnostician/Systemizer provides an interface through which the PE can query an up-to-date model for resources which meet task requirements. PE and RAA collaborations include the notification of new job events. Further resource information such as availability, operating status, and reactivity is also made available by the RAA and updated in the system model.

Although the task of resource allocation is often viewed as a constraint satisfaction problem, the proposed framework does not subscribe to a particular algorithm or methodology for brokering resources. It is foreseeable that situations will arise in which either too many or too few resource candidates exist. This framework provides developers with sufficient information to implement heuristic weighting functions, rule-based methods, or some other variant for conflict resolution which can optimize performance based on the provided goal representation. The objective of the framework is not to provide roboticists with a "canned" solution, but to make available the tools necessary for crafting tailored solutions in a general and formalized fashion.

Having explored the Plant Engineer's involvement in the resource management process onboard autonomous ground vehicles, a more liberal definition using the phrase "resource management" is now applied. This phrase, which has been used extensively throughout this

work, conventionally refers to utilization of a sensing resource for a specific task. The framework, however, is not limited by this definition. The "resource" concept may be abstracted to include the entire vehicle platform, and the management of the platform's sensing capabilities may be achieved via selecting an appropriate vehicle location that provides an optimal vantage point. This scenario is indicative of typical patrol missions, an area of ongoing UGV research. Military programs such as REDCAR have sought to use these platforms as first response entities capable of relaying potentially threatening environmental changes to an operator. The operator can decide subsequent actions based on the information at his disposal. Many of these tasks are currently teleoperated but autonomous solutions are being pursued. The Cognitive Resource Management Framework was developed in direct support of such advanced autonomous capabilities.

Concept of Operation

Having completed a detailed discussion regarding the framework architecture, knowledge representation scheme, and intelligence elements, a more holistic overview of framework operations is now presented.

Operational Setting

The framework, although general in nature, is intended to function within the context of some intelligent system. The framework assumes the host system or platform adheres to a predefined and documented operating architecture. This architecture should provide mechanisms for transmitting and receiving data from other computing elements (transport-layer) and protocols and vocabulary governing the control of actuators and effectors (message-set, for example). These minimum capabilities grant the system a specific level of autonomy which it uses to issue commands which provide low-level resource control. Before cognitive resource management can take place the host environment must possess some representation of mission

goals or operating objectives which the framework must interpret before any custom allocation and brokering strategies can be developed.

An operational overview of the framework is now provided. A Resource Analyst, assigned to a hardware resource, initiates the discovery process by sending a Resource Object to the RAA. The DSA uses the knowledge attribute information contained within the Resource Object to develop a representative resource model. Subsequent Resource Objects sent to the RAA provide status and health updates used by both the RAA and the DSA. The Communicator Analyst provides mission and goal abstractions to the RAA and PE in the form of Meta-Knowledge. When a software application requires the utilization of a resource, the Application Analyst populates a Job Request with design-time-specified framework knowledge attributes. The RAA logs the request and the Plant Engineer, using DSA Meta-Knowledge, determines the value of a job. Highest value jobs are processed first. The PE queries the DSA for resource candidates that can fulfill the job and assigns it to the most suitable candidate. Job completion information is collected and analyzed by the RAA and can be used by the PE to modify conflict resolution or other resource allocation strategies. In this fashion the framework is able to achieve cognitive resource management.

Framework Environment

The framework provides a benign setting in which application and resource representatives can freely interact with the common goal of optimizing the effectiveness of the overall system. Participants in this forum can expect a level of trust in their relationships. Resource Analysts, who represent actuators and effectors, do not oversell their capabilities. Information disclosed within the resource objects are fair estimates of a resource's capability and performance attributes. Similarly the Application Analysts, who represent the software entities seeking the utilization of a resource, submit Job Requests which contain reasonable estimates of resource

utilization requirements. There are no incentives for cheating. In fact, the RAA evaluates the effectiveness of resources utilization for meeting software needs and can highlight any operating inefficiencies and has authority to regulate resource usage accordingly.

Intermediate and Executive layer architecture elements ensure that transactions between Resource and Application Analysts occur at arm's length. All parties act with their own best interest in mind and are not subject to pressure or duress by the other party. The Cognitive Resource Management Framework fosters a cooperative environment in which framework components exhibit a symbiotic relationship allowing all parties to operate with the common goal of realizing efficient system performance.

Information Transmission

Data contained within the framework knowledge representation formats is transmitted on an event-driven basis. An on-change event, for example, would minimize communication bandwidth by only initiating transmissions when new information is present. This of course introduces other implementation complications such as the need for guaranteed delivery or ACK/NAK. Implementing these mechanisms at the transport layer can be cumbersome and costly especially in a publish/subscribe model when multiple recipient connections must be managed. However, the ramifications of missing a single message can prove to be far more costly, as no new information will be received until another change occurs.

Data marshalling within the framework can operate in one of two ways. A decentralized messaging searching scheme, such as the previously mentioned publish/subscribe model, is favorable when intermediate and executive layer analysts are implemented across multiple computing hardware nodes. A subscription process must take place as each component comes online and each component is tasked with managing its collection of subscriptions. Once a subscription has been activated, the publisher sends information directly to the subscriber. This

approach requires that either *a priori* knowledge of publishers is known when creating the subscriber components or a discovery process must be created to allow subscribers to seek out the publishers.

A centralized messaging approach, such as that which is used with blackboards or knowledge stores, is preferred when intermediate and executive layer elements are implemented on a common computing node. Information is transmitted once directly from the source analyst to the Resource Appraiser Analyst. The Plant Engineer, along with Communicator and Diagnostician/Systemizer analysts can then access or copy the knowledge and manipulate it as needed. This approach requires less communication bandwidth than the decentralized messaging scheme and simplifies the delivery management system on the sender's side since the message has a single destination. A drawback of this approach lies in its associated performance tradeoff. The framework performance is limited by the speed of the centralized knowledge repository, which in this case is the Resource Appraiser Analyst. This approach is favorable if RAA update rates are sufficiently high and is utilized in the Reference Implementation.

Reasoning Mechanism

Framework operation is achieved through an asynchronous, iterative, event-driven, opportunistic reasoning mechanism for converting facts into Resource and Application attribute knowledge, which is deliberated upon to modify and execute allocation strategies which generate desired behaviors. The asynchronous and iterative approach facilitates the fusion of knowledge from framework elements, which may be spread over a number of computing nodes, by allowing them to operate at update rates commensurate with their duties and host hardware capabilities. The opportunistic reasoning model allows the framework to achieve optimal performance through the application forward and backward reasoning at different times in a manner that effectively satisfies resource management needs. Forward chaining reasoning is used when

deriving Resource Objects, Job Requests, and certain Meta-Knowledge elements, tasks where a set of facts are known but conclusions remain to be drawn from them. Backward chaining is appropriate within portions of the Plant Engineer and Diagnostician Systemizer Analyst when a desired capability is known and the resources needed to achieve it must be determined. For a given iteration cycle, framework elements examine inputs (whether facts or conclusions) and apply rules, algorithms, or other AI techniques in order to generate knowledge attributes or seek out suitable candidate resources which satisfy a goal condition. This approach requires a hybrid of iterative and event-driven operations to ensure the timely detection and dissemination of information while reducing the amount of redundant computation.

Control Strategy

The Cognitive Resource Management Framework utilizes a centralized control for managing resources and to support supervisory control. A soft-state-management approach is employed to mitigate resource starvation issues that might arise during resource sharing. Specific resource control is achieved either through delegating a component controller or by enabling the Plant Engineer to send specific commands directly to the resources.

The Cognitive Resource Management Framework adheres to a centralized management model. The key distinctions between the methodologies were discussed in the Control/Management Approaches section of Chapter 2. Many differences between centralized and distributed resource management approaches are highlighted by Figure 3-7. Centralization of the discovery process helps minimize communication bandwidth needed to broadcast discovery message traffic. By centrally locating system modeling, value judgment, and behavior generation operations, the need for redundant processing of and transmission of data across multiple machines is eliminated. Centralizing these activities takes the processing burden away from the resource and application sides of the framework.

A decentralized approach is illustrated on the top-right side of Figure 3-7 which shows the increased message traffic and redundant computation associated with dividing those responsibilities among the resource and the applications. The lower-right portion of Figure 3-7 depicts an alternative configuration where the Resource Analyst is responsible for modeling, monitoring, allocating, and scheduling resources. The increased computation load on these individual resources may prohibit integration on existing platforms by excessively taxing those resources, whose nodes were intended for simple I/O communication. This figure closely mimics market based approaches discussed in the literature [59] where Job Requests are sub-contracted out among resources, with no single resource possessing a complete representation of the overall operating picture. This approach requires increased message traffic, can lead to sub-optimal results, introduces more time-steps for disseminating information, and requires redundant computation at each resource. The centralized approach advocated by this framework simplifies management protocols, eases implementation, and facilitates data consistency, all with less message traffic than partially decentralized or fully decentralized approaches.

The Framework, through its architecture and centralized management policy, facilitates the implementation of supervisory or fully autonomous resource control. Chapter 2 discussed the need for operator intervention support mechanisms in AGV platforms. This dual control is realized through the Plant Engineer, which can exist either as a fully automated or human directed process. Engineers would need to design a user interface for relaying knowledge representation scheme elements to the operator. Overall system coordination would need to account for the increased decision time associated with a human operator. The design of such an interface is beyond the scope of this work; however, the framework provides the necessary hooks required for implementing either one or a combination of both styles.

Resource control is achieved by way of the Plant Engineer, which grants a software application permission to command a resource. This may be implemented in a number of ways. In one approach, such as the approach advocated in JAUS, the PE would utilize the Resource Analyst to instruct the resource to accept commands only from a controlling software application. In this approach all software applications would spam the resource with commands, but only commands from the PE designated controller would be accepted. An alternate approach would use the Application Analysts to relay resource control permissions. In this approach, applications would not be allowed to issue commands without the receiving prior authorization from the PE. In either case, a soft-state-management approach is employed to avoid deadlock and livelock concurrency-related issues. Under this approach, the Plant Engineer leases resource control for a specified time interval. Regardless of the method implemented, only the Plant Engineer has authority to delegate and interrupt component control.

Framework Tools

A collection of framework tools were designed to formally document the content for representing resource knowledge and the processes needed to achieve effective resource management. These tools outline the vocabulary, syntax and semantics that make up the resource management language. The following discussion presents the tools created to support system developers as they design and implement the Cognitive Resource Management Framework. This presentation references template tools provided at the end of this chapter. Readers are encouraged to review Appendices B and C, which contain the framework glossary of terms and populated worksheets developed for Reference Implementation discussed in Chapter 4.

Terminology

Vocabulary is the set of words that make up a language. Communication requires that information be encoded, transmitted, received, decoded, and understood. Understanding can only

occur when the information has been properly decoded and incorporated into the receiving entity's knowledge-representation. This holds true whether the entity in question is a machine or a human. The Cognitive Resource Management Framework toolset provides a Glossary of Terms used to unambiguously define relevant framework terminology. This reference standardizes the definition of key words or expressions within the framework operating context and facilitates understanding among system designers. This reference mitigates development errors attributed to ambiguous terminology. It can also be used as a training tool to help new developers get up to speed.

Knowledge Representation Tools

The framework provides a collection of templates designed to assist developers during the knowledge engineering (information abstraction) process. Use of these templates not only ensures that all framework knowledge is formally documented, but also certifies that knowledge attribute selection satisfies the information needs for resource management operations. Designing attributes in accordance with the worksheets ensures the judicious selection of information to represent framework knowledge. Once fully populated by the system architect, the template can be handed off to software developers who can use it as an implementation guide.

Resource Attribute Definition Worksheet

This template serves as a design tool, specifying a resource attribute which encapsulates resource specific information for use within the framework. The Resource Object framework knowledge representation is composed of a collection of Resource Attributes. Figure 3-8 shows the template used to define each resource attribute. The Resource Attribute Definition Worksheet achieves three principle objectives. The first objective is attribute identification. Each attribute is given a unique name, which is then included in the glossary of terms with a synopsis of its

contents. Included in the worksheet are fields for identifying the parent or host resource along with the platform it is designed for; information that will prove useful come implementation time. The second objective of the worksheet is to describe the purpose of the attribute and some justification for its need. Designers are asked to include a description of the attribute along the intended use. Because Resource attributes are intended to support the framework's representative resource model located at the DSA, a brief description of the model should be included to provide the necessary application context information. The final objective of this knowledge engineering tool is to provide an explanation of how the attribute information is provided. This explanation includes the expected frequency with which information is updated, or if event-based, what action or algorithm acts as the trigger. The final portion of the template describes how the knowledge is to be encoded.

Job Request Attribute Definition Worksheet

This template, depicted in Figure 3-9, is included for the specification of new attributes used to describe a task within the Cognitive Resource Management Framework. Like their Resource Attribute template counterparts, this tool is designed to identify the attribute, justify its need and describe its purpose, and give an explanation of how the information is provided. The worksheet contains an entry for identifying the host software application name. The data contents field lets designers select from standard programming language data types such as char, int, float, double, string, or allows for the specification of proprietary data representations. Should attribute information be relayed as an enumeration, the correlation between the enumeration and its operational significance must be defined in the provided mapping table. Any algorithms or rules applied in discerning facts from a host application data representation should also be captured in the table next to its representation. For complex, custom data types a schema or other illustration should be included.

Meta-Knowledge Element Definition Worksheet

This template defines a Meta-Knowledge Element which is utilized by intermediate and executive layer framework components to support resource management endeavors. Figure 3-10 shows a structure of this template. Like the Resource Attribute Definition sheet, information vital to the identification, use, and implementation of a Meta-Knowledge Element is presented. The template details which framework analyst is responsible for producing the data, which analyst(s) would use it, and how the data is to be represented.

Framework Component Tools

The framework provides a collection of design tools which upon utilization provide a cohesive and coherent overview of framework operation and implementation specifics. These tools provide a roadmap depicting the integration of all previously defined framework Knowledge Representation Scheme elements.

Resource Description Overview Worksheet

The worksheet, illustrated in Figure 3-11, describes a specific Resource Object used for representing a hardware resource within the Cognitive Resource Management Framework. The "Class Name" field denotes contains a name that unambiguously identifies the Resource Object. Inheritance information is conveyed within the "Parent Class" field and is visually depicted by an illustration showing the extensible hierarchy of Resource Objects. The "Class Description" field describes the new functionality and information conveyed within this newly specified Resource Object. Because Resource Object directly supports the framework's representative resource model located at the DSA, a brief description of the model is again included to provide the necessary application context information. In addition, the worksheet provides a table which maps the specific resources types which are derived using this Resource Object. The final and perhaps most critical function of this template is located on the second page of the Resource

Object Description Worksheet. A schema is included which lists the Resource Attributes along with their associated data types. This worksheet is the capstone which unifies the resource attribute knowledge engineering process. This tool highlights all critical resource knowledge elements needed for managing and modeling a resource and should be checked for cohesiveness, ensuring that all defined Resource Attributes are in fact utilized in the representation.

Job Request Definition Worksheet

This worksheet, presented in Figure 3-12, provides an overview of the elements that constitute a specific CRMF Job Request. The "Job Type" field uniquely identifies the job. The "Vehicle Identifier" entry is used to implicitly identify the resource and platform through which this job is to be fulfilled. The "Setting Overview" provides a textual description of the operating environment in which this job is expected to occur. This field includes any assumptions or conditions which are expected during operation. Subsequent template entries of Resource Specification, Advanced Reservation, and Target Type describe specific resource brokering modeling activities which must be supported by the Plant Engineer and Diagnostician/Systemizer Analyst. Lastly the sheet identifies, in both tabular and schematic form, the collection of attributes that must be included for completeness when creating a Job Request of this type.

Resource Appraiser Analyst Performance Monitoring Worksheet

This tool identifies the areas of system performance that are monitored and evaluated under the Cognitive Resource Management Framework. The overview entry, shown in Figure 3-13, should contain a narrative summarizing the performance monitoring roles supported by this Resource Appraiser Analyst implementation. Pertinent information might include whether the RAA has influence in the PE deliberation process and to what extent. The principle role of this tool is to catalog the entries that appear within the system log and the framework analysts

responsible for providing such information. Auxiliary roles include defining a format for the log file along with a sample output, and identifying the performance metrics which are computed based on the accumulated data. Each performance metric is listed by name in a table along with the algorithms used to calculate it.

Communicator Analyst Worksheet

The Communicator Analyst template serves as the interface control document identifying the various interactions between the Cognitive Resource Management Framework and other system architectures. Like many other worksheets already discussed, this sheet outlines implementation-specific input and output parameters. The overview section provides a high-level description of what type of functionalities are supported on either side of this interface. The remainder of the worksheet, depicted in Figure 3-14, provides a venue where designers can identify and organize items involved in the information exchange. This worksheet can be of great utility. It can be used during design time brainstorming sessions to quickly identify which parameters may be needed and the hurdles that must be overcome to incorporate them. This sheet then serves as a checklist during implementation, tracking which elements have yet to be incorporated. Once the system reaches prototype phase, this populated worksheet provides documentation for engineers performing field testing.

Diagnostician Systemizer Analyst Worksheet

The DSA worksheet, shown in Figure 3-15, provides an overview of framework supported modeling capabilities. A narrative description of model performance and functionality should be included in the "DSA Model Description" field. This description should concisely describe the roles the modeling analyst plays in the framework implementation. The form contains fields for specifying input parameters in the form of framework specified knowledge elements along with the entity charged with providing such information. The output field provides a narrative of the

types of framework Meta-Knowledge that is produced, which should be documented separately in a Meta-Knowledge Definition Worksheet. In addition to describing model outputs, this tool includes a "Model Detail" section for outlining the algorithms utilized in modeling along with any and all relevant assumptions. The final element of the worksheet allows designers to append any illustrations that may help convey the DSA's functionality. This could include a graphical representation of coverage, or perhaps the envisioned output of a DSA visualizer to assist implementations requiring supervisory control.

Plant Engineer Worksheet

This tool outlines the brokering strategy and acts as an interface control document for the Plant Engineer framework element. The overview field, shown in Figure 3-16, contains a description of the overall operation and strategy that is employed at the executive layer of the framework. It provides an overview of the operations at the PE level that enable resource management. The "Model Inputs" field is used to catalog component inputs and their respective authors. The "Outputs" section outlines how jobs are handled after allocation. Protocol execution specifies the behaviors which occur as a result of the PE deliberation and execution cycle. The "Illustrations" segment of the worksheet provides a venue for showcasing flow charts, diagrams, or other images that capture the overall concept of operation for the framework implementation.

The aforementioned framework tools are of great utility during the design, implementation, integration, and test phases of the engineering life cycle. They can be used during design time brainstorming sessions to quickly identify which parameters may be needed and the hurdles that must be overcome to incorporate them. These forms then serve as a checklist during implementation, tracking which elements have yet to be incorporated. Once the system reaches prototype phase, each populated worksheet provides documentation for engineers performing field testing. These design tools, which upon utilization, provide a cohesive and coherent

overview of framework operation and implementation specifics. During the production phase these tools provide a roadmap depicting the integration of all previously defined framework knowledge representation scheme elements.

-

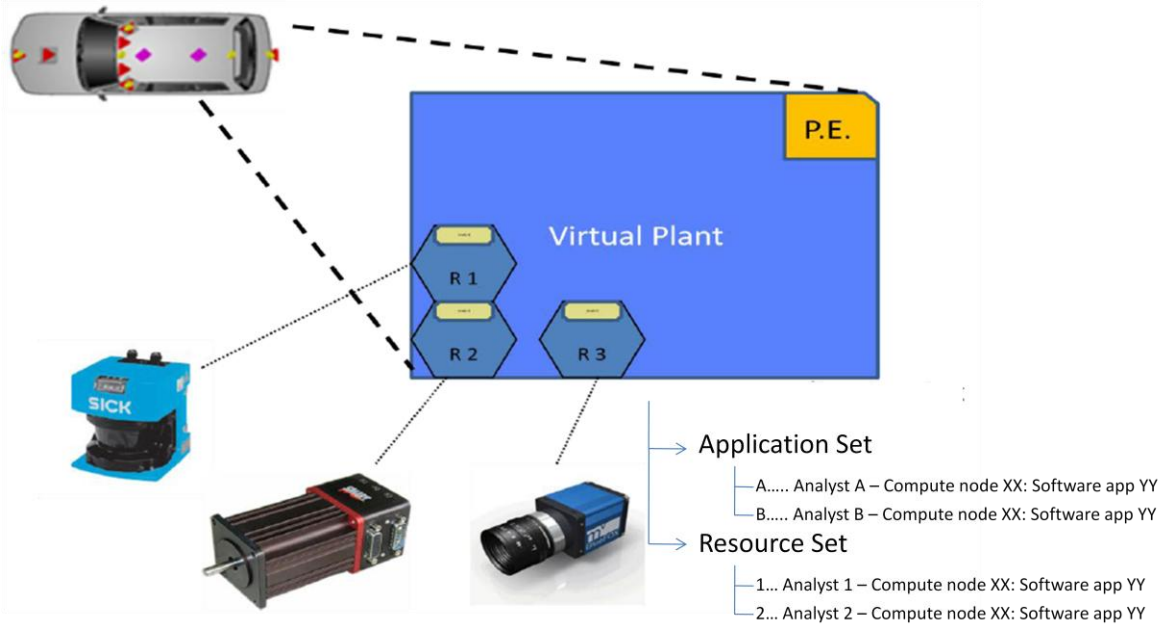


Figure 3-1 CRMF Logical Organization of AGV Resources as a Virtual Plant

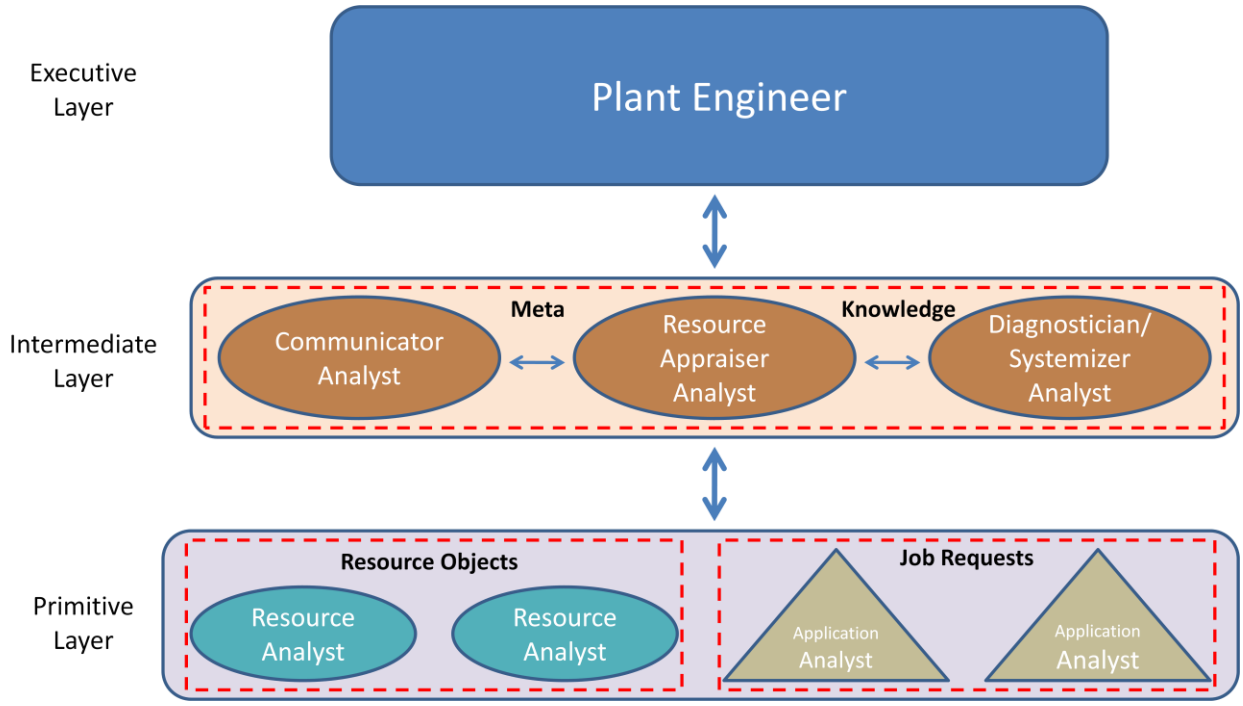


Figure 3-2 CRMF hierarchical architecture and information pathways

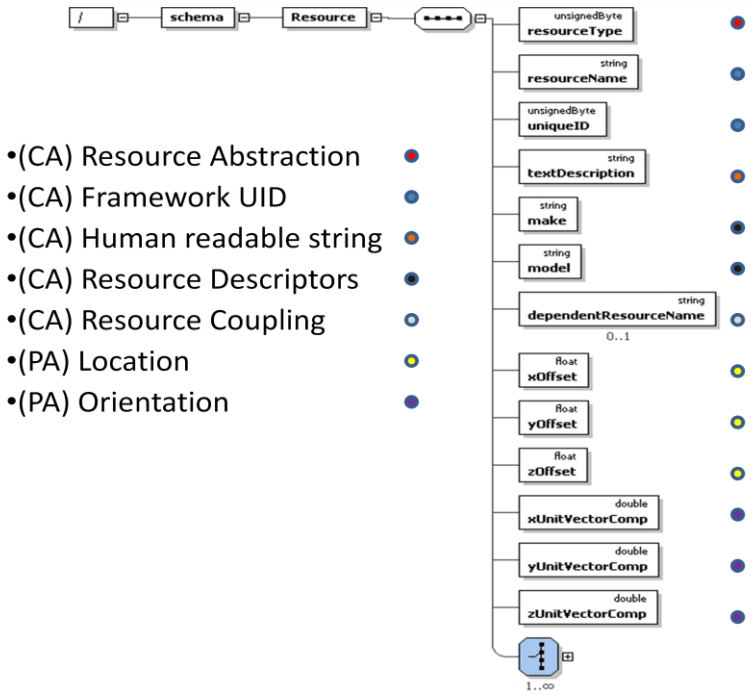


Figure 3-3 Resource base class Capability and Performance Attributes example

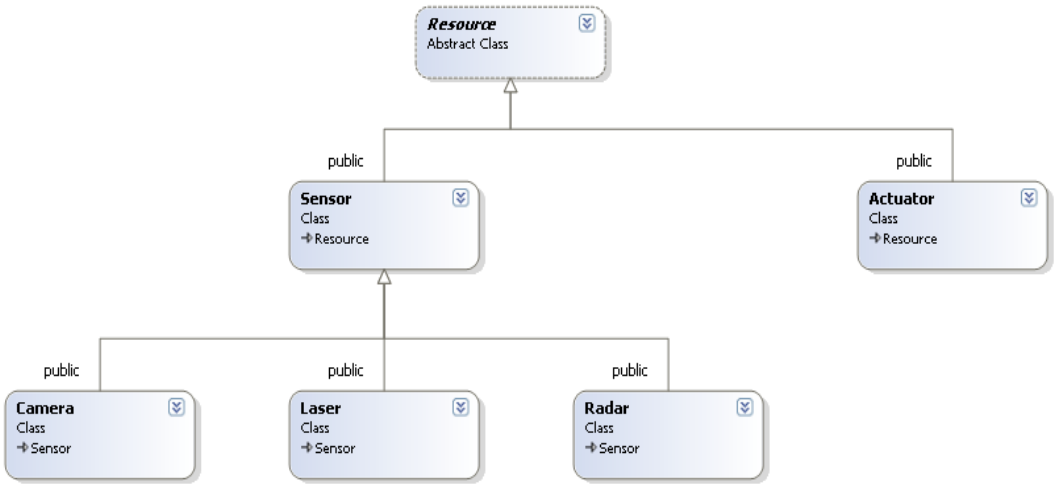


Figure 3-4 CRMF Resource Object

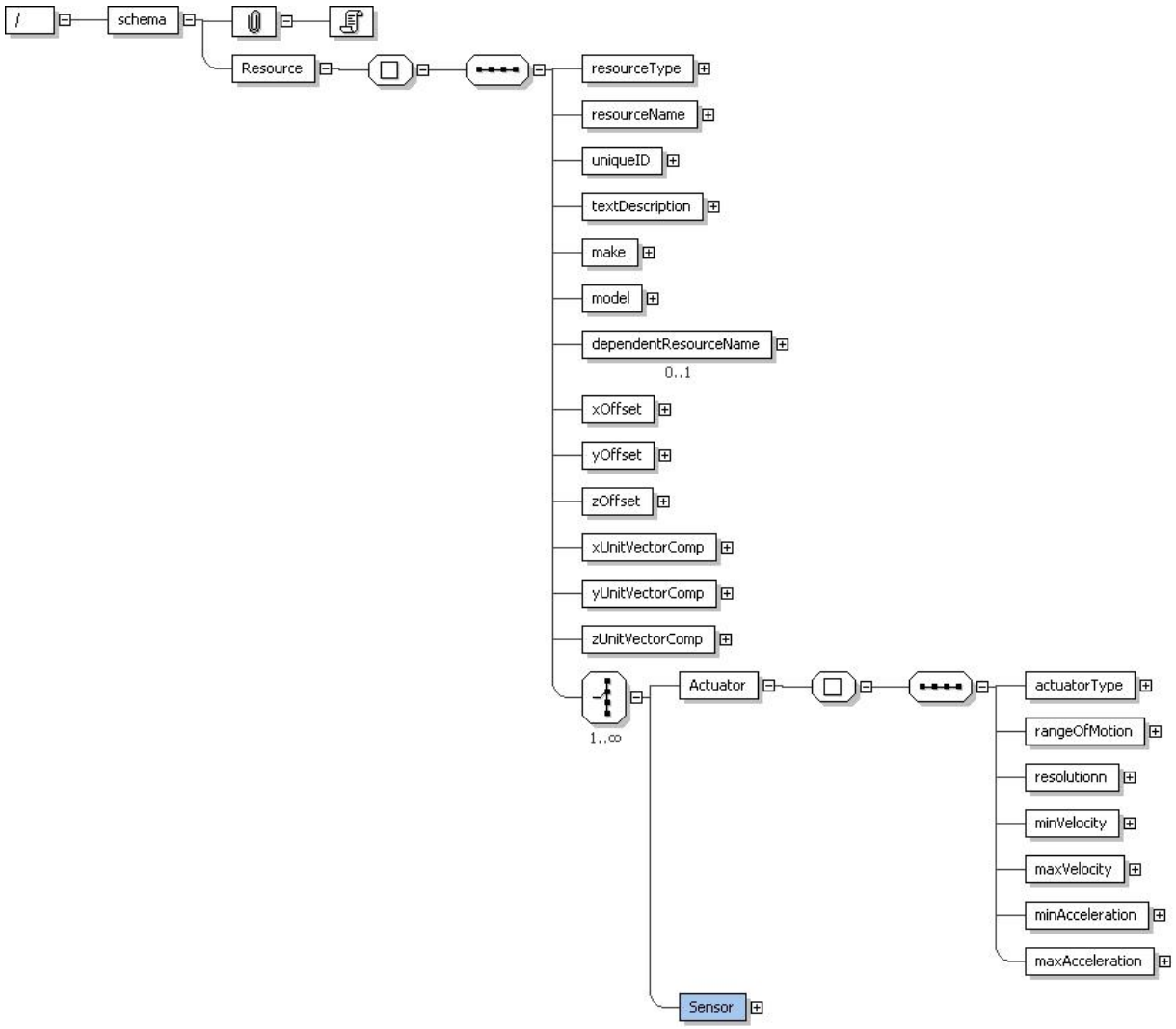


Figure 3-5 XML Resource Object schema with full description of SDOF actuator

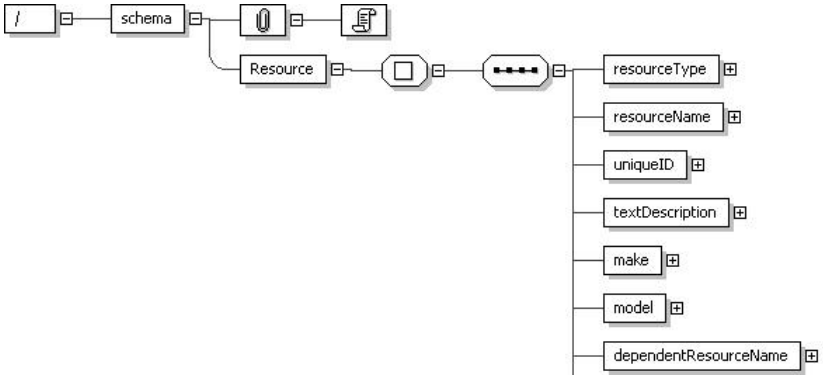


Figure 3-6 XML Resource Object schema for SDOF actuator identification only

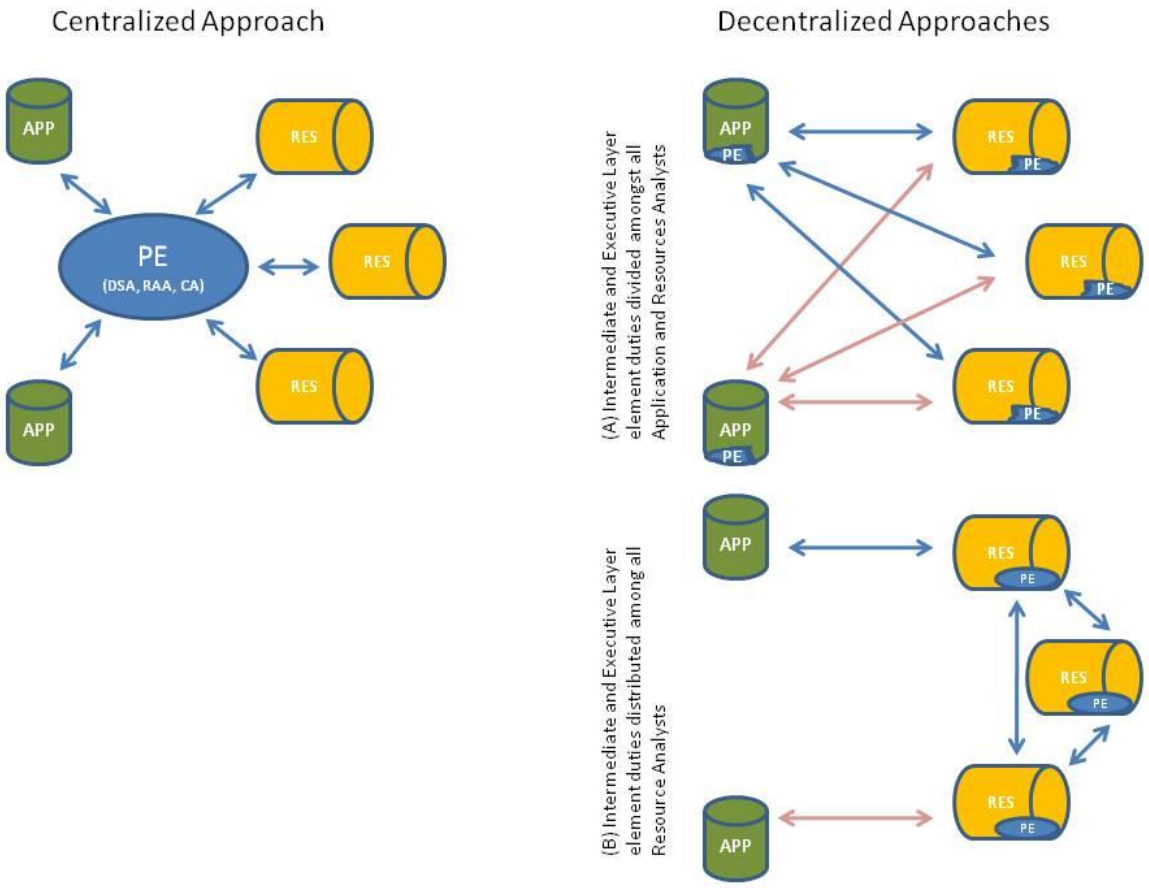


Figure 3-7 Communication pathways and functionality for different resource management approaches

Resource Attribute Definition Worksheet

This worksheet is an engineering design tool for specifying a resource attribute that describes resource capability within the Cognitive Resource Management Framework.

Attribute Name:

Attribute Type: capability/performance Type: _____

Parent Resource Name:

Attribute Description:

Intended Use:

Vehicle Identifier:

DSA Model description: 3D spatiotemporal model. See DSA worksheet for more details

Update Frequency: Rate/Event **Rate:** ___ Hz

(if event)

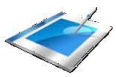
Event Trigger	**Description**

Data Contents: Value/Enumeration/String/Other Type: _____

(if enumeration: populate apriori mapping of values)

Enumeration	Interpretation/Significance

(if other: include schema or other data structure illustration)



Rev b. updated 1/14/10

Figure 3-8 Resource Attribute Definition Template

Job Request Attribute Definition Worksheet

This worksheet is an engineering design tool for specifying a new Job Request attribute that describes a task within the Cognitive Resource Management Framework.

Attribute Name:

Attribute Type: Capability Request/Target Attribute Type: _____

Parent Analyst Name:

Host Application Name:

Attribute Description:

Intended Use:

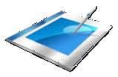
Vehicle Identifier:

Data Contents: Value/Enumeration/String/Other Type: _____

(if enumeration: populate apriori mapping of values)

Enumeration	Interpretation/Significance	Algorithm(s)/Rule(s)

(if other: include schema or other data structure illustration)



Rev a. updated 1/13/10

Figure 3-9 Job Request Attribute Definition Template

Meta-Knowledge Element Definition Worksheet

This worksheet is an engineering design tool for defining Meta-knowledge elements which support resource management initiatives within the Cognitive Resource Management Framework.

Element Name:

Parent Analyst Name:

Element Description:

Intended Use: (which framework functions does it support)

Vehicle Identifier:

Update Frequency: Rate/Event **Rate:** ___ Hz

(if event)

Event Trigger	**Description**

Data Contents: Value/Enumeration/String/Other Type: _____

(if enumeration: populate apriori mapping of values)

Enumeration	**Interpretation/Significance**	**Algorithm(s)/Rule(s)**

(if other: include schema or other data structure illustration)

Rev a. updated 1/13/10

Figure 3-10 Meta-Knowledge Element Definition Worksheet

Resource Object Description Worksheet

This worksheet serves as an engineering design tool for identifying a set of resource attributes for describing a specific resource's capabilities.

Class Name:

Parent Class:

Class Description:

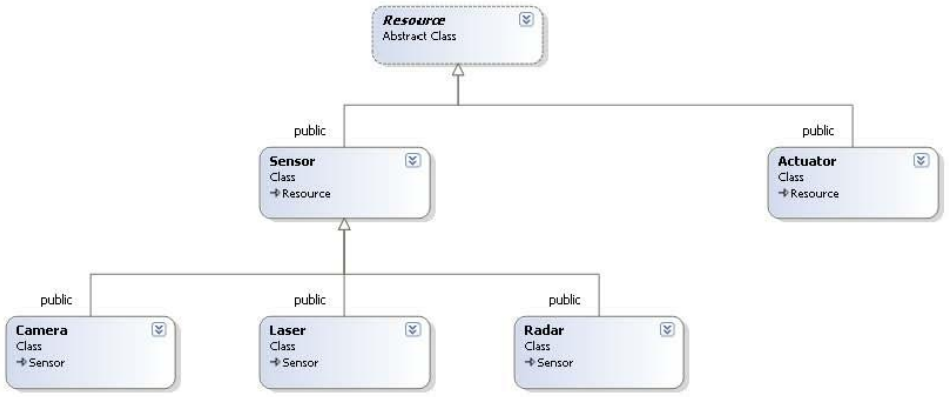
Vehicle Identifier:

DSA Model: (Describes the type of model to be used.. in other words what parameters must a resource disclose in order for the DSA to model it's capabilities

Derived Resources Supported:

Type	Trace
Linear, Rotary	Resource->Actuator
Camera	Resource->Sensor
...	...

Illustrations:

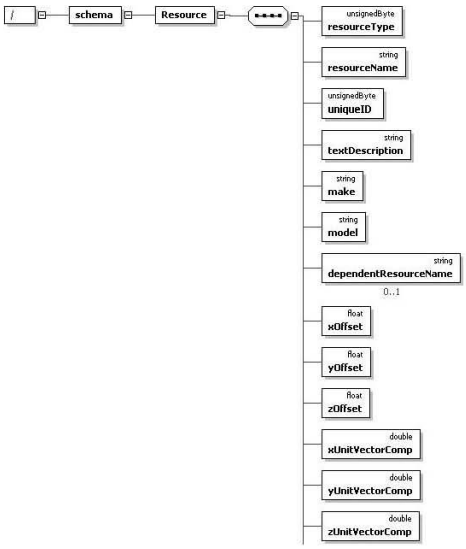


Rev b. updated 11/5/09

Figure 3-11 Resource Object Description Template

Resource Object Description Worksheet

XML Schema:



Rev b. updated 11/5/09

Figure 3-11 Continued.

Job Request Definition Worksheet

Note: This worksheet is an engineering design tool for defining the elements that comprise a job request utilized by the Cognitive Resource Management Framework.

Vehicle Identifier:

Setting Overview:

Job Type:

Resource Specification: Supported/Not Supported

Advanced Reservation: Supported/Not Supported

Target Type: point/ area/ other

Attributes Name	Attribute Type
List attributes here	Capability Request/Target
…	…

XML Schema:

Rev b. updated 1/13/10

Figure 3-12 Job Request Definition Template

Resource Appraiser Analyst Performance Monitoring Worksheet

Note: This worksheet is an engineering design tool identifying areas of system performance evaluation monitored by the Cognitive Resource Management Framework Resource Appraiser Analyst.

Vehicle Identifier:

Overview:

Inputs:

Input	Source
Input Name	Analyst Name
...	...

Outputs type:

Output Format: (Sample Output)

Parameter Analysis: Real-time/Offline

Performance Metric	Algorithm(s)

Rev c. updated 1/14/10

Figure 3-13 Resource Appraiser Analyst Performance Monitoring Template

Communicator Analyst Worksheet

Note: This worksheet is an engineering design tool outlining interactions between the Cognitive Resource Management Framework and external system architectures.

Vehicle Identifier:

Overview:

Parameters to import:

Parameter	Source	Format	Use

Parameters to Export:

Parameter	Destination	Format

Rev b. updated 11/5/09

Figure 3-14 Communicator Analyst Information Control Template

Diagnostician Systemizer Analyst Worksheet

Note: This worksheet is an engineering design tool for identifying the characteristics used by the Cognitive Resource Management Framework for modeling the virtual plant capabilities.

Vehicle Identifier: Vehicle Name

DSA Model Description: Describes the type of model to be used, e.g. spatial model, geospatial

Model Inputs:

Input	Source
Resource attributes	Indentify analyst responsible for producing data (separate Resource Worksheets must be completed for each)
...	...
Job Request	Indentify analyst responsible for producing request (separate Job Request and Analyst Worksheets must be completed)

Outputs: Describes the end product of a model query indicating the data that can be used by the PE

Model Details:

Algorithm	Assumptions

Illustrations: (Optional) Include visual aides to help portray the model concept

Rev b. updated 11/5/09

Figure 3-15 Diagnostician/Systemizer Analyst Template

Plant Engineer Worksheet

Note: This worksheet is an engineering design tool for defining the brokering strategy utilized by the Cognitive Resource Management Framework Plant Engineer.

Vehicle Identifier:

Overview:

Model Inputs:

Input	Source

Outputs:

Protocol Execution:

Algorithm	Assumptions

Illustrations:

Rev b. updated 11/5/09

Figure 3-16 Plant Engineer Template

CHAPTER FOUR
REFERENCE IMPLEMENTATION AND FIELD TESTING

This chapter chronicles the implementation of the Cognitive Resource Management Framework as a core functional element of CIMAR's Environmental and Mapping and Change Detection technology demonstration for the Air Force Research Laboratory. The framework was integrated into CIMAR's existing JAUS-based implementation deployed in the DARPA Urban Challenge. It provides supplemental intelligent resource management capabilities necessary for meeting the stringent requirements of autonomous reconnaissance, surveillance, and target acquisition activities. This exercise serves as proof of concept of the framework as presented in Chapter 3.

Reference Implementation Architecture and Design

RSTA operations, whether fully automated or with supervised autonomy, involve the integration of autonomous navigation and surveillance capabilities onboard a single mobile robot platform. The goal of the reference implementation is to demonstrate how the CRMF enhances the autonomous navigation capabilities set developed for the Urban Challenge to simultaneously support detailed environmental mapping and monitoring while exhibiting safe and effective navigation.

The operating scenario is as follows. An AGV with *a priori* road network knowledge and a predefined patrol region is deployed in an environment about which it has no other information. Utilizing only onboard sensing, perception, and reasoning capabilities the robot will survey the environment and develop a world model representation of all objects, structures, and entities detected to be maintained within a knowledge store. Upon successful characterization of the operating environment the platform commences execution of the patrol mission in which it

searches the region looking for perturbances. Appendix A provides a detailed overview of the research setting in which this work was performed.

As implemented, the Cognitive Resource Management Framework oversees an array of MATRIX Vision BlueFox cameras mounted on a rotary stepper actuator used to capture image information needed for object classification and characterization. The reference implementation resource virtualization graphic is shown in Figure 4-1. The remainder of this section describes the implementation and functionality of the framework elements as defined in Chapter 3 to satisfy the requirements of the Environmental Mapping and Change Detection demonstration.

Primitive Layer Analysts Defined

The Primitive Layer analysts are tasked with distilling sensory level signal data into specific framework knowledge attributes which are then utilized to meet the information needs of higher level framework elements. The Reference Implementation architecture design prescribes three Resource Analysts and an Application Analyst to satisfy these requirements. They are described in further detail below.

Resource Analyst

The Resource Analysts provide a generalized abstraction of AGV hardware resources and serve as the bridge between the framework and the hardware. The framework requires a Resource Object definition that encapsulates essential Capability and Performance Attributes. The implemented Resource Object extensible class structure is depicted in Figure C-5. All resources are derived from an abstract "Resource" which contains Capability and Performance Attributes common among all derived resource types. The Sensor and Actuator classes inherit from this abstract base class. While the reference implementation necessitated the design of three Resource Analysts and their associated Resource Object representations, the framework also includes representations for the most common AGV sensors and actuators.

The first analyst was tasked with representing the rotary stepper motor actuator. Its Resource Object architecture is depicted in the schema found in Figure C-5. The actuator contains a Capability Attribute denoting whether the actuator is of the linear or rotary type. This distinction allows the model to determine how the resources line-of-action, specified by the Z-axis unit vector, effects motion. A number of Performance Attributes follow which characterize resource operation in terms of specified performance metrics. These attributes provide the DSA with the operational parameters necessary to model a resource's performance characteristics. As implemented these metrics are hard coded and derived at design time or after suitable field testing. Alternate implementations could dynamically update these attributes based on local intelligence such as resource health.

The two other Resource Analysts were charged with producing Resource Objects for the camera sensors affixed to the panning mechanism. The Sensor class contains a generalized "sensorType" Capability Attribute field for identifying which of the specific classification the resource falls into. Other Performance Attributes such as field of view, measurement range, update rate, and reconfiguration time are used by the model to quantify the sensor's performance. A Camera class, derived from the Sensor class, was designed to incorporate the "camerType" Capability Attribute, along with Performance Attributes disclosing view angle information. By applying this knowledge, the DSA uses a rule based approach to match a Job Request to a resource's current capabilities. The specifics regarding how the model utilizes framework knowledge are addressed in the Diagnostician/Systemizer section.

All resource hardware was connected to a single computer on the Urban NaviGATOR. Moreover, the software interface to the hardware was implemented using a single experimental JAUS component built using CIMAR's proprietary implementation. The three Resource

Analysts were implemented as distinct and independent routines operating within that single component. This implementation highlights the versatility and interoperability of the framework when integrating on existing systems. Hardware connectivity does not impact the design of the resource analysts, merely where their functionality is hosted. Development and integration are further expedited when adding redundant resource types to a system, see Figure 4-2. Once the initial Resource Analyst design has been completed; there is, therefore, no additional development cost associated with changing algorithms to accommodate the new resource. Engineers simply need to redeploy the existing RA adjusted to the new hardware resource. The framework will discover the resource and automatically manage its use according to protocols already supported by the framework.

Application Analyst

Application Analysts can be thought of as a Resource Analyst's software counterpart. These analysts form a link between the framework and the AGV software applications providing an abstract representation of desired tasks which require the utilization of resources. The reference implementation calls for the design of an Application Analyst hosted on a standalone experimental JAUS component. This component must ensure that a photograph is taken of every new object and then committed to the World Model Knowledge Store as object metadata. This Application Analyst is tasked with formulating a Job Request every time an "object of interest" has been detected. The framework will take in the request, assess its priority, and fulfill the request at an appropriate time. Once it is time for job execution, the Plant Engineer leases control authority over the necessary resources to the software application responsible for acquiring the data.

Application Analysts exist in a symbiotic relationship with their host application. A single AA is needed for each software application; these analysts can submit multiple job

requests of differing types depending on the task to be performed. For the purposes of the reference implementation, a single “imageCapture” job type is utilized. Figure C-6 shows the elements included in the implemented Job Request knowledge representation. A number of Capability Request and Target Attributes are defined and now discussed.

The jobNumber field provides a unique identifier for each task to be performed. This parameter is set by the Plant Engineer and utilized throughout the framework to reference this job. The jobType field is provided as an input to the PE and DSA deliberation processes and is used to assess the priority of the request and for correlating a job to a set of candidate resources. The resourceID and resourceName attributes support the explicit tasking of a resource, should the system design require using a specific resource for a given task. The remaining Capability Attributes are included to provide support for advanced reservation of resources.

Intermediate Layer Analysts Defined

At the intermediate architecture layer, the Diagnostician/Systemizer Analyst, Communicator Analyst, and Resource Appraiser Analyst assimilate all accumulated knowledge into a higher level more abstract representations, to be used by the Plant Engineer when fulfilling its duties. Their implementation details are now discussed.

Diagnostician/Systemizer Analyst

The Diagnostician/Systemizer is tasked with maintaining an up-to-date system model that the Plant Engineer can query during periods of deliberation. The development of a real-time 3-D spatiotemporal sensor model is a significant contribution of this research.

The first step towards developing the model requires creating a suitable approximation of a resource’s coverage. An analysis of common senor technology was performed and a list of suitable modeling parameters was developed. These parameters, encapsulated within the framework Resource Object, were then used to construct a 3-D geometric representation of a

sensor's field of view. Figure 4-3 illustrates the approximation parameters utilized for a rectilinear camera lens. View angle information is reported in hardware specification sheets or can be determined empirically. For every incremental distance away from the lens an observable vision plane can be constructed. The DSA approximates a camera's coverage volume by computing the coordinates of plane at the minimum observable distance and again at the maximum observable distance. These planes are then "stitched" together to form a frustum, which acts as the geometric representation for the sensor's coverage; this process is illustrated in Figure 4-4.

The SICK LADAR sensor, used heavily by autonomous vehicles, emits a plane of laser strikes and reports ranging information based on time-of-flight. Figure 4-5 illustrates this functionality. Critical parameters including minimum and maximum range distance, field-of-view information, and resolution between strikes are all reported in a Laser sensor Resource Object. The DSA models the pie-slice coverage area using a trapezoidal approximation using curve chords as seen in Figure 4-6. In addition to providing range information, some models also support measuring the reflectivity of a particular object. The Laser Resource Object discloses both the sensor's range and reflectivity measurement capabilities.

Fixed antenna radar sensors such as the Eaton Vorad radar are used in the commercial trucking industry for adaptive cruise control. Radar sensor coverage is typically characterized according to effective distance and beam width as shown in Figure 4-7. Beam width is the angular distance between half power points passing through the center of the beam centerline. The 3-D geometric representation of radar coverage is best fit by a conical representation. To simplify computational requirements, the DSA assumes a convex hexahedron, or frustum, geometric representation. Figure 4-8 presents a volumetric analysis of this representation. The

analysis shows that this assumption over-estimates the observable volume of the sensor by approximately 20 percent. It should be noted that the "sweet spot" of the sensor is fully encapsulated and that the traditional "conical" representation of the radar's coverage is itself an approximation. Thusly, the actual frustum approximation is more accurate than the often sought after "80 percent solution" shown by the purely volumetric analysis performed here.

The DSA must also account for tightly coupled resource, i.e. a sensor actuator pairing. The geometric representation of a sensor's coverage must include all regions which the sensor can observe under articulation. The DSA discretizes the actuator articulation into a fixed number of steps depending on the actuators range of motion. Figure 4-9 shows this process. Using this 3-D spatial representation of resource coverage, the Diagnostician/Systemizer Analyst must determine if a georeferenced point or polygon, specified in LLA, lies within a given sensors coverage polygon. The DSA then runs a point-in-polygon algorithm, frequently used in computer graphics applications, using a 2-D projection of the resource coverage area and target point in the vehicle XY plane. The point in polygon test is depicted in Figure 4-10. If the number of times the ray intersects the line segments making up the polygon is even then the point is outside the polygon. Whereas if the number of intersections is odd then the point (x_p,y_p) lies inside the polygon. If this test passes, the DSA runs the algorithm again using geometric projections in the vehicle XZ plane. This process of breaking the 3-D problem into 2 planar problems is efficient in that the second plane is only evaluated in the event that the first plane condition is true. This process is summarized in Figure 4-11 and in the following:

1. Create a model of sensor cover in the vehicle-referenced fixed coordinate system
2. Convert the desired point into the vehicle fixed coordinated
3. Create XY plane projection and test
4. If 3 passed, create XZ plane projection and test, else return false
5. If 4 passed, return true

In instances where sensor articulation is necessary the DSA temporal model uses actuator velocity performance attributes to provide an estimate of time till ready. At present, the temporal model provides a worst case estimate based on maximum articulation distance at minimum velocity. This ensures that scheduled jobs can be completed in the allotted resource lease time.

Before any spatial or temporal models are created and evaluated, the DSA employs a rule-based approach to matching Job Request Capability Attributes to discovered resource types. The DSA maps these job types to a set of candidate resource types, then in an iterative fashion creates and evaluates the spatial model and computes the temporal information if necessary. This information is encapsulated in a Candidate Resource List Meta-Knowledge element which is evaluated by the Plant Engineer when selecting the "best" resource to complete a job.

Communicator Analyst

The Communicator Analyst acts as the gateway between the CRMF Virtual Plant and other vehicle control and planning architectures. The reference implementation necessitates a CA capable of interfacing with both the Urban NaviGATOR's JAUS and Adaptive Planning Framework implementations. As implemented, the JAUS/APF CA plug-in exists as a separate class which can be swapped out or supplemented as needed to support different interfaces.

Communicator Analyst/Adaptive Planning Framework interactions include the receiving and decoding along with the encoding and publication of APF Metadata. Elements such as Vehicle Operating Behavior, Vehicle Operating Sub-Behavior, and Vehicle Operating Sub-Behavior Status provide notions of mission type and operating guidelines. This information is converted into CRMF Meta-Knowledge utilized by the Plant Engineer when prioritizing jobs and selecting the most appropriate resource from the candidate pool. While currently not supported by either the APF or CRMF, future implementation could provide environmental information such as luminosity, which may be used by the PE to choose between a night-vision camera and a

conventional camera. On the export side, the Communicator Analyst supports a Loitering Behavior Specialist suitability recommendation. Should the Plant Engineer determine there is an appreciable back-up of jobs in the queue, it is the responsibility of the Communicator Analyst to publish the recommendation for a loitering behavior to allow the system to "catch-up". Ultimately the decision of whether to accept this recommendation and change the vehicle operating behavior lies outside CRMF, relying on the APF Decision Broker and JAUS components to implement the behavior.

Communicator Analyst/JAUS interactions were implemented on an as needed basis. At present, this interface supports the requesting and receipt of a GPOS service connection, which is used to supply the DSA with globally referenced vehicle position and orientation in terms of latitude, longitude, roll, pitch, and yaw. In addition, the CA supports experimental JAUS/AS-4 messages for communicating with the World Model Knowledge Store. This analyst is responsible for creating change event notifications for every object that is currently in the queue. Upon receipt of one such notification, the Plant Engineer must update the object attributes contained in the Job Request and re-assess the priority of the task. The implementation of a complete interface for all JAUS messages is beyond the scope of this work. However, as the CRMF gains acceptance and use within the robotics community the development of a complete JAUS interface is a realistic possibility.

Resource Appraiser Analyst

The Resource Appraiser Analyst is responsible for overseeing the collection of framework data for real-time and offline analysis. This analyst's primary role concerns system monitoring and performance management; as such is involved with every facet of framework operations. The DSA records critical performance management information regarding resource utilization time, requests for capabilities, and job completion data. This information is logged as

a tab delimited text file using the output format specified in the Resource Appraiser Analyst Worksheet shown in Figure C-7. To facilitate the post-processing of information an Excel spreadsheet, seen in Figure 4-12, was developed. The workbook parses the RAA log file, generates necessary plots, and compiles a statistical overview of system performance metrics.

The current RAA implementation catalogs the receipt of Resource Objects during resource discovery and during change event updates. It also records job status updates from the various framework Resource Analysts. Job Request submissions along with Job assignment information is recorded from the Application Analysts and PE respectively. Current computed performance metrics include tallies of jobs received, assigned, successfully completed jobs, aborted jobs, removed jobs, job completion time, and DSA processing time.

There exists and important distinction between aborted jobs and removed jobs. Aborted jobs are ones that could not be completed in the allotted time possibly due to hardware failure or an insufficient resource lease time. Removed jobs are jobs in which a confirmation status message was not received. The status of these jobs is uncertain; they may have been completed successfully but the message relaying this information was never received. This distinction is just one example of the many unforeseen issues that arose during implementation time that resulted in framework modifications. Chapter Five provides more insight into implementation issues along with some concluding remarks.

Executive Layer Defined

The Executive Layer, which contains the Plant Engineer, is responsible for utilizing all accumulated framework knowledge from lower-layer analysts when performing value judgment and behavior generating activities involving the allocating and provisioning framework-discovered resources.

Plant Engineer

The Plant Engineer functions as the centralized resource broker for a given Virtual Plant. By utilizing the framework knowledge representation scheme, which assimilates knowledge from distributed disparate resources and applications, the Plant Engineer is able to craft allocation strategies that optimize mission performance. Application Analysts submit job requests to the Plant Engineer, which are logged by the DSA. The PE reviews the job request, assigns a value to the job, and places in a Job Queue. The CRMF job submission process is diagrammatically depicted in Figure C-10.

Value judgment is affected within the Plant Engineer during the job submission process and when the terms of a Job Request are updated. The PE stores all pending Job Request in a Job Queue structure of type deque. This queue is sorted according to the PE-assigned Job Value parameter. Job Value is assed based on the algorithm shown in equation 1.

$$Job\ Value = K_{situation} * \alpha + K_{distance} * \beta + K_{reactivity} * \gamma + K_{priveleged\ user} * \delta \quad (1)$$

This proportional weighting function assesses value based on four parameters. The first term $K_{situation} * \alpha$, evaluates the current capability request in light of the perceived operating environment/mission and subsequently computes a corresponding alpha value. The $K_{distance} * \beta$ term assigns a value dependant on the proximity of the detected object to the vehicle. The closer an object's proximity is to the vehicle, the greater the need to characterize it in a timely fashion. Similarly, the $K_{reactivity} * \gamma$ term evaluates the potential variability in an objects location. Should the initial object detection assess that an object exhibits high dynamicity, the more critical the need to characterize it while it is still observable. At present, this level of assessment is not supported by the Moving Object sensor onboard the Urban NaviGATOR. Such assessment capabilities are expected to come online shortly and are already supported by the CRMF. The

final term, $K_{priveleged\ user} * \delta$, allows for the creation of preferred user groups whose requests receive preferential treatment. In supervisory control situations, for example, a user at an operator control unit may detect a situation which the vehicle autonomy has overlooked. The operator can create a Job Request from his terminal and submit it for processing. The Plant Engineer will evaluate the submission and give it the appropriate priority level.

The synthesizing or tuning of gain parameters (Ks) is in many ways a "dark art". Care must be taken to ensure that any one parameter does not upset the balance of the weighting function. In certain instances thresholds or ceilings and floors were introduced to bound the effect that any one parameter might have on the overall Job Value.

To ensure that high value jobs are executed first a sorting operation must take place. Furthermore, to ensure that stale information is not in use a job's value often needs to be computed a number of times. Newly received jobs are assigned an initial job value. Job Values are recalculated based on change event triggers. Information pertaining to any of the aforementioned parameters immediately triggers a reassessment of the affected job's value. Similarly, the Job Queue is resorted each time a new job is added or when an existing job's value is recalculated. These measures ensure that the framework is executing informed management policy based on current information.

The Plant Engineer job execution cycle commences by popping the highest value job off the queue. This process is illustrated in the flow chart of Figure C-10, but is detailed here. The PE then initiates DSA query for discovered resources which can satisfy the requested job. The DSA returns a candidate resource list structure which contains resource IDs and an estimate of time until ready. In addition to the Job Queue structure, the Plant Engineer contains a structure known as the "PE Resource Collection". This structure contains a vector of PE Resource

Collection Elements, each containing a copy of the framework Resource Object and a vector of Jobs Requests assigned to each resource. Currently Plant Engineer conflict resolution strategy selects the first available resource, i.e. the one with the shortest time till ready parameter. It then checks its PE Resource Collection to determine if the resource is currently available for job assignment. If so, the job is added to the resource collection and will be processed at the next iteration cycle. If the resource is unavailable, the PE selects the next preferred resource from the DSA candidate resource list and attempts to locate a free resource. For instances where resources are tightly coupled, the PE must first verify that all dependent resources are also available before assigning the job to any resource. If all candidate resources are exhausted, and no available resource exists then job is added to the end of the Job Queue.

Reference Implementation Messaging Design

In order for framework analysts to communicate Resource Objects, Job Requests, and Meta-Knowledge amongst themselves in a JAUS transport setting a compatible messaging mechanism was needed. This approaches leverages transport functionality already implemented on the platform. A collection of JAUS experimental messages was crafted to act as carriers of this new data.

The CIMAR JAUS library contains a set of C-language source and header files that provide a JAUS-compliant messaging structure used within the lab. This messaging infrastructure was extended by adding 3 experimental messages to support the data transmission needs of the Cognitive Resource Management Framework Reference Implementation. The newly created messages included “ReportResourceObject”, “ReportJobRequest”, and “ReportJobCompletionStatus” messages. The internal construct of message data utilized JAUS compliant data-types allowing for the reuse of jausXXtoBuffer and jausXXfromBuffer byte buffer message packing and unpacking functionality. The XXs in the previous sentence are place

holders for any one of the data-types specified in the JAUS Reference Architecture. These messages now exist as part of CIMAR's JAUS implementation. As attributes are added to or removed from the implemented framework Knowledge Representation Scheme, the "toBuffer" and "fromBuffer" functions within these messages will need to be modified to reflect those changes.

Reference Implementation Development

This section discusses how the Reference Implementation architecture and design was implemented in software. The integration of software systems was made possible through the collaborative efforts of many CIMAR graduate students. The strategy behind the implementation emphasized technology reuse while seeking to minimize integration costs. Existing Urban NaviGATOR sensing hardware developed for a Robotic Assisted Convoy Operations project [68] along with preliminary Environmental Mapping and Change Detection software architecture designs were used as the starting point. Figures 4-13 and 4-14 illustrate the differences between the initial software architecture conceptualization and the Reference Implementation. As indicated by these figures the framework implementation required both the modification of existing JAUS components along with the development of new experimental components.

Modifications to Existing Urban NaviGATOR Components

To facilitate implementation and promote technology reuse, the existing convoy target tracking component was redesigned. The program already possessed low-level hardware interfaces needed for controlling the actuator and camera resources while providing a JAUS interface to the rest of the vehicle software components. Existing tracking algorithms were modified to accept a goal point, specified using LLA, after which the camera array would servo to a position which satisfied the goal condition. Once the position was achieved, a picture was taken using the framework specified resources. Subsequently, this picture was sent to the World

Model Knowledge Store component as a metadata element of the object specified in the initial Job Request. The modified component was then renamed the Panning Camera component.

CRMF specified modifications included the integration of three Resource Analysts within the Panning Camera component. Resource Analysts for the actuator and both cameras were designed to disclose capability and performance characteristics to the RAA and DSA during resource discovery. Attributes contained within the Resource Object representation needed to convey sufficient information to support the DSA spatiotemporal model as specified in the Resource Object Description and Diagnostician Systemizer Analyst Worksheets shown in Appendix C. Because the framework maintains the computational overhead of ensuring that submitted jobs can be fulfilled, implementations on the resource component side can be simplified.

Creation of New JAUS Components

The Reference Implementation required the creation of two new experimental JAUS components. The first experimental component implemented was the Plant Engineer. Due to the close coupling of activities between Intermediate and Executive Layer Analysts, elements from both layers were integrated to cohabitate within a single software component represented in Figure 4-14 by the Plant Engineer component block. Figure 4-15 shows a similar visualization of framework components expanded to encompass additional system resources. The Plant Engineer experimental JAUS component was responsible for performing all framework duties assigned to the elements it contained as discussed earlier in this chapter and outlined in the framework worksheets of Appendix C.

The second component added to the architecture was termed the Photographer Application Analyst. This component assumed some of the responsibility initially reserved for the Knowledge Store Manger component envisioned in early system design iterations. The

component requests new object creation event notifications from the Knowledge Store and is responsible for constructing a Job Request message and submitting it to the Plant Engineer component. The decision to create a standalone component rather than to embed this Application Analyst functionality in to the Knowledge Store was to compartmentalize integration thereby isolating any changes made during testing to this one ancillary component. A logical future step should include the integration of this Application Analyst functionality into the WMVKS to remove the associated overhead of adding a dedicated experimental JAUS component.

Testing

Having satisfactorily completed the integration of the Cognitive Resource Management Framework with all hardware and software elements onboard the Urban NaviGATOR, testing could commence. The following sections detail the test plans which were developed and the results of both controlled and field testing experiments.

Controlled Testing and Performance Benchmarking

Before attempting full-fledged field testing, a series of experiments under laboratory controlled conditions were devised. These experiments, involving a hybrid of real and simulated systems were performed in two phases for the purpose of providing performance benchmarks proving framework suitability for real-time application.

Test Plan

The first round of testing took place inside the CIMAR lab as depicted in Figure 4-16. These tests involved a GPOS simulator providing fixed vehicle localization data and three simulated static targets. Simulating these conditions assured that all targets were observable by the sensing resources given the vehicles current position and orientation. A series of tests were run, the first of which consisted of 100 repeated job submissions for the image capture of an object located directly in front of the vehicle. These requests did not explicitly specify a target

resource, nor did they specify the advanced reservation of resources. The second test consisted of 100 job requests for all three objects inputted in a circular fashion (1,2,3,1,2,3,1…). From these tests, critical framework performance benchmarks were derived depicting DSA model processing time, PE processing time, and the time difference between PE job assignments. For these tests a supplemental set of logs were generated in addition to the RAA logs capturing the system time data. Processing times are computed by recording the times when the program enters and exits the different processing functions. This data is post processed to compute the differences. Time is captured using the Windows timeGetTime () function which possess millisecond accuracy.

It should be noted that for the purposes of these tests, success is defined as the assignment of a Job Request to a resource and the receipt of a confirmation message indicating that the resource was able to capture an image and send it to the Knowledge Store. Errors resulting from transmission problems between the application and the WMVKS were excluded as they are beyond the scope of the Cognitive Resource Management Framework.

The next evolution of testing limited the simulated capabilities to five desired targets shown in Figure 4-17. Having successfully verified the system's real-time viability, the purpose of this test was to verify proper DSA functionality. Particularly with respect to the coordinate transformations necessary for converting georeferenced location information into vehicle coordinates. This test was designed to expose anomalous behavior between the DSA and PE in situations during which all jobs cannot be satisfied. During this testing, the vehicle was orientated such that only a subset of the desired targets was observable. Seventy-five jobs were submitted in a circular fashion as before.

Test Results

A graphical representation of the results from the single target test is presented in Figure 4-18. A cursory review of the data reveals that the DSA processing time, that is, the time it takes the DSA to generate a candidate resource list for a given job has a uniform value near zero. The PE Total time, which appears to fluctuate violently, represents the time period from when the job was received to when it was assigned. The values escalate because the resources are not capable of fulfilling the jobs as quickly as they are received. Much of the time shown is spent in the Job Queue awaiting allocation. The local minima and maxima are an artifact of the fact that jobs are not processed in the order in which they are received but rather by Job Value. In the case of this test, where all job values are equal, the job queue is iterated through much faster than jobs are assigned and as such the jobs are processed in a semi-random fashion.

Figure 4-19 presents a "zoomed in" view by highlighting a subset (the first twenty jobs assignments) of the data set. Like Figure 4-18 this view confirms the consistency of the DSA processing time along with the stable rate at which jobs are assigned. The red line, indicates the time lapse between job assignments. This data indicates that there is a delta time of approximately one second between assignments. Restated verbosely, it takes a full second for the framework to assign a job, have the job execute, receive confirmation the job has completed, process the next job in the queue, allocate resources for the new job and assign the job to a resource.

Figure 4-20 shows a scatter plot of PE and DSA processing times. PE processing time refers to the total time taken to select a job from the Job Queue, process and assign it to a resource for execution. This metric includes the DSA processing time. A look at the averages and standard deviations presented in the figure highlights the stability of the data. The deviations for PE and DSA processing times are .006916 and .000198 seconds respectively. It is important

to note that the modeling and allocation phases of the framework only consume approximately .0143 seconds of processing time which correlates to an operating rate of 70 Hz, which far exceeds that of any component currently operating onboard the Urban NaviGATOR.

Figures 4-21 thru 4-23 depict the results of the benchmarking test given multiple targets. As was the case with the single target test, the results for DSA and PE processing time are a good indicator of stable framework operation. Figure 4-22 depicts an increase in the time lapse between job assignments. This is expected since the jobs now require articulation of the camera sensor before the image capture capability can be satisfied. As with the first test, all jobs were completed successfully and the average values for DSA and PE processing times were well within the standard deviations of the first series of tests. Objects 4-1 and 4-2 show videos of each test regimen being replicated at the University of Florida Commuter lot test site. The data analysis just presented was captured running the same test procedure within the lab, these videos have been added for completeness and to demonstrate the degree of articulation required during testing.

The results of the second test confirmed that the DSA model and PE Job Queue functions were in-fact working as expected. Job Requests for targets that were immediately observable were executed first, while jobs that were beyond the limits of current platform sensing resources remained in the PE Job Queue. The vehicle was then repositioned in such a manner where the remaining targets were observable. Immediately, the PE began assigning jobs to the resources until all jobs were successfully executed.

Field Testing

Having successfully established performance metrics and proven the functionality of underlying framework components, it was time to deploy the Urban NaviGATOR and evaluate its performance in situ. The Reference Implementation was field tested at the University of

Florida commuter parking lot test site. The site provided a suitable test facility due to the presence of a flat drivable surface with number of cars, recreational vehicles, trees, poles and other structures acting as objects. No simulated components or data were used; the system was operating in a fully autonomous fashion using knowledge acquired through local sensing and perception algorithms.

Test Plan

The test plan consists of simulating a scaled down version of the Environmental Mapping and Change Detection technology demonstration for the Air Force Research Laboratory at Tyndall Air Force Base. The vehicle is situated at the test site with the positioning system calibrated and in a ready state. All necessary JAUS components are then started up according to their prescribed boot sequence. First to boot is the Subsystem Commander which doubles as the Adaptive Planning Framework Decision Broker. This component is responsible for setting the vehicle operating behavior which encapsulates notions of mission goals. Next, World Model Vector Knowledge Store and the Moving Objects (MO) sensor are started up.

The WMVKS acts as the source and sync for all ascertained knowledge regarding objects detected by the MO component. The MO component uses a SICK LD-LDRS1000 LADAR which possesses a 270 degree field of view out to 180 meters. Simply stated, the MO sensor is capable of detecting objects which exist far beyond the observable range of the cameras. Furthermore, at the time of testing this sensor experienced a large susceptibility to sensor-noise due to imperfect measurements and vehicle positioning drift. In an effort to mitigate these effects, the MO sensor was used to capture a "snapshot" of the environment. Using data from this snapshot alone, the component attempts to classify readings into new objects which are then committed to the knowledge store.

The final applications started up are the Photographer Application Analyst, Panning Camera, and Plant Engineer components which integrate the Cognitive Resource Management Framework into the Urban NaviGATOR JAUS compliant software architecture. Once started, the Photographer Application Analyst begins transmitting Resource Objects to the Plant Engineer Component where the RAA logs the discovery and subsequent updates of resource capabilities, while the DSA constructs a real-time model of the sensing resources.

New objects are detected when an operator initiates the "snapshot" function within the Moving Objects Component. The Photographer Application Analyst receives notice of the new objects and subsequently creates a unique Job Request for each identified object of interest now stored in the WMVKS. The Plant Engineer prioritizes the incoming requests and places them into the Job Queue and sorts the queue by Job Value. It then seeks to process jobs possessing the highest value first. Should the DSA indicate a suitable candidate resource exists and the PE verifies that said resource is not currently tasked; the job is assigned to the resource for execution. This process continues until all achievable jobs are complete. Because the sensor used for object detection has a more expansive viewable area than the cameras it is likely that jobs will remain in the PE's Job Queue.

A human operator then begins to drive the vehicle around the test site. As the vehicle position changes, the PE recalculates the value of the jobs remaining in the queue and subsequently sorts the queue. All the while it is attempting to process the remaining jobs. As objects become observable the DSA will begin to populate Candidate Resource lists which the PE uses to assign the job to an available resource. The vehicle is driven around the course until no further jobs remain in the Queue.

Test Results

Object 4-3 links to a video that was made while field testing the Cognitive Resource Management Framework Reference Implementation on December 6, 2009. An excerpt from the Resource Appraiser Analyst log corresponding to this test run is found in Appendix D. Upon taking a single MO snapshot approximately 30 new objects were detected and added to the World Model Vector Knowledge Store. The test evidences that the CRMF provides mechanisms for discovery, modeling, and monitoring of sensing resources while providing a knowledge representation scheme which enables goal abstraction and a uniform approach to job submissions; all critical aspects needed to achieve intelligent autonomous real-time resource management.

Several key concepts were demonstrated while proving their viability in field testing the Cognitive Resource Management Framework Reference Implementation:

- Application of Plant Engineering analogy with hybrid scheduling theory paradigms to provide near real-time resource management of a full scale autonomous vehicle platform undergoing RSTA activities
- The notion that distributed cooperating analysts, implemented as software entities, distill critical system information into knowledge attributes which result in actionable behavior generation through resource brokering
- Application of 3-dimensional dynamic spatiotemporal model of common AGV sensing resource capabilities
- Development of a modular and extensible Resource Object defining Capability and Performance Attributes which embodies the essence of a resource
- Framework promotes technology reuse through the use of a uniform the knowledge representation scheme and reasoning mechanism thereby reducing integration costs
- Use of a distributed deliberative reasoning mechanism operating in near real-time
- Framework components and tools aid in design, implementation, and evaluation of complex multi-mission unmanned systems

- Integration with existing autonomous ground vehicle standards and frameworks to provide intelligent resource management

Object 4-1 Sample Video of Cognitive Resource Management Framework Controlled Testing-single target case

Object 4-2 Sample Video of Cognitive Resource Management Framework Controlled Testing-multiple targets case

Object 4-3 Video of Successful Cognitive Resource Management Framework Field Testing at the University of Florida Commuter Parking Lot Site

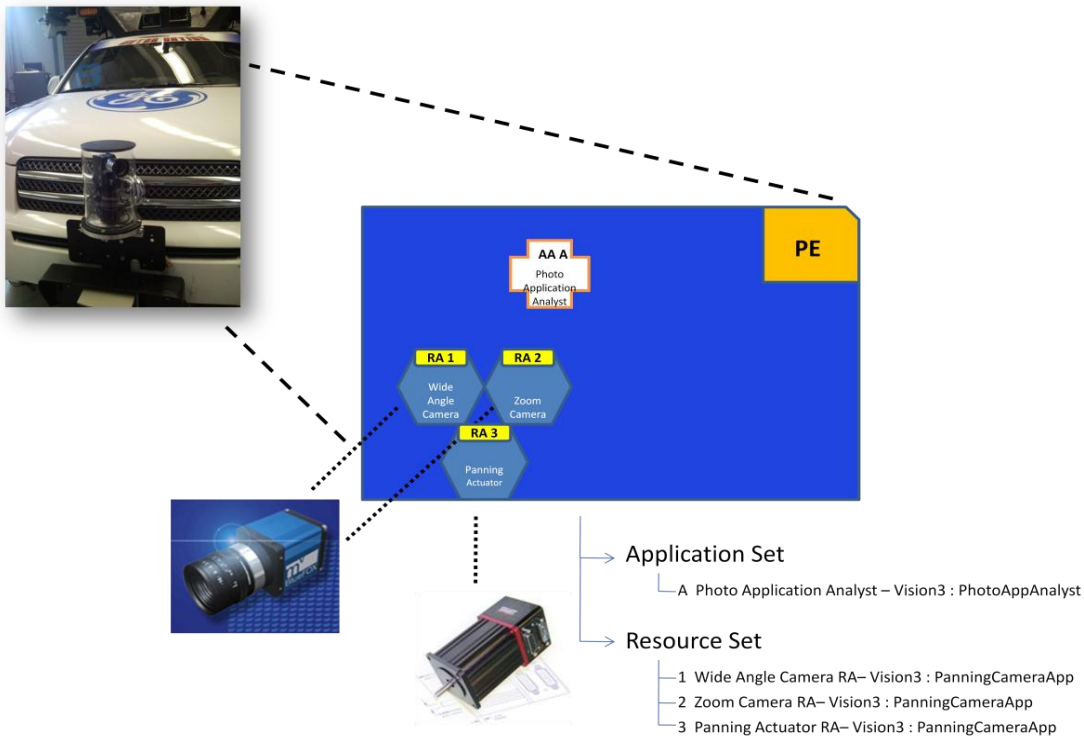


Figure 4-1 CRMF Reference Implementation Resource Virtualization

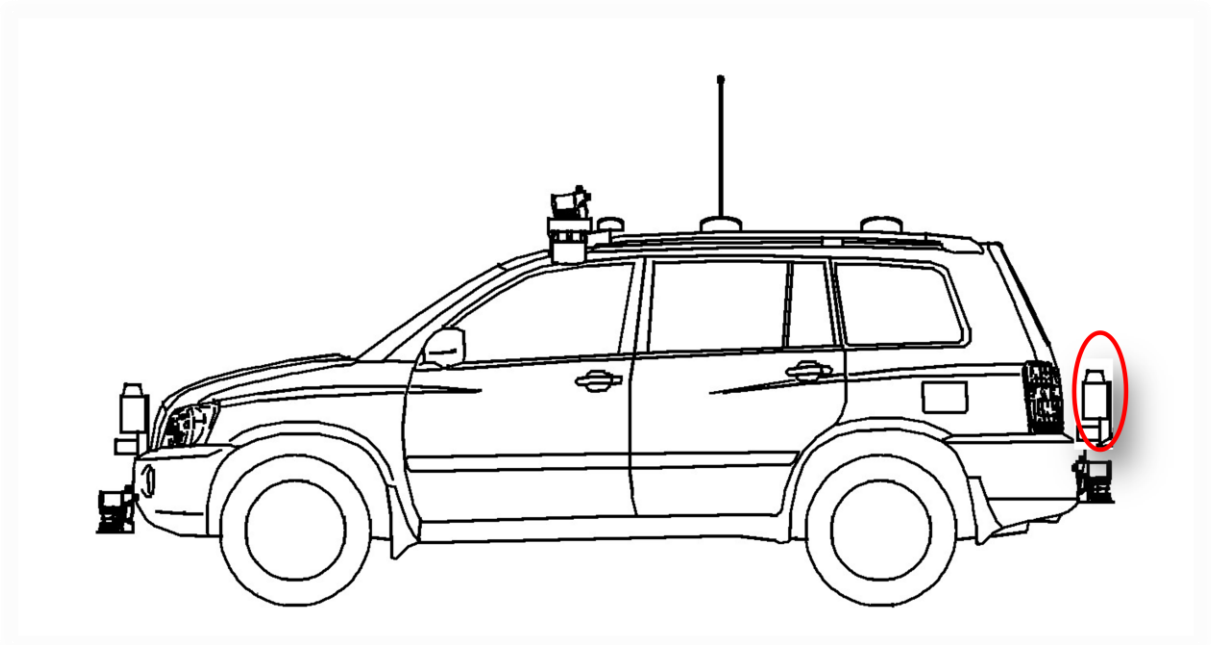


Figure 4-2 Redundant resource added to vehicle platform

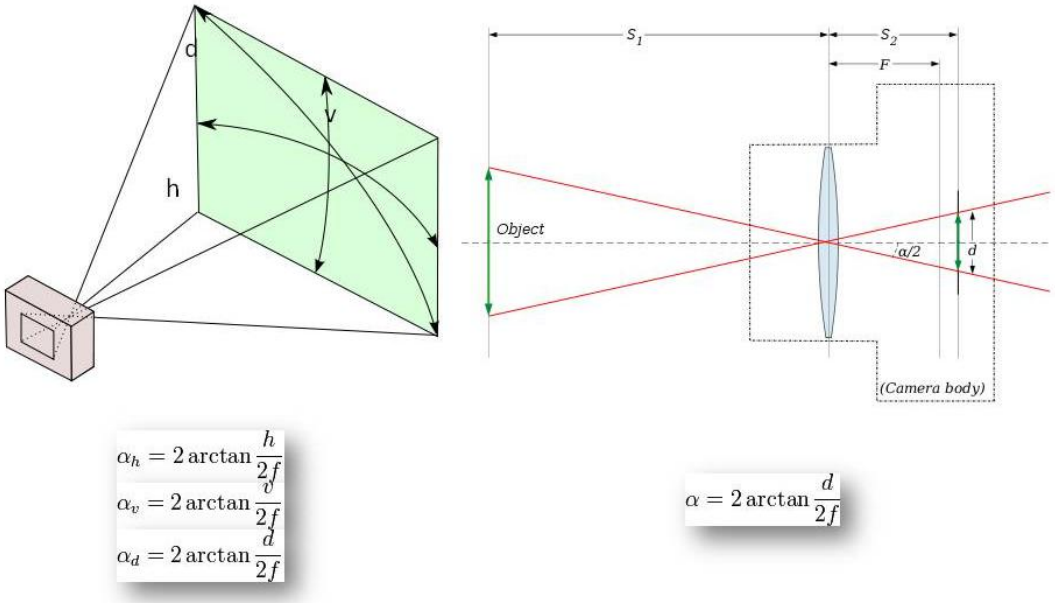


Figure 4-3 Camera sensor geometric representation parameters

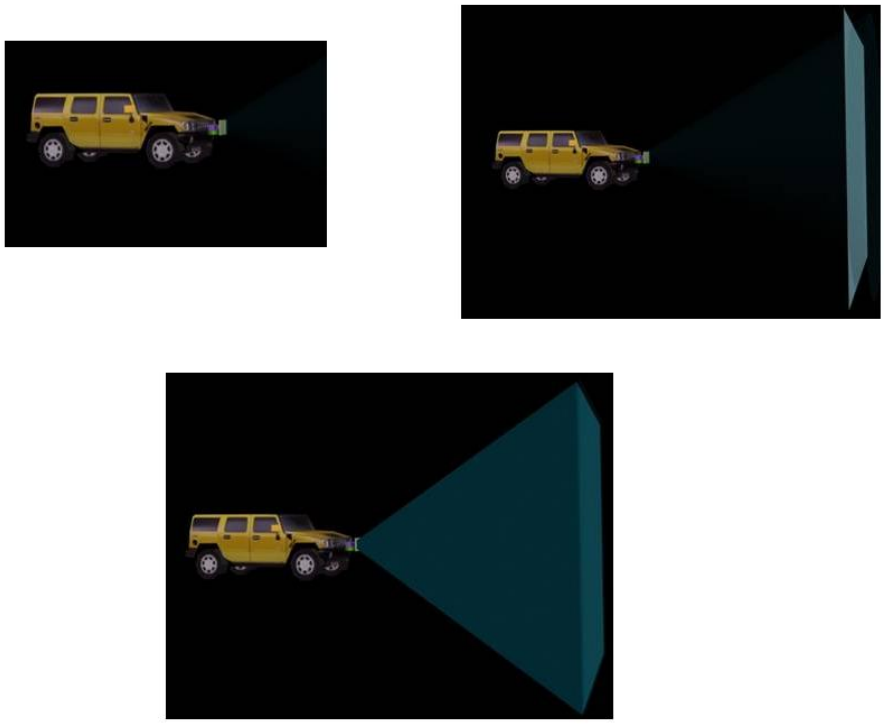


Figure 4-4 Camera sensor frustum geometric representation

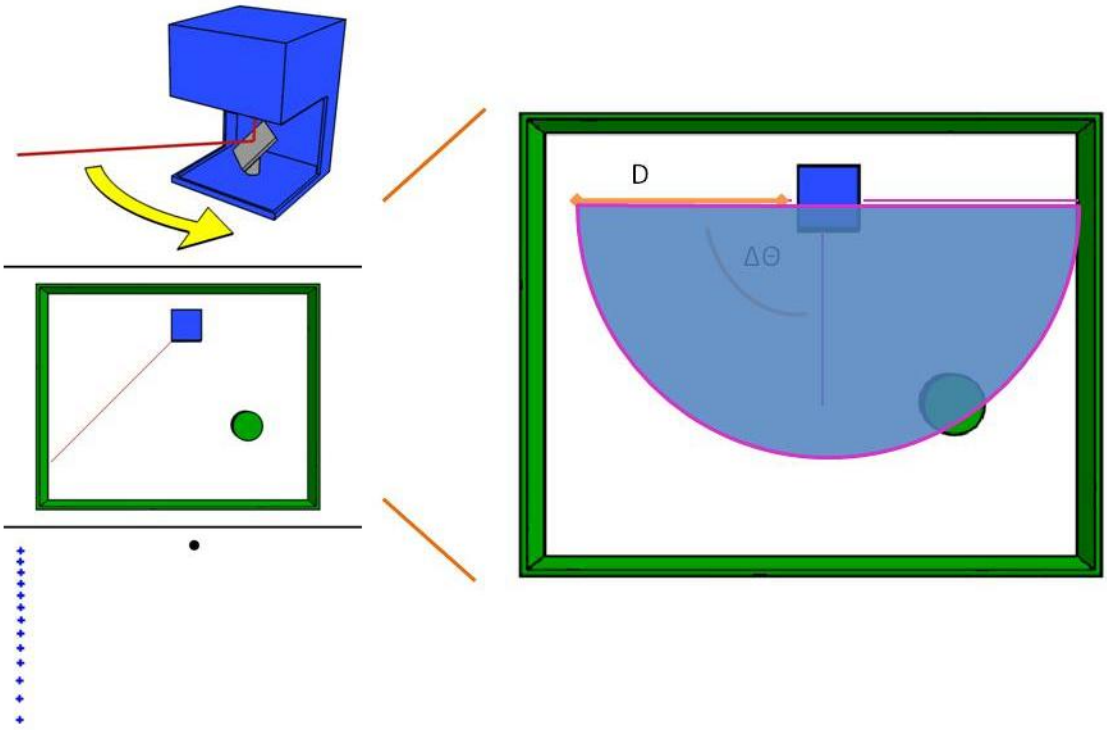


Figure 4-5 LADAR sensor geometric representation parameters

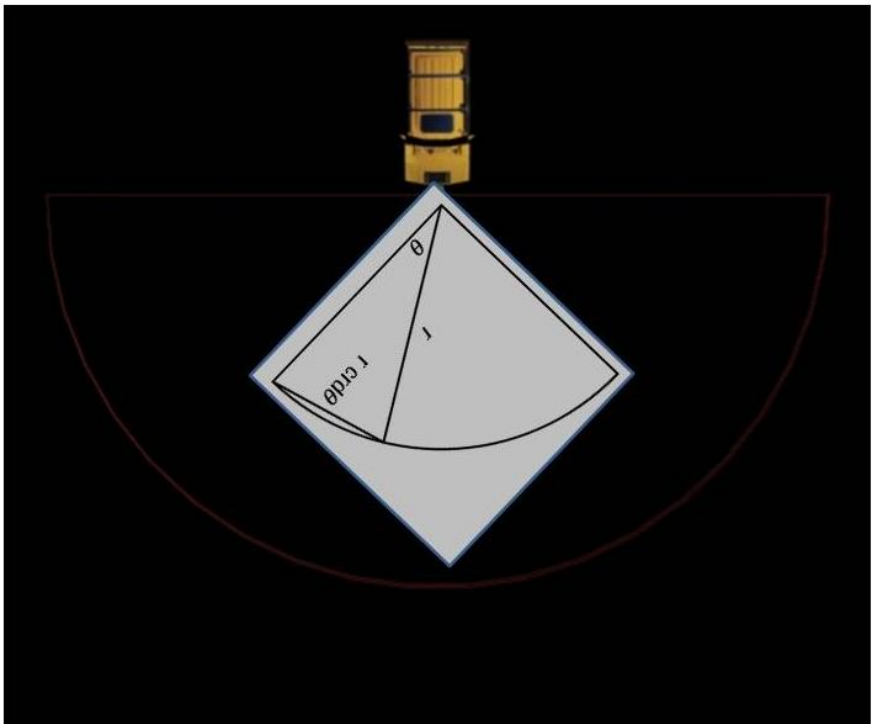


Figure 4-6 LADAR chord geometric approximation

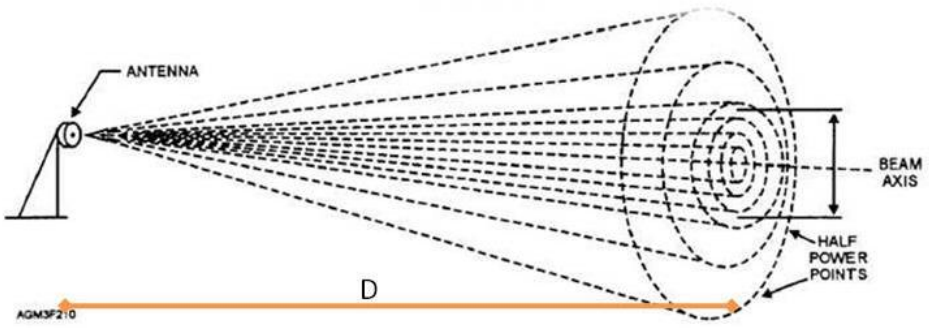


Figure 4-7 Radar sensor geometric representation parameters

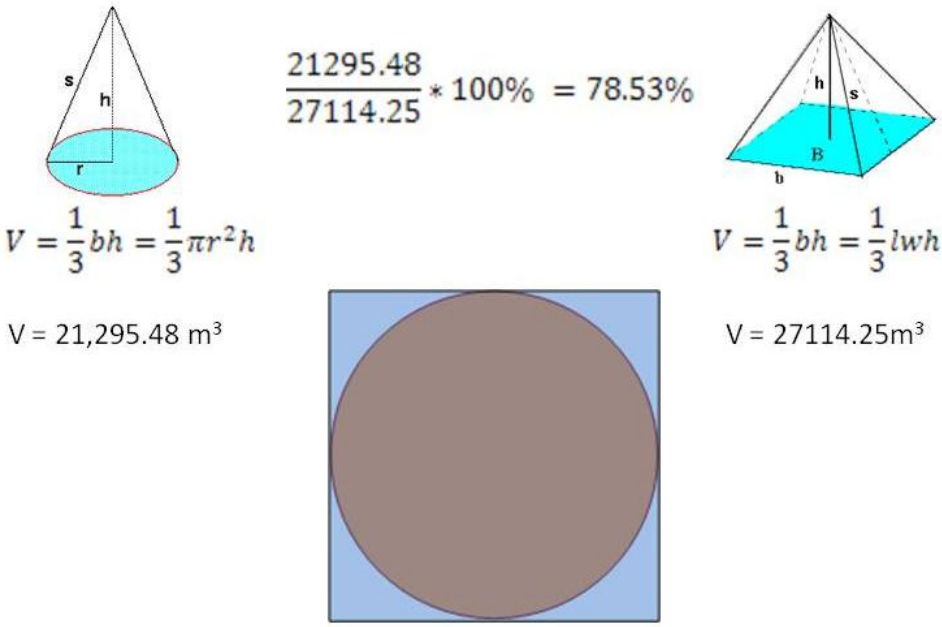


Figure 4-8 Radar sensor geometric approximation analysis

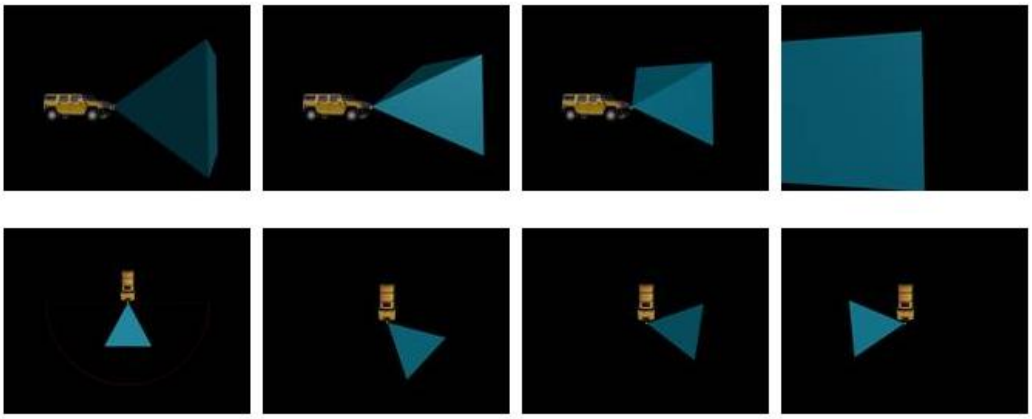


Figure 4-9 Sensor and actuator coupling

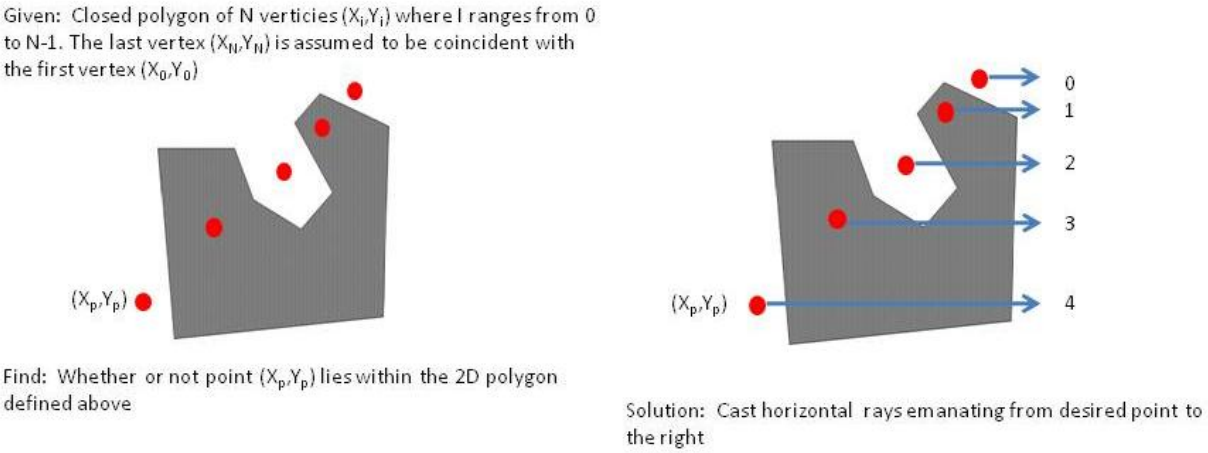


Figure 4-10 Generic point-in-polygon problem and solution

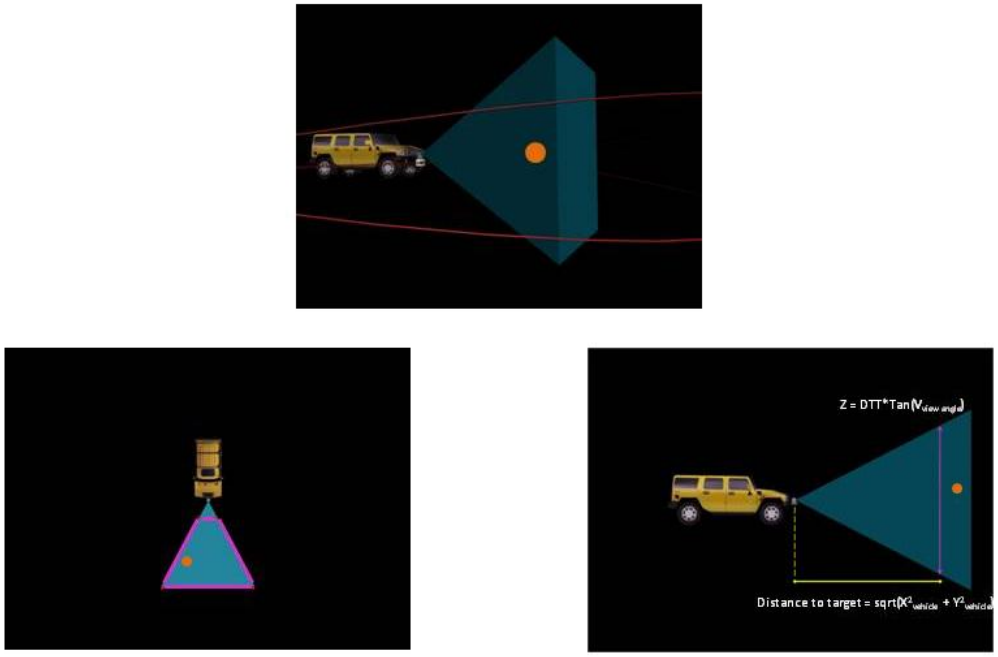


Figure 4-11 Point-in-polygon algorithm application

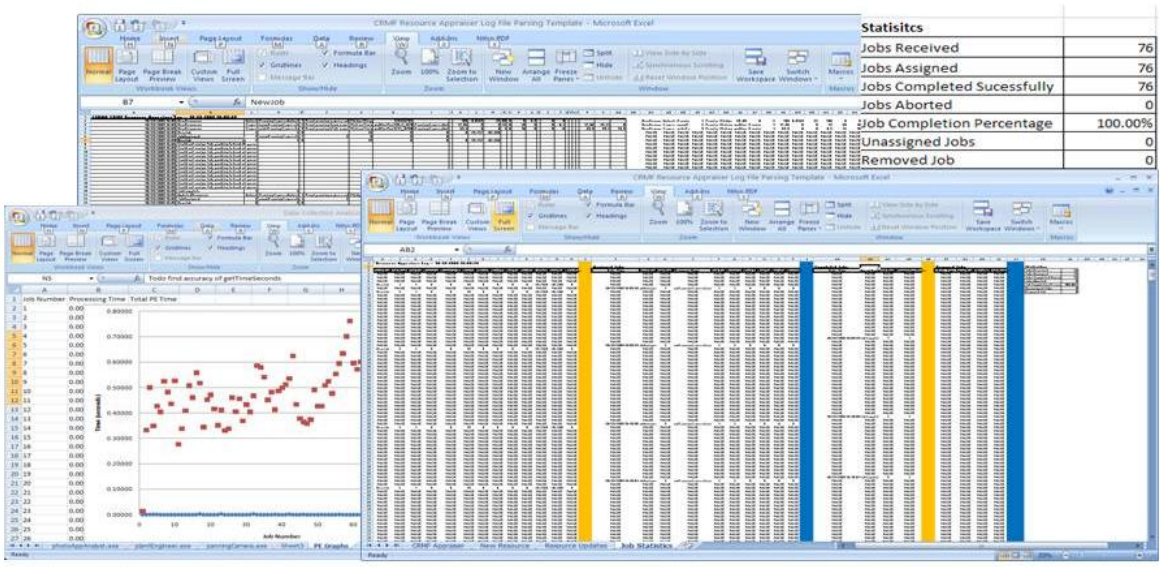


Figure 4-12 RAA Excel spreadsheet workbook snapshot

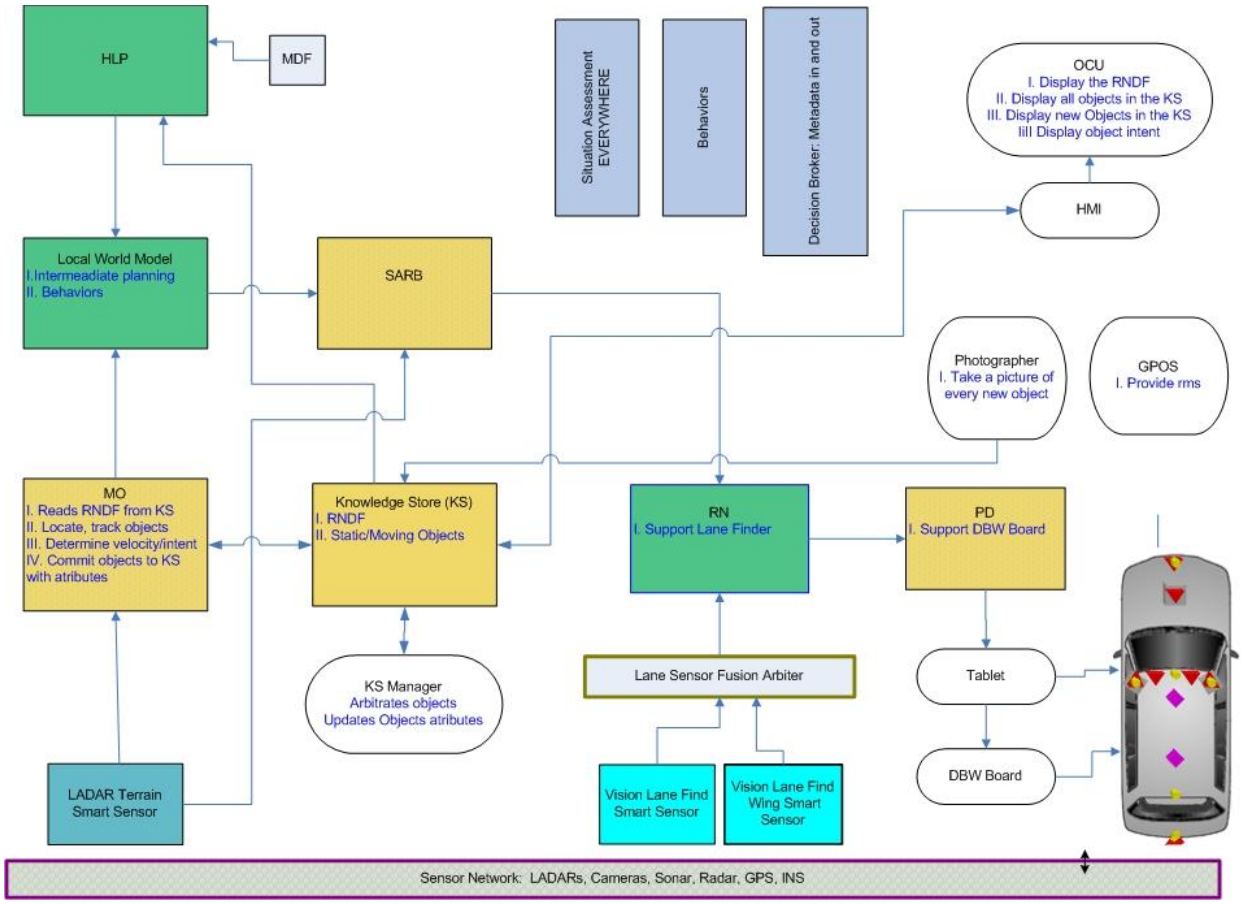


Figure 4-13 Environmental Mapping and Change Detection demo initial architecture

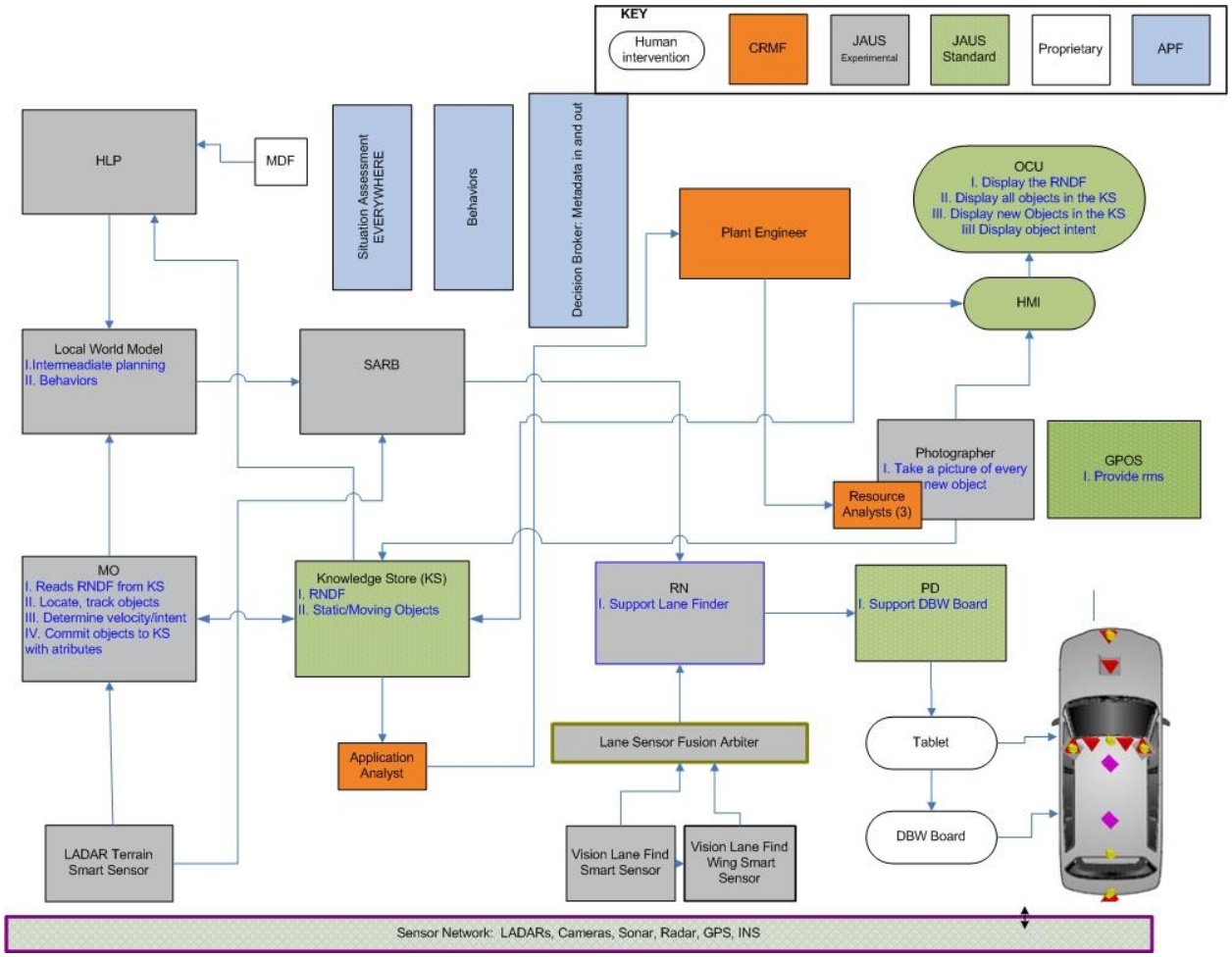


Figure 4-14 Reference Implementation architecture (featuring CRMF)

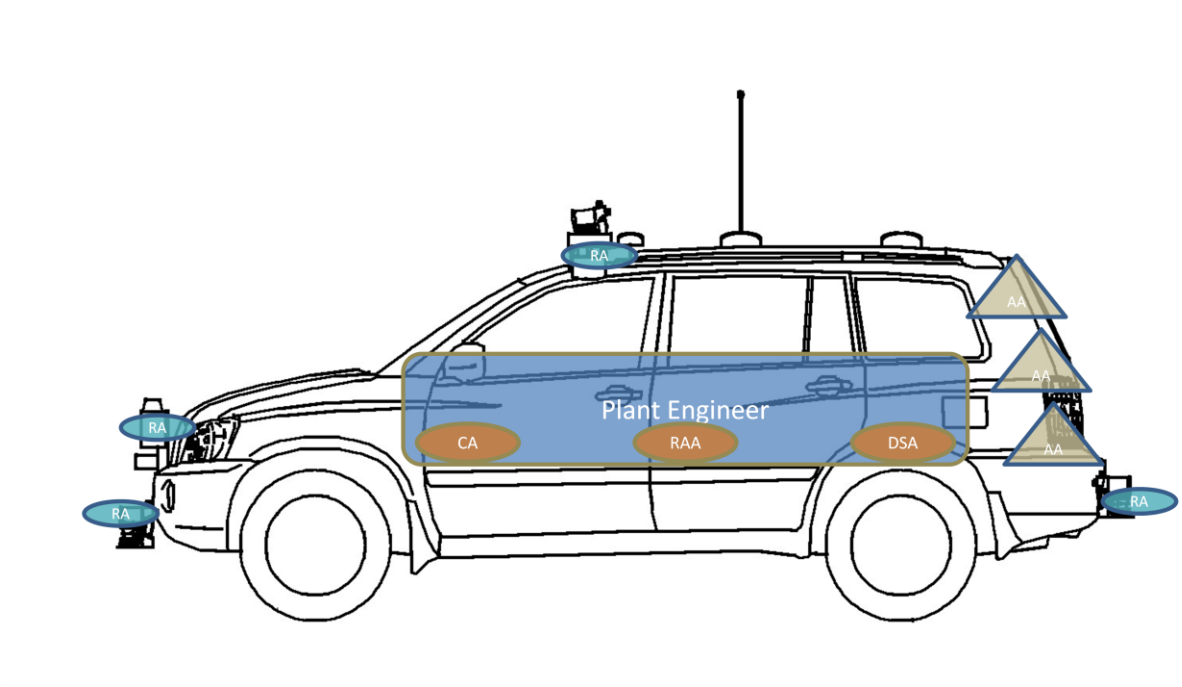


Figure 4-15 Alternate visualization of CRMF system resources

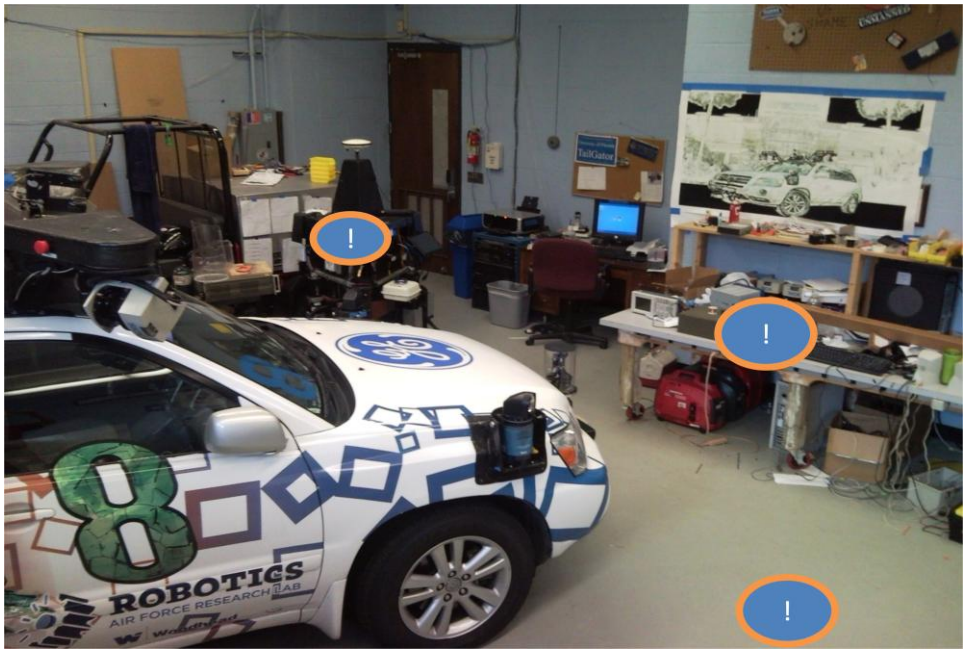


Figure 4-16 Laboratory performance benchmarking setup, exclamation points signify the location of simulated Job Request targets

Figure 4-17 Controlled testing outside CIMAR lab

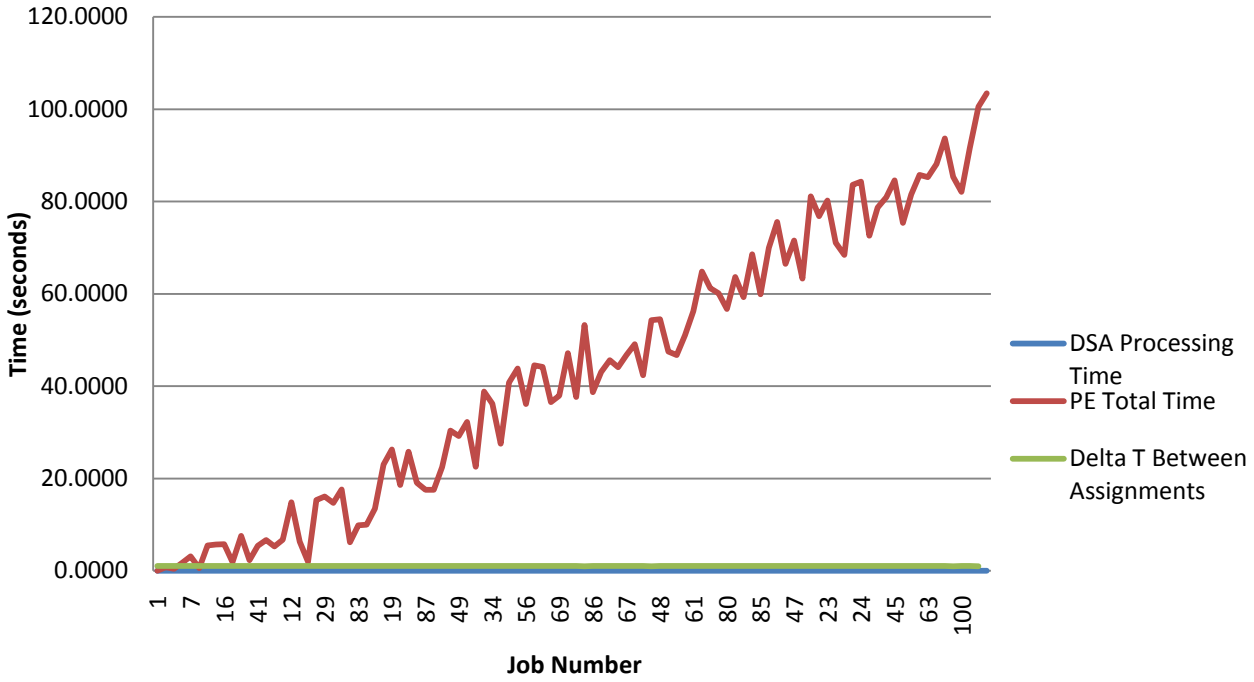


Figure 4-18 CRMF controlled benchmark testing results – single target, no articulation, JobQueue buildup

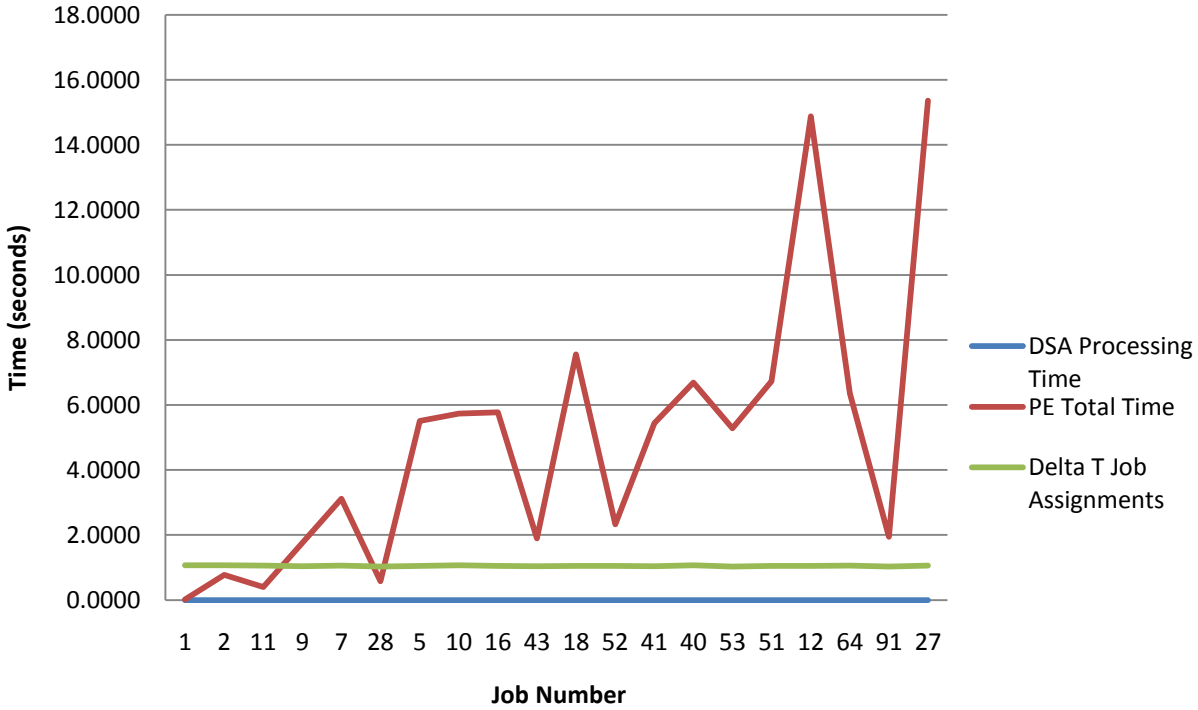


Figure 4-19 CRMF controlled benchmark testing results – single target, no articulation, JobQueue buildup (first twenty results)

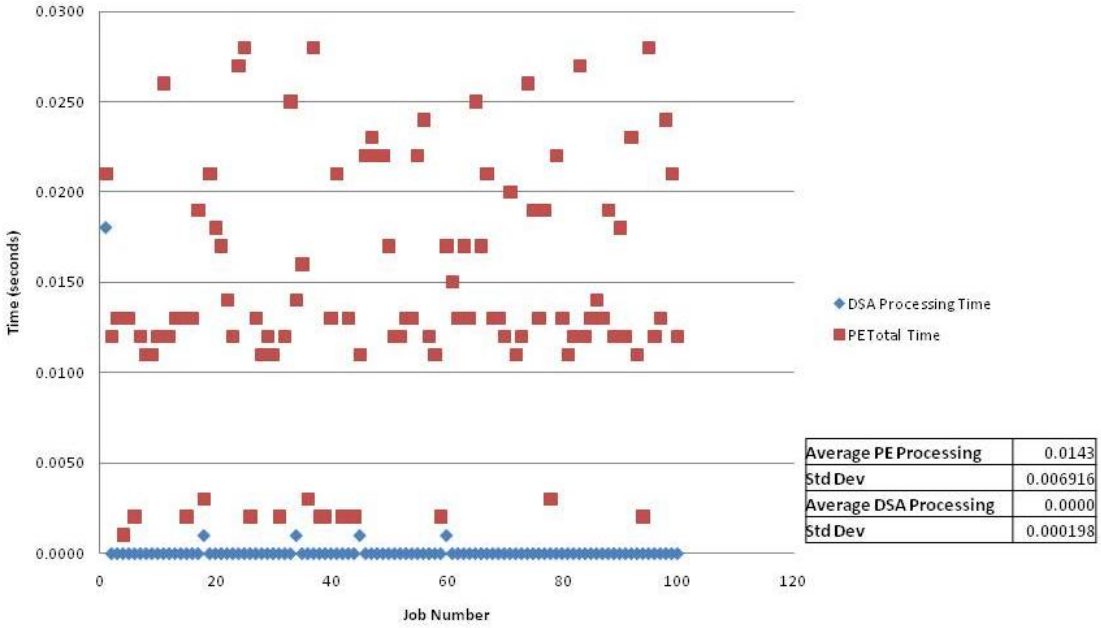


Figure 4-20 CRMF benchmarking results- single target scatter plot with statistical analysis

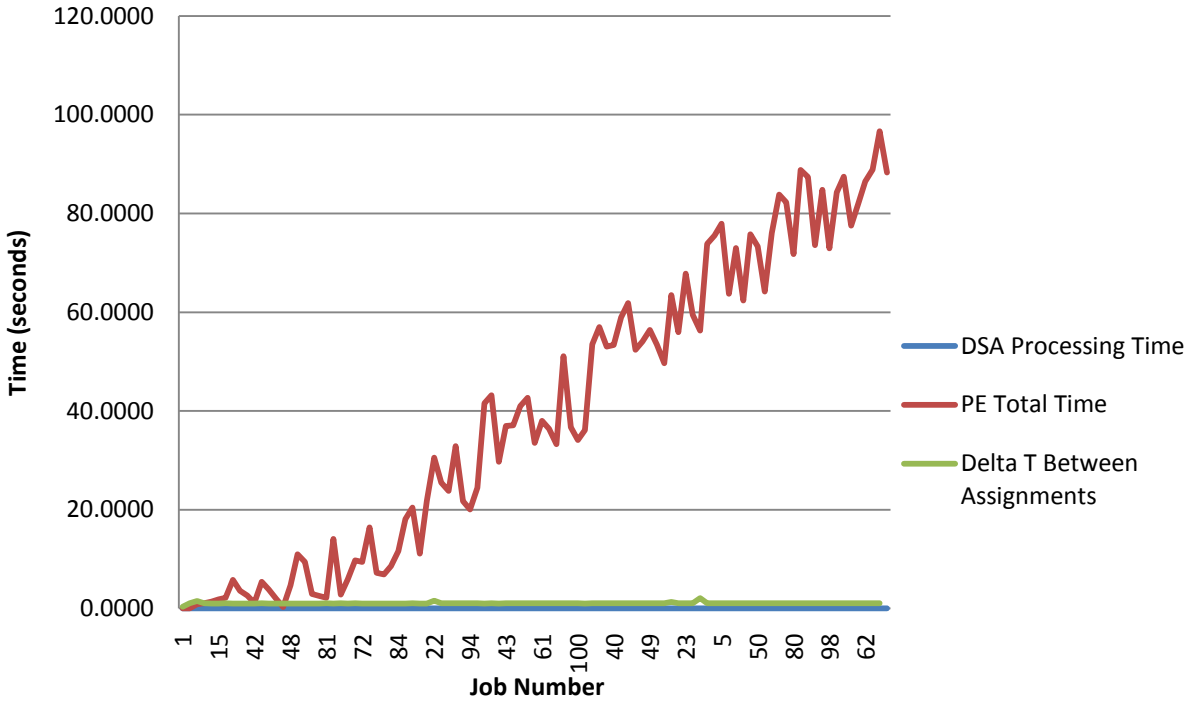


Figure 4-21 CRMF controlled benchmark testing results – multiple targets with articulation and JobQueue buildup

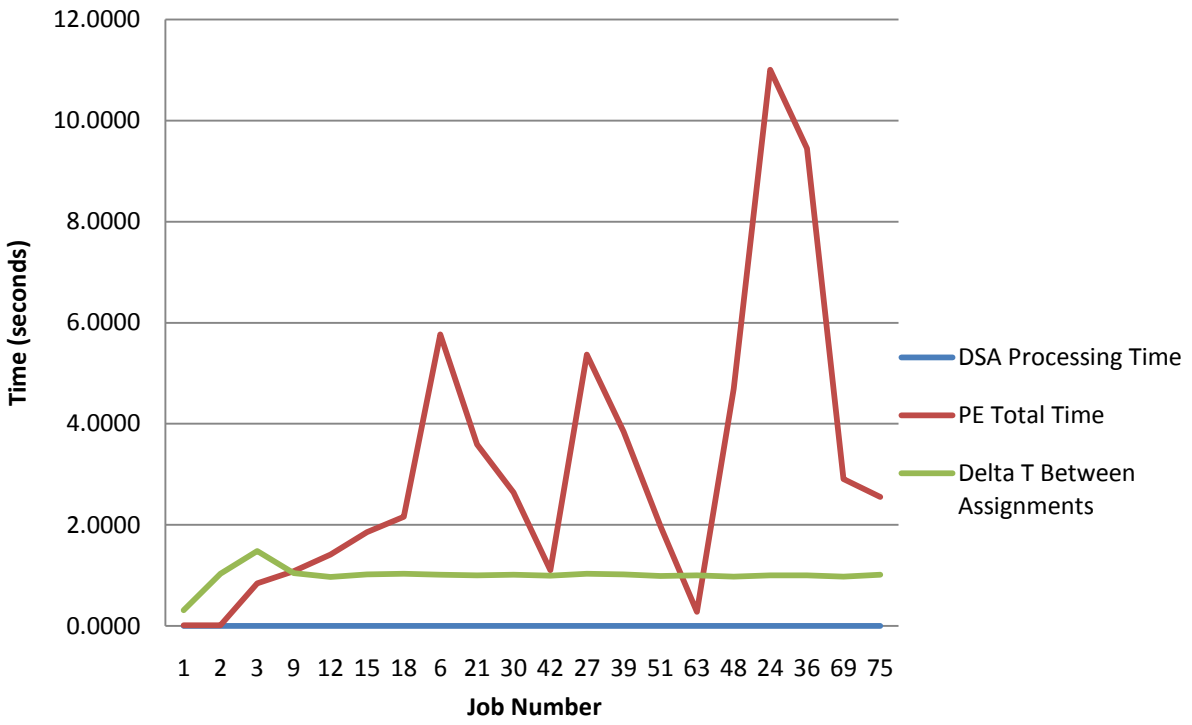


Figure 4-22 CRMF controlled benchmark testing results – multiple targets with articulation and JobQueue buildup (first twenty results)

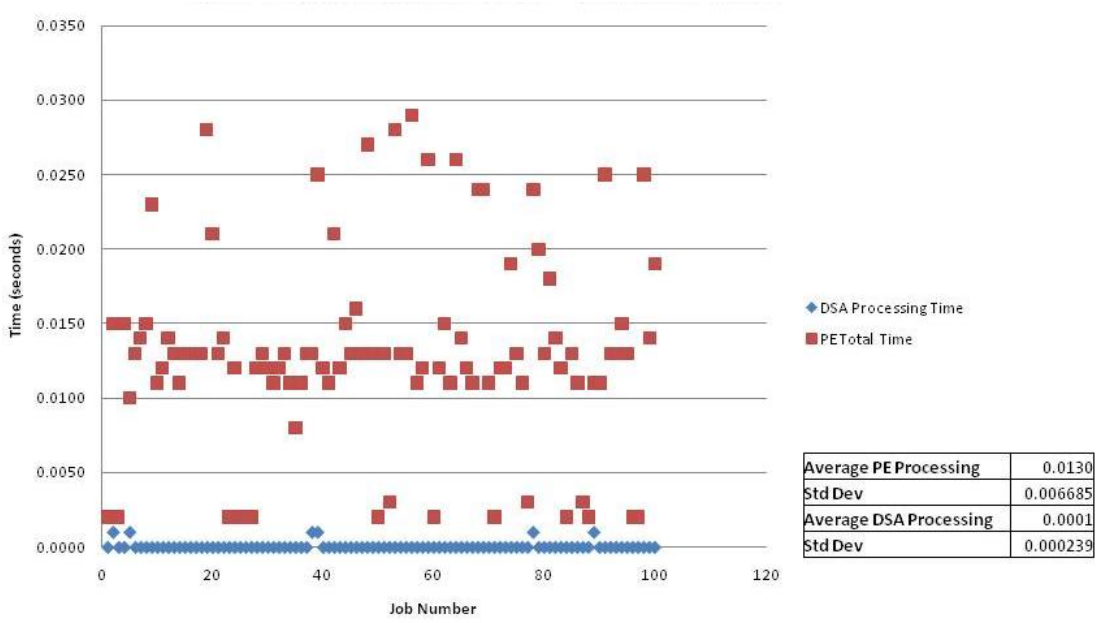


Figure 4-23 CRMF benchmarking results- multiple target scatter plot with statistical analysis

CHAPTER FIVE
DISCUSSION AND FUTURE WORK

The design, development, and implementation of a Cognitive Resource Management Framework for autonomous ground vehicle sensing was presented in Chapters 3 and 4. This chapter presents the author's assessment of the work while introducing potential areas of future work to further enhance and evolve this framework.

Assessment of the Cognitive Resource Management Framework

The Cognitive Resource Management Framework has been shown to be both a viable method for modeling and managing a distributed collection of heterogeneous resources and resource-requests and a valuable contribution to the researchers whose algorithms depend on the reliable availability of resources to fulfill mission operating requirements. The framework's viability and contributions were demonstrated in both the Reference Implementation, discussed in Chapter 4, and during the CIMAR's Environmental Mapping and Change Detection demonstration for the Air Force Research Laboratory conducted at the Gainesville Raceway test facility. This work will not only continue to benefit researchers at the University of Florida who will build upon the foundation created by the Reference Implementation, but could impact the robotics community as a whole by gaining acceptance in standards bodies such as SAE's AS-4 committee.

The framework's value is evidenced by the pronounced role it has assumed in the Urban NaviGATOR software architecture. Researchers at the Center for Intelligent Machines and Robotics will continue utilizing the Cognitive Resource Management Framework while developing advanced technologies that broaden the frontiers of autonomous ground vehicle application. Currently researchers are developing algorithms for simultaneous localization, mapping, and object tracking using multiple 2-D laser scanner sensors. Not only can the

framework be utilized to model and monitor these resources, it can intelligently select the most appropriate resources to utilize as inputs to the mapping and tracking algorithms should other applications require the intermittent use of a particular scanner. Furthermore, the Plant Engineer value judgment algorithm in the Reference Implementation was designed to incorporate object tracking data provided by the MO component.

The framework maintains a central role in the follow-on work to the 2009 Environmental Mapping and Change Detection demonstration. In addition to refining the object tracking and motion classification sensor, new technologies such as an object classifier have yet to be deployed. This classification capability will use the image captured using the Photographer Application Analyst and Panning Camera component along with other knowledge in an attempt to discern the nature of the object. Subsequently, the classification process may require the use of other resources onboard to gain a better understanding of the object. The creation of such requests is now possible using the Reference Implementation and the framework tools developed in this work as a blueprint[1]. Another potential area of application for the framework involves CIMAR's role in the Robotic Range Clearance Competition (R2C2) sponsored by the Joint Ground Robotics Enterprise in collaboration with the US Army Corps of Engineers and the Air Force Research Laboratory [69]. The creation of this competition exemplifies the desire to field high-capability high autonomy vehicles which simultaneously demonstrate advanced navigation and payload operation capabilities. This competition provides a venue to showcase the CRMF's potential application to both sensing and agricultural payloads should future researchers decide to undertake this work.

[1] This process will likely require the creation of new Application and Resource Analysts along with modifications to the Job Request and Resource Object knowledge representations. DSA system model alterations will require new testing and evaluation as described in Chapter 4.

Future Work

The research present herein established the need for a Cognitive Resource Management Framework to supplement the shortcomings of existing robotics standards while highlighting the advantages of such an approach. During the course of the current work, a number of improvements along with new areas of application were identified as opportunities for future research. These opportunities, which are summarized below, are categorized based on whether the work is more theoretical in nature or if it more implementation driven.

Theoretical Opportunities

The Diagnostician/Systemizer Analyst greatly affects the composition of the framework knowledge representation scheme which is designed to encapsulate all pertinent situational information. The following paragraphs discuss potential DSA model permutations which supplement the current spatiotemporal modeling capabilities developed for the Reference Implementation. It is understood that with each new model comes the responsibility of defining the knowledge representation scheme elements which feed the model.

Perhaps the most applicable area of future study involves the development of a 3-D spatial model which accounts for rugged terrain. While the current model accounts for vehicle roll and pitch it assumes an idealized flat planar surface. This assumption is suitable for a number of operational scenarios, but can prove ineffective in rugged operating environments are encountered; environments such as those encountered during the R2C2. Researchers at the University of Florida are currently investigating approaches for constructing and modeling a ground plane representation within a PostGIS knowledge store. The author's preliminary DSA design work investigated this approach, however, it was determined that development of such model was in and of itself a substantial body of research. A number of potential pitfalls with this approach were identified which future researchers will need to investigate and address. Perhaps

the most critical of these involves the modeling and planning time conundrum. The WMVKS played a critical role in the Environmental Mapping and Change detection demonstration, however, at times this model becomes inundated with query and commit requests which adversely affect the reactivity of the model. Use of this model would certainly not have provided the sub-millisecond response times demonstrated using the Reference Implementation DSA. However, it is foreseeable that such a model is possible through advancements in software algorithms and computational hardware techniques such as outsourcing certain computation operations to a graphics processing unit.

Another area for future development involves the creation of a sensor knowledge repository. This entity would exist as part of the DSA and could either utilize *a priori* information, rules, or it might dynamically tune itself based on real-time Resource Appraiser Analyst findings. The goal is to supplement the current Candidate Resource List, which the PE uses when allocating resources, with a suitability recommendation. Before rendering an opinion, the repository would use assimilated job, sensor, and environment information before rendering an opinion on the suitability of a resource. This recommendation could exist in the form of a reported confidence value associated with each resource match to a particular PE query. This metric provides more granularity than the Boolean "fit or unfit" functionality currently supported in the Reference Implementation. This added information not only provides a mechanism for conflict resolution, but also facilitates the development of more sophisticated allocation algorithms resulting in enhanced system performance.

Some final thoughts regarding the DSA model concern its use for configuration management. At present, the DSA configuration modeling and modification is limited to position, velocity and acceleration for actuators, and specific field-of-view and resolution related

parameters for sensors. Researchers are encouraged to investigate the inclusion of quality metrics within a Job Request submission. These metrics might reference scan density or image resolution which would help prune the search for suitable candidate resources. Once incorporated, the model will need to assess the potential impact of these changes on current resource utilization needs. This opens the door for a new area of intelligent allocation and brokering study with a focus on developing solutions and configurations which result in sub-optimal resource utilization for certain tasks in order to optimize others.

One final area of theoretical work involves the extension of the CRMF to manage a system of systems. The author's initial conceptualization, which employs a system level Intermediate and Executive framework layers, is depicted in Figure 5-1. Each subsystem maintains its local framework implementation responsible for local low level resource management duties with the addition of a new subsystem Resource Analyst. This analyst would be responsible for providing a more abstract description of the subsystem's collective capabilities which would be utilized by the system-level framework components. At the system-level, human operators or other intelligent battle management systems might act as the Application Analysts, detecting the need for a job and describing that need to the Plant Engineer in terms of a new Job Request.

Researchers are tasked with conceptualizing new Resource Object representations which effectively convey a platforms overall capabilities. Capability and Performance attributes would need to be defined along with a suitable system-wide DSA model representation. The Job Request knowledge representation provides another fruitful area of research. Researchers must define new Capability and Target Attributes that adequately relay the task parameters to the System DSA and the subsystem level PE. The emergence of specialized robotic platforms and payloads market results in an operating setting where multi-vendor robot collaboration is

required to complete complex missions; the Cognitive Resource Management Framework offers a means through which this can be achieved.

Implementation Opportunities

The Reference Implementation resulted in the development of a significant body of software which has been added to the CIMAR's C libraries in addition to existing software components. This research now provides several opportunities for researchers at the University of Florida to build upon.

One area involves the integration of the Cognitive Resource Management Framework with CIMAR's 2007 DAPRA Urban Challenge software architecture. Presently, the reference implementation Resource Object supports all the sensors and actuators currently deployed onboard the Urban NaviGATOR. The task for future researchers is to implement and document the many Resource and Application Analysts needed for resource discovery and job submission. Application Analysts must be created to handle the various information requirements of the various Situation Assessment and Behavior Specialists implemented as part of the Adaptive Planning Framework shown in Table A-1. New Capability Attributes representing the information needs of a particular application analyst will need to be designed. At the Intermediate and Executive architecture layers Plant Engineer value judgment algorithms will need to be tailored to suit the needs of autonomous navigation in structure and dynamic environments. This task will surely require modification to the Communicator Analyst which is tasked with importing data into a framework manageable form. While this migration requires a concerted effort on the part of researchers, such efforts will not be in vain. The upshot is that once completed, the platform will not only benefit from the fault tolerance afforded by an intelligent resource management framework, but will also benefit from the performance evaluation functionality offered by the RAA. Perhaps the most important contribution directly

relates to the Urban NaviGATOR's role as a test-bed for new technology. By formalizing the process of modeling, monitoring, and managing resources, the integration costs of adding or removing new sensors as technologies are developed is significantly reduced.

The Reference Implementation highlighted another area in need of development, a transport mechanism with guaranteed delivery. In an effort to reduce message traffic, the framework relies on event based notifications. CIMAR's JAUS implementation relies on a UDP interface between components, which is susceptible to packet loss, resulting in lost or dropped messages. Testing indeed confirmed the occurrence of such events and subsequent measures had to be taken to mitigate the effects of an unreliable transport mechanism. As software architectures and standards such as AS-4 migrate towards on-change notifications the need for having automatic delivery confirmation becomes more pronounced. This feature would simplify the implementation of both the Cognitive Resource Management and Adaptive Planning Frameworks.

The final proposed area of future work involves the Migration of the CRMF technology and CIMAR's software architecture to the new SAE AS-4 Standard. At present, CIMAR maintains compliance with the JAUS RA 3.3 with the addition of experimental components and messages as needed. With the JAUS working group having been officially de-chartered, much of the JAUS RA work has migrated to the AS-4's service-oriented standards development. Code generation tools are currently under development to aid roboticists with this transition. What remains to be done is integrate CIMAR's proprietary intellectual property, which now includes this research to achieve full compliance with the standard. Through undertaking these efforts research technologies developed at the University of Florida will gain greater exposure within

the robotics community which will afford numerous future opportunities for collaborative experimentation and joint research partnerships.

Conclusion

This dissertation presents a new approach to intelligent resource management onboard autonomous ground vehicles. The framework formalizes the discovery, modeling, monitoring and configuration of AGV resources. This work provides new levels of autonomy not currently supported by predominant unmanned systems technology standards. The framework promotes fault-tolerance and robust behavior while operating in highly dynamic environments. Furthermore, it provides tools which serve as design-time aids, form the guidelines for implementation, and becomes the reference documentation for development, integration, testing, and production. The framework knowledge representation scheme provides an extensible set of methods for organizing and disclosing resource, job and meta- information which its intelligent architecture leverages to gain self-awareness through which optimal decision-making is achieved. Once implemented, this research provides the foundation upon which new sensing, planning, and scheduling methodologies pertaining to intelligent resource utilization can be implemented and evaluated with the hope of promoting the advancement of highly autonomous platforms capable of sophisticated interactions with unknown, dynamic environments.

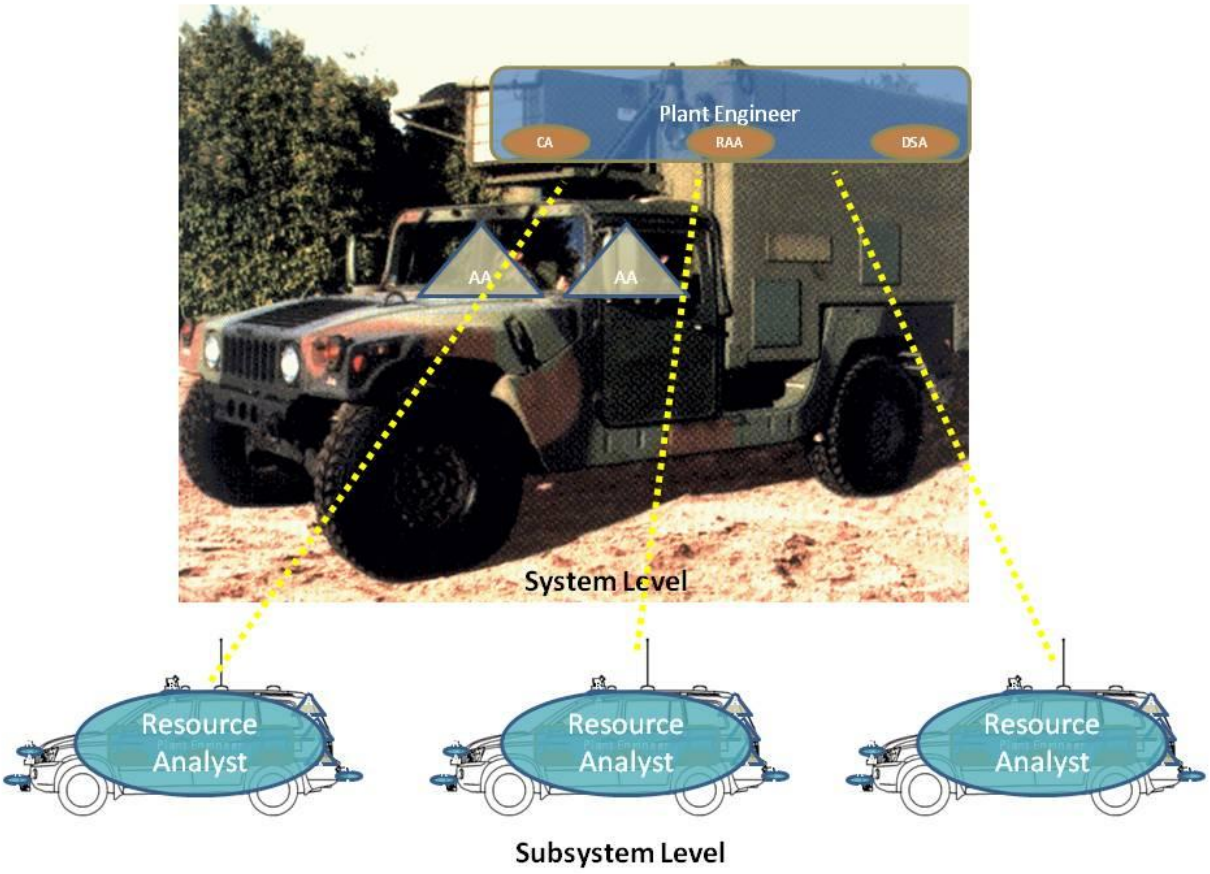


Figure 5-1 Potential CRMF extension to System of Systems

APPENDIX A
RESEARCH SETTING

The Center for Intelligent Machines and Robotics has contributed to the evolution of unmanned mobile robotics through the development of precision positioning systems, off-line path planning, and vehicle control approaches which, upon integration, yield an autonomous platform capable of navigating along a planned path. Recent areas of research advancement including high speed autonomous vehicle control, environmental sensing and modeling, enhanced localization systems, mission planning, and intelligent decision-making architectures have telegraphed the need for more intelligent and autonomous resource management. The following sections focus attention on the evolution of intelligent decision-making and capabilities management, areas critical for the development of effective and sophisticated autonomous vehicles, while noting the roles in which a resource management framework could extend a UGV's intelligent capabilities.

Intelligent Decision Making and Capabilities Management

Elements of Touchton's knowledge representation scheme and accompanying planning and decision-making framework were first applied by Team CIMAR in the 2005 DARPA Grand Challenge. His work was later formally documented in [25]. The framework, through the application of forward chaining reasoning, manages a virtual collection of Situation Assessment, Behavior, and Decision Specialists. Knowledge in the form of "findings" and "recommendations" is passed throughout the framework specialists. Situation Assessment Specialists (SAS) attached to perception elements continually update findings regarding a set of previously identified conditions, states, and events that are of importance to other specialists. The Behavior Specialists (BS), which exist in a one-to-one correlation for each predefined behavior profile, monitor findings from the SAS and render a recommendation regarding the suitability of

its respective behavior for controlling the vehicle. The Decision Specialist, or Decision Broker, must weigh the findings and recommendations of all SAS and BS and select the most appropriate vehicle operating behavior. The remainder of the chapter chronicles both the evolution of the APF with AGV navigation challenges and the emerging need for specialized sensor resource management systems as mission guidelines require greater autonomous capabilities.

DARPA Grand Challenge

The principle objective of both DARPA Grand Challenges was to prove that autonomous vehicle technologies, including platforms and hardware, were mature enough to handle extended operation along an *a priori* path while negotiating any unknown static obstacles encountered *en route*. Vehicles were expected to operate in an off-road like setting consisting of dirt roads, dry lake beds, tunnels, and mountain passes at prescribed speed limits.

Team CIMAR designed and automated the NaviGATOR, depicted Figure A-1, in response to the previously mentioned operating requirements. The platform, based on a four-wheel-drive rock crawler, was more than suitable for the expected terrain types. A computer enclosure complete with sealed air-conditioning unit and redundant power distribution systems were also added to meet race endurance and platform robustness requirements. A vibration isolated sensor cage was added to house the stereo vision camera system, two cameras used for vision based path finding, two LADAR range finders for terrain and negative obstacle detection, and a front planar LADAR for detecting static obstacles lying in the path. All raw sensor data is processed and converted into a uniform rasterized grid map of traversability, thus facilitating the sensor fusion process.

The fused grid representation shown in Figure A-2 is consumed by the A* receding horizon local path planner previously discussed. Red cells are associated with a high traversability cost while blue cells are the most desirable. Objects persist locally but once rolled

off the grid must be resensed. A limitation of the rasterized grid map approach is it only accommodates a 60 meter by 60 meter world model, which after empirical testing, provided sufficient look ahead to allow for reactivity when traveling at speeds of 20 miles per hour. Designers were mindful not to overburden the computers with unnecessary and excessive data manipulation and transfer.

"The Great Robot Race," as the Grand Challenge was dubbed, although a milestone in the history of AGV research, only required vehicles demonstrate one autonomous capability, navigation. The mission goal was clear, finish the race as quickly as possible by navigating through all points along the path. As a result, first iteration of the APF did not require deep reasoning or behavior arbitration capabilities; only one primary behavior was needed to control the vehicle.

Figure A-3 depicts the component software architecture for Team CIMAR's NaviGATOR. Two Situation Assessment Specialists were developed whose findings were used by the JAUS Subsystem Commander to set the vehicle maximum speed limit. The first specialist, housed in the Planar LADAR component, utilized raw point cloud data from the sensor to provide valuable look-ahead information beyond the 30 meter look ahead provided by the grid map approach. This specialist, although primitive, allowed the vehicle to operate at speeds in excess of twenty miles-per-hour. Upon detecting the presence of a long range obstacle, the specialist would send a message back to the SSC which set the speed limit to the maximum speed limit at which obstacle avoidance was possible. The second specialist, housed within the Velocity State Sensor component, monitored vehicle roll, pitch, and yaw rates and other accelerometer data to estimate the ruggedness of the terrain. These specialists provided an abstraction of processed sensor data to be used at higher level to effectively optimize the mission goal. This rudimentary implantation

of select APF components proved effective for meeting the DARPA Grand Challenge requirements; however, significant improvements have since been made for recent projects which contain more complex missions and require operation in more complex environments.

DARPA Urban Challenge

Building on the tremendous success of the Grand Challenge, Urban Challenge sought to push autonomous vehicle technologies to new level on a scale never before seen. While the primary goal was still to finish the race as quickly as possible, vehicles were required to supplement the primary goal with intermediate goals in the form of local path plans. Mission requirements necessitated a vehicle capable of driving through urban environment while safely interacting with other dynamic vehicles. While the list of behaviors is quite extensive some key tasks included: driving within a lane, obeying speed limits, maintaining safe following distance, passing slow moving or stopped vehicles, merging into traffic, negotiating an intersection, navigating an obstacle field and detecting road blockage, and performing any subsequent course corrections [11]. These requirements demanded a vehicle with modeling, perception, and reasoning capabilities as well as response times akin to those of human driver.

The dynamic nature of the environment required that vehicles possess a more detailed and capable local world model representation. In addition to increasing the map size to 300 meters by 300 meters, the representation also included the road network and desired mission path. Where previous iterations of the world model only required generating a traversability grid of the *a priori* path, the newly developed WMKS was responsible for determining if an object was occupying the travel lane, establishing and maintaining a following distance, sending intermediate goal nodes to the Roadway Navigation component, and a host of other duties detailed in [26]. The increased responsibilities of the world model required it have knowledge of driving rules and sufficient perception data before any actions could be undertaken.

The Grand Challenge realized a revolution in AGV sensing technologies and the Urban Challenge served as the venue for demonstrating their power. Ibeo and Velodyne are just two prominent companies that developed multiline laser scanners for the challenge. In some cases, a single scanner can provide the vehicle with 270-360 degrees of high definition lines scans; however, this technology came with a forty to seventy-five-thousand dollar price tag. Furthermore, the prohibitive price tag and specified mounting configurations limit the design of redundant systems. As an alternative to these high priced sensors vehicles, such as University of Central Florida's Knight Rider, depicted in Figure A-4, and Team Gator Nation's Urban NaviGATOR, depicted in Figure A-5, utilized articulated sensors to expand the field of view. Knight Rider utilizes an articulated radar at intersections to gauge the velocity of oncoming traffic to determine if it is safe to enter an intersection. The following discussion details Team CIMAR's sensor articulation

The Urban NaviGATOR LADAR sensor suite is composed of four fixed laser range scanners and four articulated single degree of freedom laser range finders. Like Knight Rider's actuated radar, these directable laser scanners enhance the vehicle field of view, but are also able to provide higher density point cloud data. Due to time constraints, the front and rear planar LADARs remained in a static configuration. The "vertical fan" scanners located above the driver and passenger side windows were articulated and used for enhanced localization, road edge detection during an n-point-turn, and clearing a desired travel lane before performing a lane change. Sensor allocation and tasking was implicitly performed such that the current prevailing operating behavior dictated the sensors utilization.

The software architecture and dissemination of information onboard Team CIMAR's DUC platform, depicted in Figure A-6, illustrate the extent to which data is shared. All lasers and

actuators are physically connected to a computing node. That node, referred to as the laser control node, is responsible for interfacing with the sensor and actuator, making any necessary hardware setting modifications and the subsequently multicasting the raw data throughout the system via the sensor bus. As indicated in Figure A-6, laser data is consumed by the Moving Object Characterization, Lane Finder Smart Sensor, Terrain Smart Sensor, Path Finder, and a number of APF specialists not depicted in the figure. As a result, the reallocation of a sensor resource towards another task will undoubtedly have downstream consequences. The onus lies with the consumers of the data to determine its validity.

While a detailed explanation of the APF implementation for the Urban Challenge is presented in [26], a terse explanation is now provided. In addressing the challenges of DARPA's new competition, Team CIMAR sought to evolve as much of the proven 2005 architecture as possible. A comparison of Figures A-3 and A-6 attest to this fact. The problem of autonomous navigation in complex and structured environments was decoupled into a finite set of operating behavior and sub-behaviors meticulously tailored to exact explicit behaviors from the AGV to achieve some intermediate goal. The two specialists utilized in the 2005 architecture ballooned to 48 Situation Assessment Specialists and Behavior Specialists, shown in Table A-1, and a Decision Broker, which deliberated on the data. For each SAS and BS envisioned, there exists a corresponding findings worksheet, Figure A-7 depicts the lane change findings worksheet. Likewise, each behavior protocol was designed using a corresponding behavior use case template, see Figure A-8. These tools are utilized at design time to orchestrate APF-related interactions. This methodology, while proven, does not readily support capabilities upgrades and is quite cumbersome.

It was mentioned that sensors and actuators are implicitly managed using the Adaptive Planning Framework, a detailed example is now provided. The AGV is driving in the right-most lane down the street in Roadway Navigation (RN) behavior. During RN the driver side vertical fan laser is assigned to the laser path finder and is utilized for curb detection. During such time, no other component is allowed to redeploy the driver side VF, referred to as a blocking function in computer science. The vehicle continues to navigate down the road until it approaches a stopped car exhibiting excessive delay. Assuming that all conditions are satisfied, the passing specialist recommends the pass left behavior. At this point, the Decision Broker deems relevant the pass left behavior and the vehicle transitions into the pass left operating behavior, lane-change-left sub-behavior. Before the lane change can commence, the protocol requires that lane-change-left-clear specialist, which requires a sweep of the driver side vertical from the front of the vehicle to the back, must report back that the side is clear. Should an object appear in the blind spot, the sub-behavior will fail and the corresponding contingency protocol will execute. However, performing the sweep before transitioning to the behavior could have avoided the failure protocol execution and saved valuable race time. This sweep was not possible earlier due to the blocking nature of the implicit resource allocation scheme via the Adaptive Planning Framework.

A number of multitasking related issues were previously discussed. It should be noted that Touchton's framework does address limited capabilities degradation through the selection of alternative operating behaviors, if a suitable alternative is available. However, resource starvation issues such as deadlock and livelock are not addressed by the APF as it is designed to manage vehicle control and planning issues. These issues are best addressed by the dedicated resource management framework presented here.

Environmental Mapping and Monitoring Demo

CIMAR's most recent research endeavors involve the integration of autonomous navigation and surveillance capabilities onboard a single mobile robot platform. The goal is to enhance the autonomous navigation capabilities set developed for the Urban Challenge to simultaneously support detailed environmental mapping and monitoring while demonstrating safe and effective navigation.

The operating scenario is as follows. An AGV with *a priori* road network knowledge and a predefined patrol region is deployed in an environment about which it has no other information. Utilizing only onboard sensing, perception, and reasoning capabilities the robot will survey the environment and develop a world model representation of all objects, structures, and entities detected to be maintained within a knowledge store. Upon successful characterization of the operating environment, the platform commences execution of patrol mission in which it searches the region looking for perturbances.

Both architectural and hardware changes are needed to facilitate the necessary capabilities extension. At least two new operating behaviors are under development for realizing exploration and patrol of the environment, along with an intercept behavior which handles target interception. The increased mission complexity requires an abstraction of mission operating requirements and rules of engagement to supplement the "race" objective previously defined by the DARPA Challenges.

The Urban NaviGATOR sensor package was outfit with a new articulated camera array shown in Figure A-9, mounted atop the foreword planar LADAR. The camera array will be used to capture images of any unknown objects deemed of interest. These images are then relayed to an operator control unit where a human, operating in a supervisory capacity, can decide any subsequent intervention methods. The Cognitive Resource Management Framework plays a

critical role in supplementing existing autonomous navigation abilities with new fully autonomous RSTA capabilities, leading to a more robust and adaptable solution which enhances the Urban NaviGATOR's autonomy.

Figure A-1 NaviGATOR Team CIMAR's 2005 DARPA Grand Challenge Entry

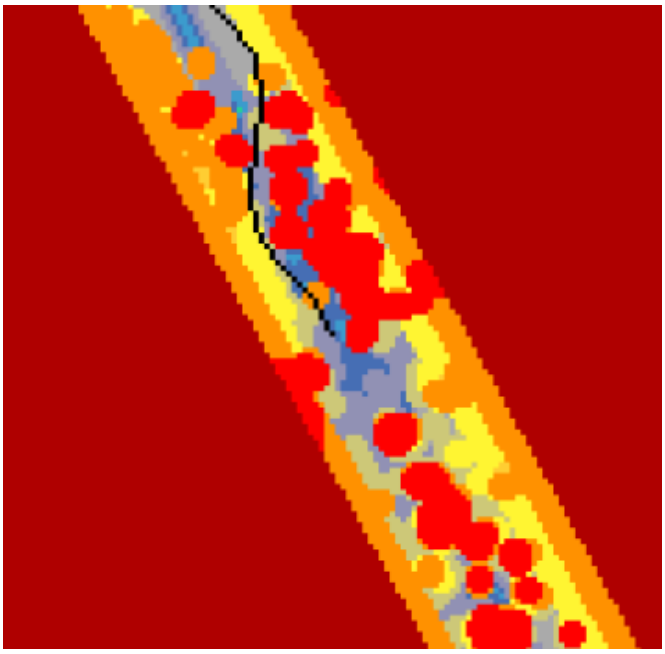


Figure A-2 Local World Model Fused Traversability Grid for 2005 DARPA Grand Challenge [4]

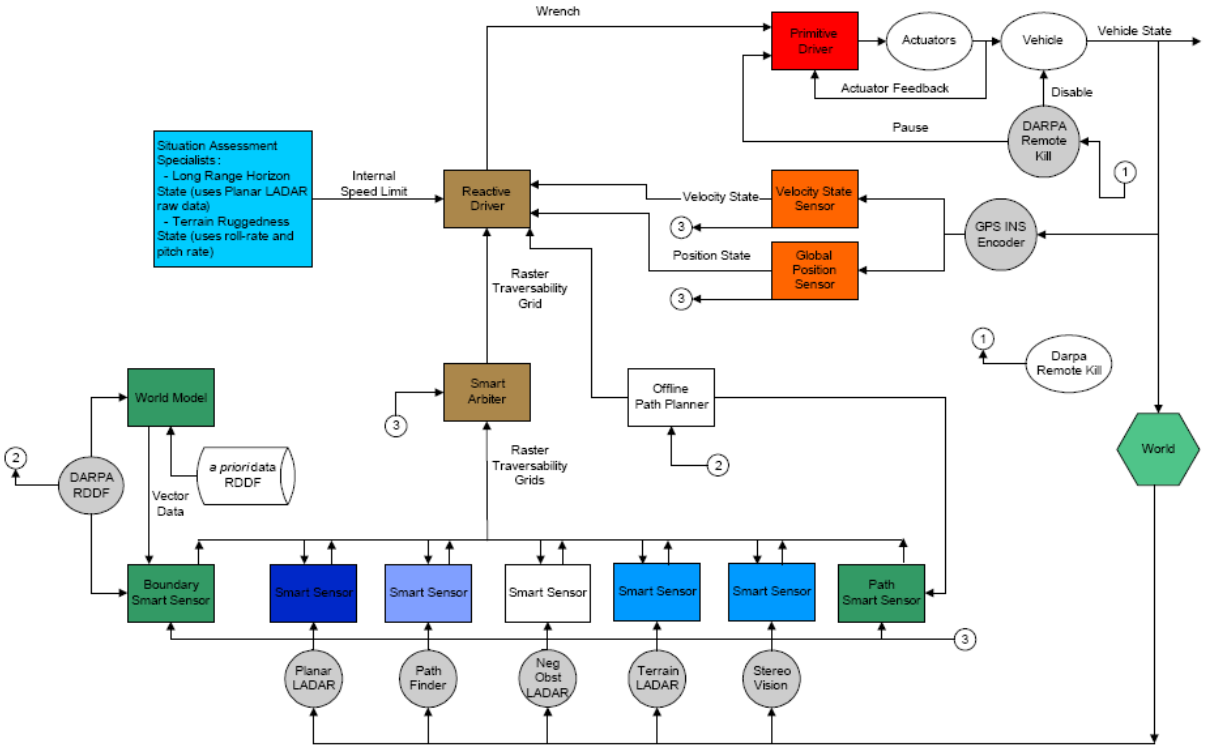


Figure A-3 NaviGator Component Block Diagram for 2005 DARPA Grand Challenge [4]

Figure A-4 Knight Rider Articulated Radar

Figure A-5 Urban NaviGATOR Sensor Articulation (Rear Planar LADAR occluded)

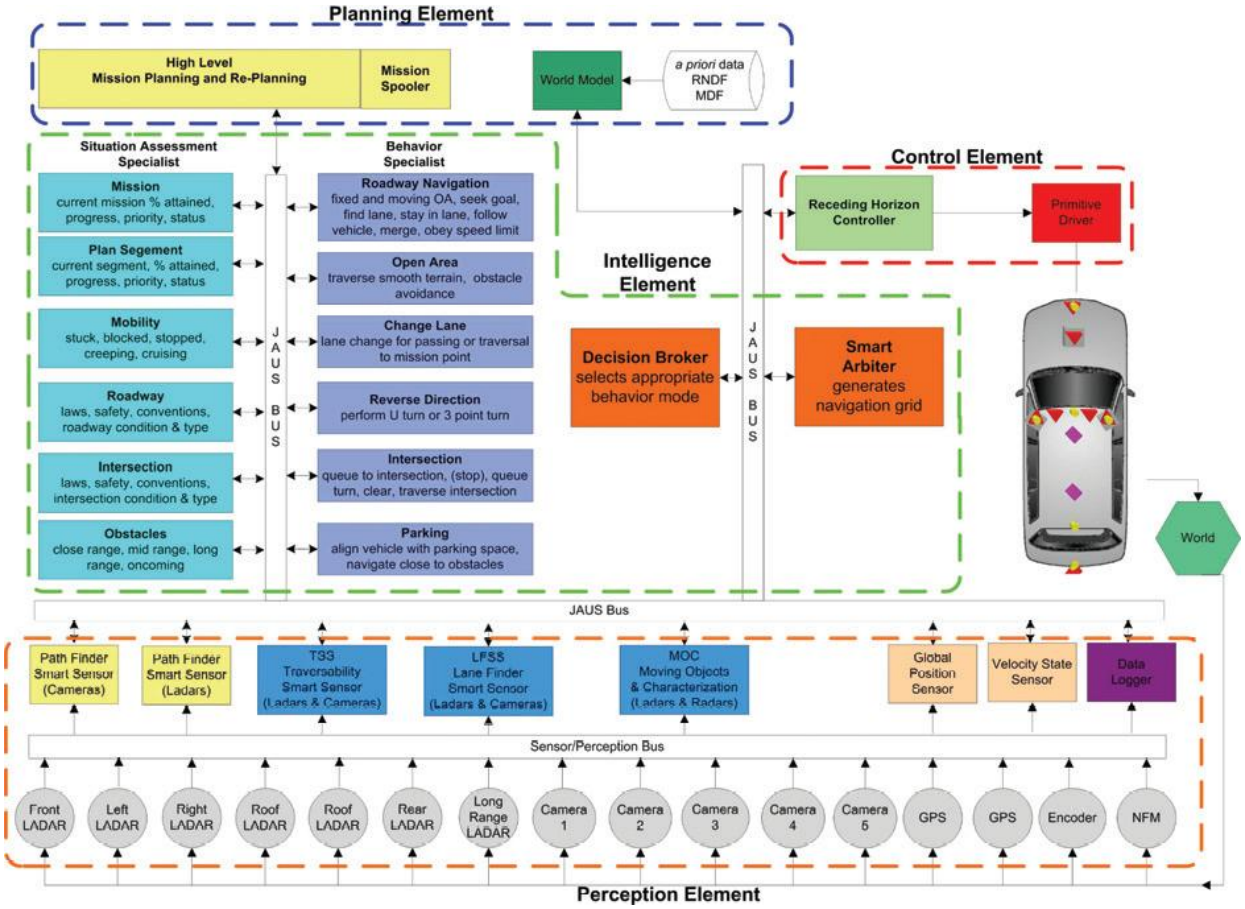
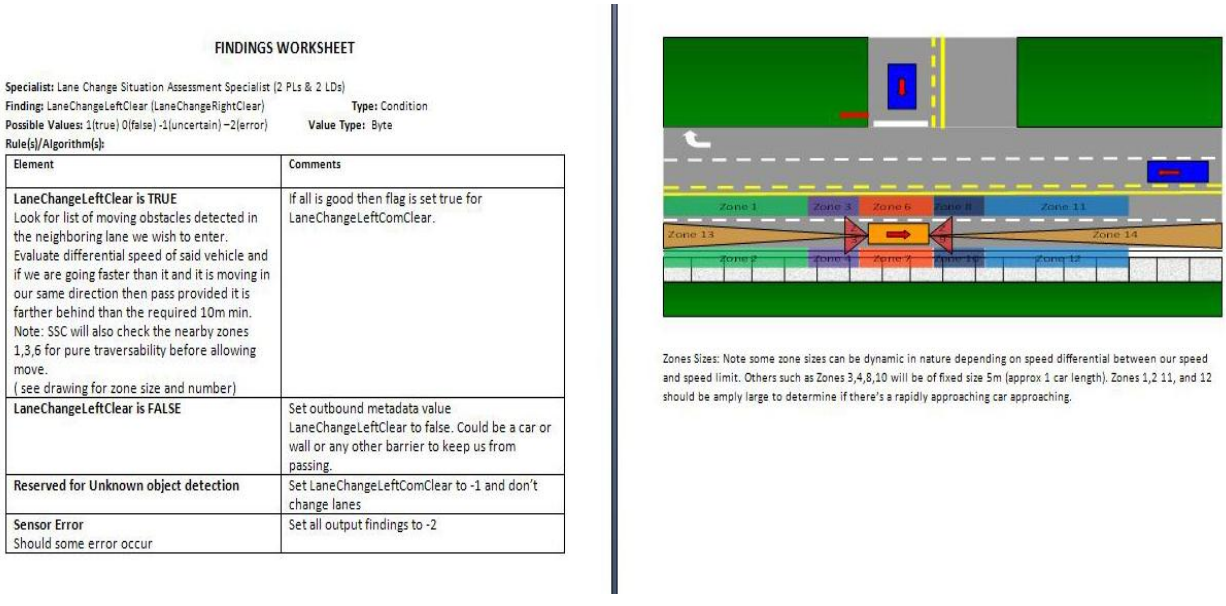


Figure A-6 Urban NaviGATOR Component Block Diagram for the 2007 DARPA Urban Challenge [26]



FINDINGS WORKSHEET

Specialist: Lane Change Situation Assessment Specialist (2 PLs & 2 LDs)
Finding: LaneChangeLeftClear (LaneChangeRightClear) **Type:** Condition
Possible Values: 1(true) 0(false) -1(uncertain) −2(error) **Value Type:** Byte
Rule(s)/Algorithm(s):

Element	Comments
LaneChangeLeftClear is TRUE Look for list of moving obstacles detected in the neighboring lane we wish to enter. Evaluate differential speed of said vehicle and if we are going faster than it and it is moving in our same direction then pass provided it is farther behind than the required 10m min. Note: SSC will also check the nearby zones 1,3,6 for pure traversability before allowing move. (see drawing for zone size and number)	If all is good then flag is set true for LaneChangeLeftComClear.
LaneChangeLeftClear is FALSE	Set outbound metadata value LaneChangeLeftClear to false. Could be a car or wall or any other barrier to keep us from passing.
Reserved for Unknown object detection	Set LaneChangeLeftComClear to -1 and don't change lanes
Sensor Error Should some error occur	Set all output findings to -2

Zones Sizes: Note some zone sizes can be dynamic in nature depending on speed differential between our speed and speed limit. Others such as Zones 3,4,8,10 will be of fixed size 5m (approx 1 car length). Zones 1,2 11, and 12 should be amply large to determine if there's a rapidly approaching car approaching.

Figure A-7 Sample Lane Change Situation Assessment Specialist Finding Sheet

Passing Left/Right Behavior Use Case Template

Phase 1 implementation

Scenario Description: Vehicle is driving down a roadway maintaining a lane and trying to navigate to a goal. A Situation Assessment Specialist detects there is a long range object in our lane. Roadway navigation will default to following behavior primitive. (SSC) If the velocity difference between the objects velocity and our desired velocity is above a certain threshold (configurable based on Risk Level) and the Behavior Specialist recommendation is to pass the SSC may then switch behavior to passing

Assumptions: On a marked, drivable road surface. A valid RNDF and MDF file. Only Sensors are 2x PL fore and aft. No Moving Object Smart Sensor active

Constraints: Obey Rules of Road. Only 1 lane change allowed at a time. Note: If both passLeft and passRight are available the SSC will default to pass left due to rules of the road and safety.

Entry Conditions: Roadway Navigation Behavior Selected; PD, SARB, RN, LSS, DanDan and SSC in ready state;

Exit Conditions: Successful completion of one complete cycle of lane changes (ex. Lane change left, pass vehicle, Lane change right) **OR** when it is no longer safe to perform the passing maneuver.

Inputs Consumed: Passing into oncoming traffic? (DanDan), Passing Situation Assessment Specialist (Lss), Road Rules Situation Assessment Specialist (legal or not), Behavior State of the Vehicle (SSC), Distance to Intersection (DanDan).

Outputs Produced:
DanDan: grid map with legal lane change are indicated marks obstruction in current lane where the car is, new setGlobalMissionPath message with points in the new desired lane.
SARB: arbitrates grid map differently, paints new lane more desirable than current lane. Paints transition area over lane marking with a more favorable traversability value.

Steps for Passing Behavior (Left/Right):

Step#	Action	Contingency Action
1	laneChangeLeft(right) sub-behavior	Abort passing maneuver and remain in current lane
2	passVehicle sub-behavior	Abort passing maneuver, apply brakes and perform opposite lane change maneuver to that done in step 1.
3	laneChangeRight(left) then behavior completed	Abort passing maneuver, apply brakes, slow down behind car and perform opposite lane change maneuver to that done in step 1 when cleared.
	Note: if any of these Actions Fail the behavior specialist recommendation will go to failure and it will follow the contingency action indicated with SSC approval	

Figure A-8 Sample Passing Behavior Use Case Sheet

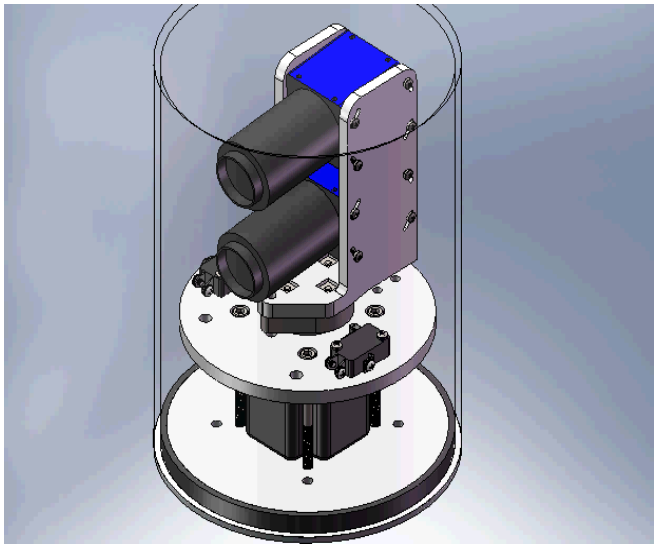


Figure A-9 Articulated Target Tracking Cameras

Table A-1 APF Situation Assessment and Behavior Specialists

Inbound MetaData (with element names, types and publishing component id's)

rnMobilityState Byte RN	rnPlanningState Byte RN	rnRecommendation Byte RN	nptRecommmendation Byte NPT	nptRecommendation Byte NPT	sarbStateMDE Byte SARB	dandanStateMDE Byte DanDan
dandanEnvStateMDE Byte DanDan	nptStateMDE Byte NPT	pdStateMDE Byte PD	rnStateMDE Byte RN	hlpStateMDE Byte HLP	tssStateMDE Byte TSS	lpfssStateMDE Byte LPFSS
lfssArbiterStateMDE Byte LFSSARB	laneChangeLeftClear Byte TSS	laneChangeRightClear Byte TSS	laneChangeLeftRec Byte DanDan	laneChangeRightRec Byte DanDan	passingOncomingTraffic Byte DanDan	crossingDoubleYellowLine Byte DanDan
passLeftRecommendation Byte DanDan	passRightRecommendation Byte DanDan	hlpPlanningState Byte HLP	forwardLeftSafe Byte HLP	rearRightSafe Byte TSS	rearStraightSafe Byte TSS	nptRequest Byte DanDan
intersectionRecommendation Byte DanDan	openAreaRecommendation Byte DanDan	tssForwardCollision Byte TSS	availableParkingSpace Byte DanDan	laneChangeLeftRequest Byte DanDan	laneChangeRightRequest Byte DanDan	followingObject Byte DanDan
excessiveDelay Byte DanDan	approachingStop Byte DanDan	safeZone Byte DanDan	intExcessiveDelay Byte DanDan	missionSegmentSpeedRange Unsigned Tuple DanDan	passingObjectState Byte DanDan	runPauseState Byte PD
twenty5mFollowedObject Byte DanDan	offRoadEnv Byte DanDan	numTravelLanes Byte DanDan	sol Byte DanDan	hlpPlanPercent Byte HLP	maximumTravelSpeed Double DanDan	

APPENDIX B
REFERENCE IMPLEMENTATION FRAMEWORK TOOLS GLOSSARY OF TERMS

Actuator:	Subset of the Resource set with the most abstract and generalized classification of a hardware element utilized for producing motion that can be described and managed using the CRMF
Actuator Type:	Numerical identifier correlating the actuator to an *a priori* defined classification of allowable actuator types e.g. linear, rotary
Angular Resolution:	Describes the angular distance between scan lines emitted from a laser measurement device; units in degrees
Beam Width:	Angular distance, in degrees, between half-power points in a plane passing through a beams centerline.
Camera:	Specific subset of Sensor class containing resource specific knowledge abstraction of vision-based sensor
Camera Type:	Numerical identifier correlating the camera sensor to an *a priori* defined classification of allowable camera types e.g. RGB, HSV
Dependent Resource Name:	String field containing unique Resource Name of tightly coupled or dependent resource, e.g. panning actuator for a sensor
Duration:	Numerical representation of desired time job execution time, relative time measurements must be consistent throughout framework; must be specified for advanced reservation along with Start Time and End Time, optional specification for other Job Requests.
Elevation:	Initial elevation in meters above sea level corresponding to World Model Object identified by Object ID; populated by Application Analyst when job request is created
End Time:	Numerical representation of desired time for a job to be completed, relative time measurements must be consistent throughout framework; must be specified for advanced reservation along with Start Time and Duration
Field of View:	Characterization of a sensor's observable area, measure overall planar view angle in degrees and assumes symmetric distribution about the sensing axis; for cases where FOV is

	characterized by multiple angles, the minimum angle should be reported to be utilized by the DSA
Job Request:	Framework defined representation of task to be performed by a managed resource; Requests are sent from Application Analyst to Plant Engineer
Job Number:	Unique numerical Job identifier assigned to each new job by the Plant Engineer
Job Type:	Numerical identifier correlating job to an *a priori* defined classification of jobs
Laser:	Specific subset of Sensor class containing resource specific knowledge abstraction of line scan time-of-flight-based measurement sensor
Laser Type:	Numerical identifier correlating the laser sensor to an *a priori* defined classification of allowable laser type e.g. single line scan
Latitude:	Initial latitude value corresponding to World Model Object identified by Object ID in decimal degrees; populated by Application Analyst when job request is created
Longitude:	Initial longitude value corresponding to World Model Object identified by Object ID in decimal degrees; populated by Application Analyst when job request is created
Make:	String field containing resource make information, populated by RA and used by DSA for gathering resource specifications from an onboard database, if so equipped
Max Acceleration:	Describes the maximum controllable acceleration of the actuator; units of degrees per second squared or meters per second squared for rotary or linear actuators respectively, used by DSA for system modeling
Max Velocity:	Describes the maximum controllable velocity of the actuator; units of degrees per second or meters per second for rotary or linear actuators respectively, used by DSA for system modeling
Measurement Max Range:	Describes the maximum distance in meters from a sensor where observations are valid; measurements with respect to x axis of fixed sensor coordinate system

Measurement Min Range:	Describes the minimum distance in meters from a sensor where observations are valid; measurements with respect to x axis of fixed sensor coordinate system
Measurement Resolution:	Describes the minimum detectable change between measurements for a sensing device; units in meters
Min Acceleration:	Describes the minimum controllable acceleration of the actuator; units of degrees per second squared or meters per second squared for rotary or linear actuators respectively, used by DSA for system modeling
Min Velocity:	Describes the minimum controllable velocity of the actuator; units of degrees per second or meters per second for rotary or linear actuators respectively, used by DSA for system modeling
Model:	String field containing resource model information, populated by RA and used by DSA for gathering resource specifications from an onboard database, if so equipped
Object ID:	Locally unique numerical identifier assigned by World Model Knowledge Store upon object creation
Object Reactivity:	Numerical identifier characterizing degree of dynamicity of object identified by Object ID relative to an *a priori* defined degrees of dynamicity
Radar:	Specific subset of Sensor class containing resource specific knowledge abstraction of pulse radio energy time-of-flight-based measurement sensor
Range of Motion:	Describes the extent of motion of a linear or rotary actuator given a stated measurement reference; units in degrees or meters for rotary or linear actuators respectively
Reconfiguration Time:	Downtime, measured in seconds, during which a resource is inoperable due to a change in configuration parameters
Resolution:	Defines the minimum controllable increment for an actuator; units in degrees or meters for rotary or linear actuators respectively
Resource:	Most abstract and generalized classification of a hardware element that can be described and managed using the CRMF

Resource Description Language:	Terminology utilized to provide an unambiguous description of resource capabilities during discover by Diagnostician/Systemizer Analyst
Resource ID:	Unique Resource identifier assigned by Plant Engineer; utilized in Job Request for explicitly specifying resource for job fulfillment
Resource Name:	Unique string representation defined by the Resource Analyst at design time; utilized in Job Request for explicitly specifying resource for job fulfillment and used during discovery to uniquely identify and model the resource in the DSA
Resource Type:	Numerical identifier correlating the category of resource to an *a priori* defined classification of resource types e.g. Sensor, Actuator
Sensor:	Subset of the Resource set with most abstract and generalized classification of a hardware element utilized for sensing the environment that can be described and managed using the CRMF
Sensor Type:	Numerical identifier correlating the sensor to an *a priori* defined classification of allowable sensor types e.g. camera, ladar or radar
Source Component ID:	JAUS component ID of Application Analyst that created Job Request; used by Plant Engineer job value estimation algorithm
Start Time:	Numerical representation of desired time for a job to commence, relative time measurements must be consistent throughout framework; must be specified for advanced reservation along with End Time and Duration
Text Description:	String descriptor transmitted by Resource Analyst with resource description for debugging and to support supervisory control
Unique ID:	Unique numerical identifier assigned during resource discovery by the Plant Engineer
Update Rate:	Nominal data update rate in cycles per second
View Angle Diagonal:	Describes the angular extent a given scene is imaged by a camera, assumes rectilinear images; measured diagonally from one corner of the frame to the opposite corner

View Angle Horizontal:	Describes the angular extent a given scene is imaged by a camera, assumes rectilinear images; measured horizontally from the left to right edge of the frame
View Angle Vertical:	Describes the angular extent a given scene is imaged by a camera, assumes rectilinear images; measured vertically from the bottom to top edge of the frame
X Offset:	Relative x offset in meters from the vehicle fixed coordinate frame
X Unit Vector Component:	x component of unit direction vector representing orientation of the axis of rotation/translation of an actuator or sensing reference frame
Y Offset:	Relative y offset in meters from the vehicle fixed coordinate frame
Y Unit Vector Component:	y component of unit direction vector representing orientation of the axis of rotation/translation of an actuator or sensing reference frame
Z Offset:	Relative z offset in meters from the vehicle fixed coordinate frame
Z Unit Vector Component:	z component of unit direction vector representing orientation of the axis of rotation/translation of an actuator or sensing reference frame

APPENDIX C
REFERENCE IMPLEMENTATION FRAMEWORK TOOLS

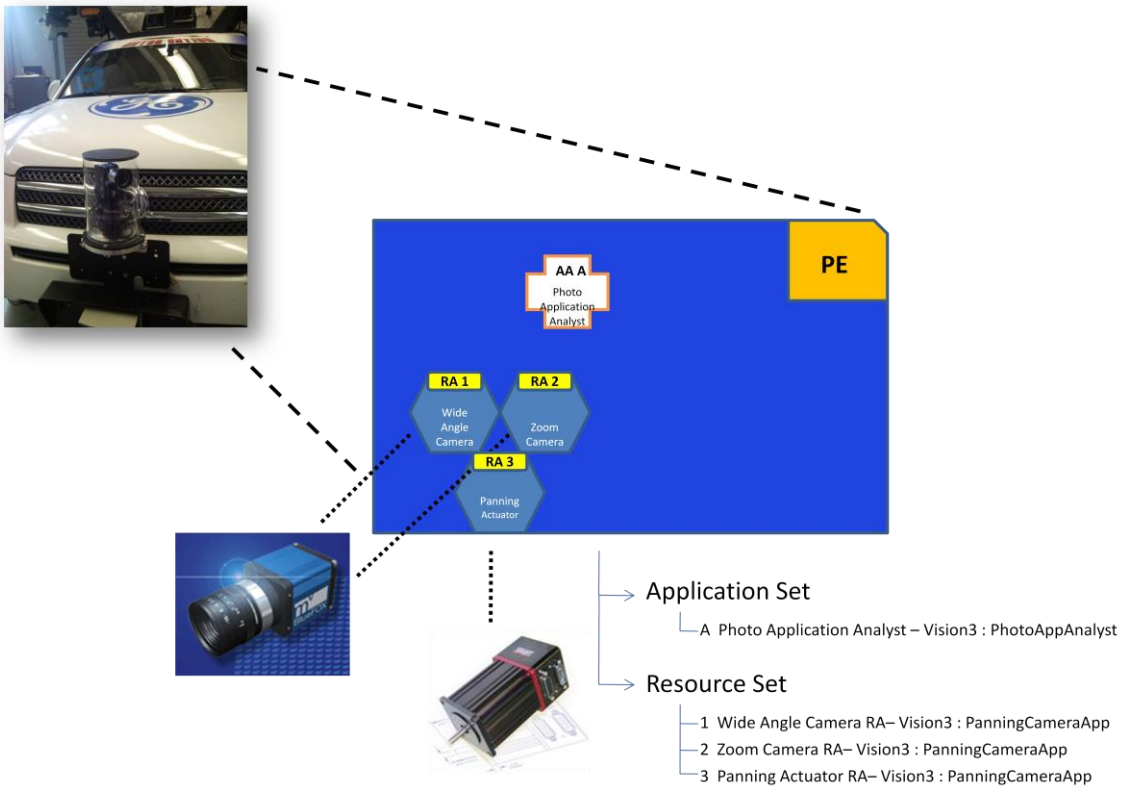


Figure C-1 CRMF Reference Implementation Resource Virtualization

Resource Attributes Worksheet

This worksheet is an engineering design tool for specifying a resource attribute that describes resource capability within the Cognitive Resource Management Framework.

Attribute Name: Camera Type

Attribute Type: capability/performance **Type:** capability

Parent Resource Name: Camera

Attribute Description: Numerical identifier correlating the camera sensor to an apriori defined classification of allowable camera types

Intended Use: Allows the model do differentiate between the type of information the resource is able to provide for proper job assignment

Vehicle Identifier: CIMAR Urban NaviGator

DSA Model description: 3D spatiotemporal model. See DSA worksheet for more details

Update Frequency: Rate/Event **Rate:** N/A Hz

(if event)

Event Trigger	Description
On change	Attribute is updated whenever a change in the configuration is detected

Data Contents: Value/Enumeration/String/Other Type: unsigned char

(if enumeration: populate apriori mapping of values)

Enumeration	Interpretation/Significance
1	RGB color space camera
2	HSV camera
3	Infrared camera
4	Night vision camera

(if other: include schema or other data structure illustration)

Rev b. updated 1/22/10

Figure C-2 Resource Attribute Definition Worksheet

Job Request Attribute Definition Worksheet

This worksheet is an engineering design tool for specifying a new Job Request attribute that describes a task within the Cognitive Resource Management Framework.

Attribute Name: Job Type

Attribute Type: Capability Request/Target Attribute Type: Cap. Req.

Parent Analyst Name: K.S. Application Analyst

Host Application Name: Knowledge Store Front-End

Attribute Description: Numerical identifier correlating the job to an apriori defined classification of allowable job types

Intended Use: Allows the model to determine which the types of activities and resources needed to fulfill this request

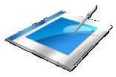
Vehicle Identifier: Urban NaviGATOR

Data Contents: Value/Enumeration/String/Other Type: _____

(if enumeration: populate apriori mapping of values)

Enumeration	Interpretation/Significance	Algorithm(s)/Rule(s)
1	Image Capture	Requires sensor of type Camera
2	Distance measurement	Requires sensor of type Laser or Radar

(if other: include schema or other data structure illustration)



Rev a. updated 1/13/10

Figure C-3 Job Request Attribute Definition Worksheet

Meta-Knowledge Element Definition Worksheet

This worksheet is an engineering design tool for defining Meta-knowledge elements which support resource management initiatives within the Cognitive Resource Management Framework.

Element Name: Vehicle Position

Parent Analyst Name: Communicator Analyst

Element Description: This element contains the georeferenced location of the vehicle platform

Intended Use: Support DSA spatiotemporal model

Vehicle Identifier: Urban NaviGATOR

Update Frequency: Rate/Event **Rate:** N/A Hz

(if event)

Event Trigger	Description

Data Contents: Value/Enumeration/String/Other Type: Other

(if enumeration: populate apriori mapping of values)

Enumeration	Interpretation/Significance	Algorithm(s)/Rule(s)
N/A	N/A	N/A

(if other: include schema or other data structure illustration)
Data type Name: LLA

Field	Name	Data Type	Significance
1	Latitude	double	Georeferenced lattidue in decimal degrees
2	Longitude	double	Georeferenced longitude in decimal degrees
3	Elevation	double	Elevation above see level in meters

Rev a. updated 1/13/10

Figure C-4 Meta-Knowledge Element Definition Worksheet

Resource Object Description Worksheet

This worksheet serves as an engineering design tool for identifying a set of resource attributes for describing a specific resource's capabilities.

Class Name: Camera

Parent Class: Sensor

Class Description: The Camera class is the most resource specific class from which a camera sensing resource can be defined. This class contains sensor attributes embodied by the most common camera components. Attributes defined in this class can be expanded and modified as needed.

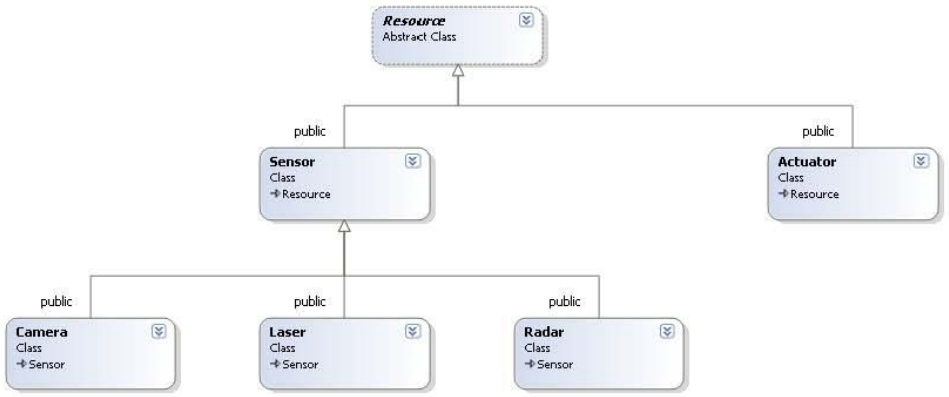
Vehicle Identifier: Urban NaviGATOR

DSA Model: Spatiotemporal sensor model

Derived Resources Supported:

Type	Trace
Matrix Vision Blue FOX 120aC – USB 2.0 RGB camera 640x480	Resource->Sensor
Matrix Vision Blue FOX 121C – USB 2.0 RGB camera 1024x768	Resource->Sensor

Illustrations:

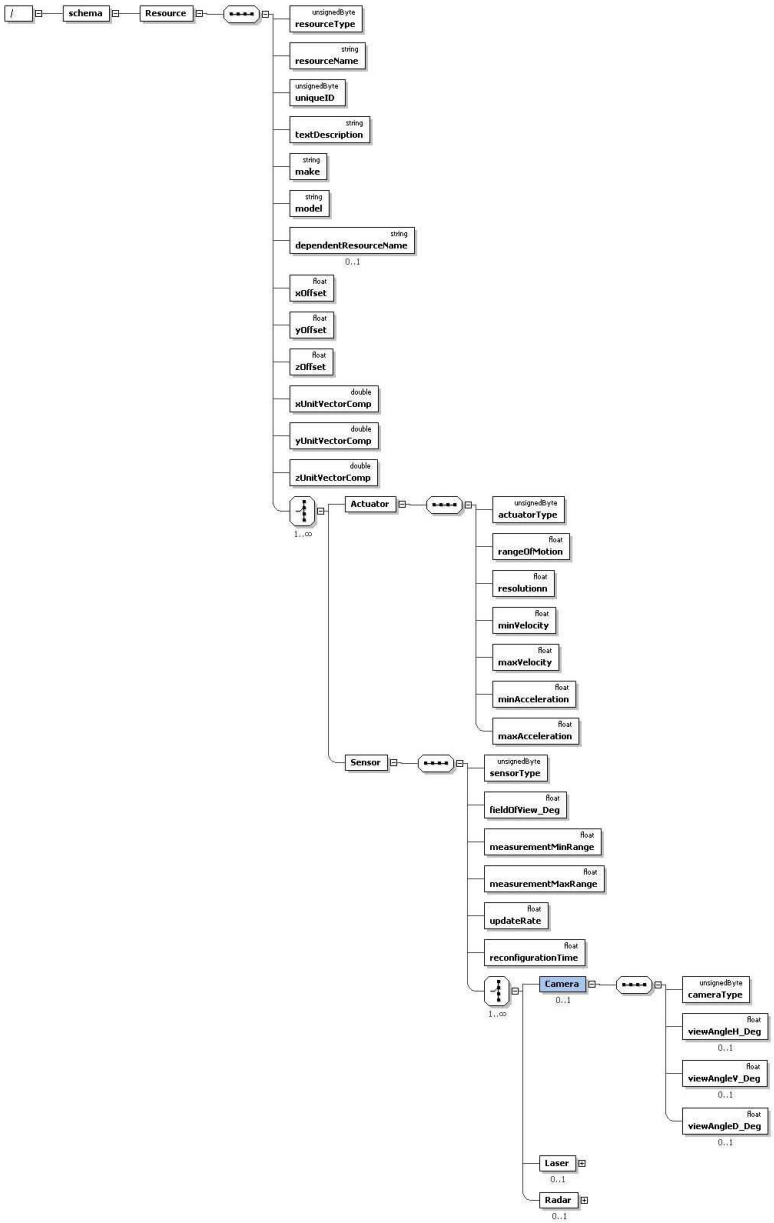


Rev b. updated 1/22/10

Figure C-5 Resource Description Worksheet

Resource Object Description Worksheet

XML Schema:



Rev b. updated 1/22/10

Figure C-5 Continued.

Job Request Definition Worksheet

Note: This worksheet is an engineering design tool for defining the elements that comprise a job request utilized by the Cognitive Resource Management Framework.

Vehicle Identifier: CIMAR Urban Navigator

Setting Overview: Mapping and change detection of a predefined area by autonomous platform to support human-in-the-loop situational awareness. As changes are detected an Application Analyst submits job requests for image for an object of interested detected by some other means.

Job Type: Image Capture

Resource Specification: Supported/Not Supported

Advanced Reservation: Supported/Not Supported

Target Type: point/ area/ other

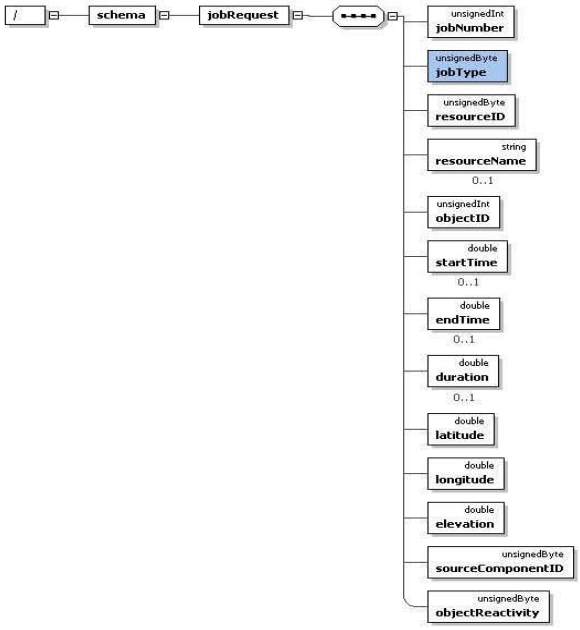
Attributes Name	Attribute Type
jobNumber	Capability Request
jobType	Capability Request
resourceID	Capability Request
resourceName	Capability Request
startTime	Capability Request
endTime	Capability Request
duration	Capability Request
objectID	Target
centroid location (LLA)	Target
object reactivity	Target

Rev b. updated 1/13/10

Figure C-6 Job Request Definition Worksheet

Job Request Definition Worksheet

XML Schema:



Rev b. updated 1/13/10

Figure C-6 Continued.

Resource Appraiser Analyst Worksheet

Note: This worksheet is an engineering design tool identifying areas of system performance evaluation monitored by the Cognitive Resource Management Framework Resource Appraiser Analyst.

Vehicle Identifier: CIMAR Urban Navigator

Overview: Plant Engineer receives a Job Request and assigns a value to the job based on a weighting algorithm based on 4 criteria, job are then placed into PE job queue sorted by value. PE brokering algorithm attempts to match jobs to available resources based on feedback

Model Inputs:

Input	Source
Resource Object (discovery)	Resource Analyst
Resource Object (change updates)	Resource Analyst
Job Request (submission)	Application Analyst
Job Request (assignment)	Plant Engineer
Job Status (completion)	Resource Analyst

Outputs Type: Tab delimited log file

Output Format: (Sample Output)

```
10-23-2009 16:09:30:  NewResource  Actuator  PanningCameraActuator  1  Front
panning camera actuator  PhidgetStepper 123456789  2  180.000000
        0.056250  45.000000  180.000000  0.000000  0.000000  3.898900
        0.000000  0.000000  0.000000  0.000000  0.000000
10-23-2009 16:09:59:  UpdatedResource  Actuator  PanningCameraActuator  1
        Front panning camera actuator  PhidgetStepper 123456789  2  180.000000
        0.056250  45.000000  180.000000  0.000000  0.000000  3.898900
        0.000000  0.000000  0.000000  0.000000  3.000000
10-23-2009 16:09:58:  NewJob1  1  0  13.000000  0  0.000000
        0.000000  0.000000  29.756858  -82.267772
10-23-2009 16:09:58:  JobAssigned  1  zoomPanningCameraSensor  1  2  0
        0  0  0
10-23-2009 16:09:59:  JobComplete  1  1
```

Parameter Analysis: Real-time/Offline

Performance Metric	Algorithm(s)
Jobs received	Summation of inbound jobs
Jobs assigned	Summation of successfully allocated jobs
Jobs completed successfully	Summation of successfully completed jobs
Jobs aborted	Summation of aborted jobs
Removed jobs	Summation of removed jobs
Job completion time	Time finished – Time assigned
DSA processing time	Time finished processing – Time started processing

Rev b. updated 11/10/09

Figure C-7 Resource Appraiser Analyst Performance Monitoring Worksheet

Communicator Analyst Worksheet

Note: This worksheet is an engineering design tool outlining interactions between the Cognitive Resource Management Framework and external system architectures.

Vehicle Identifier: CIMAR Urban Navigator

Overview: The communicator is responsible for interfacing with existing JAUS components to import data that is available via a JAUS defined message. Additionally, this analyst must also interact with Adaptive Planning Framework elements as both a consumer of Metadata and as a publisher of a behavior recommendation.

Parameters to import:

Parameter	Source	Format	Use
Vehicle Operating Behavior	Decision Broker	APF Metadata	PE- job prioritization
Vehicle Operating Sub-Behavior	Decision Broker	APF Metadata	PE- job prioritization
Vehicle Operating Sub-Behavior Status	Decision Broker	APF Metadata	PE- job prioritization
Global Pose	GPOS Component	JAUS Report Global Pose message	DSA- modeling resource coverage in Global Reference Frame
Object attributes	World Model Knowledge Store	JAUS Query World Model Object Metadata	PE - job prioritization and job updating

Parameters to Export:

Parameter	Destination	Format
Loitering Behavior Specialist Recommendation	Decision Broker	APF Metadata

Rev b. updated 11/5/09

Figure C-8 Communicator Analyst Information Control Worksheet

Diagnostician Systemizer Analyst Worksheet

Note: This worksheet is an engineering design tool for identifying the characteristics used by the Cognitive Resource Management Framework for modeling the virtual plant capabilities.

Vehicle Identifier: CIMAR Urban Navigator

DSA Model Description: DSA constructs a spatiotemporal model of current and possible sensor coverage based on reported sensor properties and coupling to dependent resources. Takes a job request as input and searches through collection of resources to determine first which resources can fulfill the job. It then populates a candidate resource list with viable options. The DSA then checks those possibilities against its coverage model to determine which of the available resources is currently most capable of fulfilling the task, given availability.

Model Inputs:

Input	Source
Position and orientation (reference from dependent resource coordinate system)	Resource Analyst
Resource Type	Resource Analyst
Resource configuration attributes: FOV, Resolution, joint velocities, etc.	Resource Analyst OR apriori DSA database
Job Request	Application Analyst
Job-Resource association list	Apriori DSA database

Outputs: Candidate Resource list of possible resources that can fulfill job based on job type compatibility and resource coverage. Resource identified by unique Resource ID number along with estimate of time till ready (used in case actuation is necessary to achieve coverage).

Model Details:

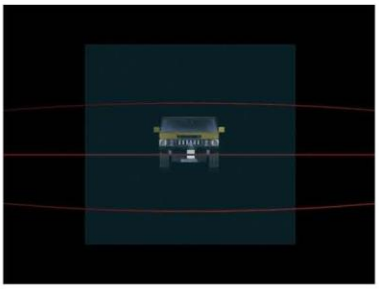
Algorithm	Assumptions
Spatial Coverage- construct polyhedron in vehicle reference frame taking into account dependent resource position and orientation, update model with multiple configurations if necessary	Sensor coverage volume modeled as Frustum, or convex Hexahedron approximation, see illustration
Temporal- if actuation is necessary calculate extent of required motion, compute time till ready estimate using min velocity	Worst case scenario estimate of time till ready guarantees that resource will be ready within predicted time

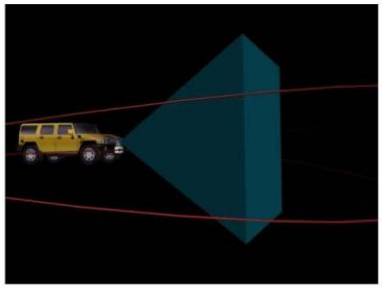
Rev b. updated 11/5/09

Figure C-9 Diagnostician/Systemizer Analyst Worksheet

Diagnostician Systemizer Analyst Worksheet

Illustrations:





Rev b. updated 11/5/09

Figure C-9 Continued.

Plant Engineer Worksheet

Note: This worksheet is an engineering design tool for defining the brokering strategy utilized by the Cognitive Resource Management Framework Plant Engineer.

Vehicle Identifier: CIMAR Urban Navigator

Overview: Plant Engineer receives a Job Request and assigns a value to the job based on a weighting algorithm based on 4 criteria, job are then placed into PE job queue sorted by value. PE brokering algorithm attempts to match jobs to available resources based on feedback

Model Inputs:

Input	Source
Object Reactivity	WMKS object metadata via Communicator Analyst
Distance to point or region of interest	Job Request provides georeferenced absolute location and JAUS GPOS message provides vehicle position via Communicator Analyst
Vehicle Situational Information	Adaptive Planning Framework via Communicator Analyst
Job Request	Application Analyst
Privileged User status	Diagnostician Systemizer Analyst apriori knowledge
Candidate Resource List	Diagnostician Systemizer Analyst

Outputs: Jobs assigned to resource job queue with Start Time and End Time fields populated by the DSA. At the appropriate time, the PE sends jobs from job queue to the appropriate Resource Analyst(s).

Protocol Execution: Upon receipt of the job request from PE, control authority for resources is granted to the application analyst for the time period specified in the request.

Algorithm	Assumptions
Assign Value to job: $Job\,Value = K_{situation} * \alpha + K_{distance} * \beta + K_{reactivity} * \gamma + K_{priveleged\,user} * \delta$	All gains are positive
Assign jobs to available resources	Jobs can be completed by resources in time allotted

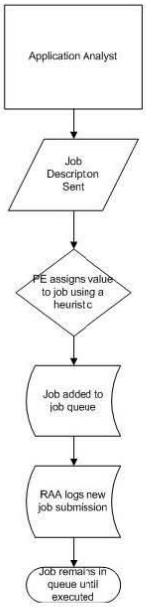
Rev b. updated 11/5/09

Figure C-10 Plant Engineer Worksheet

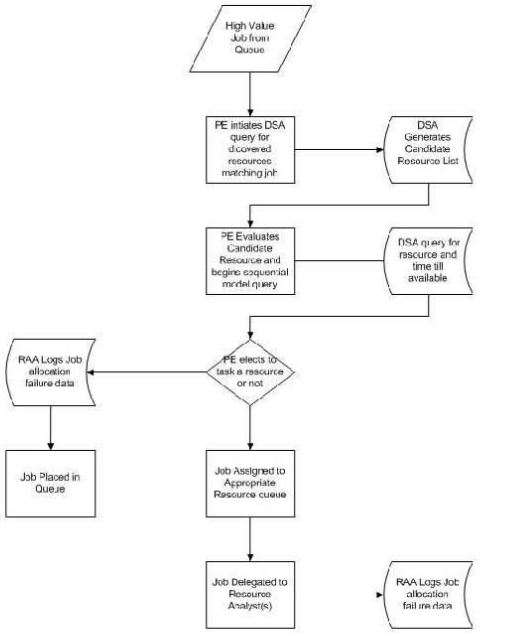
Plant Engineer Worksheet

Illustrations:

Cognitive Resource management Framework Job Submission Process



Plant Engineer Resource Brokering



Rev b. updated 11/5/09

Figure C-10 Continued.

APPENDIX D
FIELD TESTING SAMPLE RESOURCE APPRAISER ANALYST LOG

CIMAR CRMF Resource Appraiser Log -- 12-06-2009 16:00:38

12-06-2009 16:00:58: NewResource Actuator PanningCameraActuator 1 Front panning camera actuator PhidgetStepper
123456789 2 180.000000 0.056250 45.000000 180.000000 0.000000
0.000000 3.898900 0.000000 0.000000 0.000000 0.000000 0.000000

12-06-2009 16:00:58: NewResource Camera zoomPanningCameraSensor 2 Front panning Zoom
camera MatroxVision mvBlueFox120aC_BF000865 PanningCameraActuator 1 17.059999
0.000000 0.000000 15.000000 60.000000 10.000000 0.000000
0.000000 0.000000 -0.100000 1.000000 0.000000 0.000000 0
17.059999 13.110000 0.000000

12-06-2009 16:00:58: NewResource Camera widePanningCameraSensor 3 Front panning Wide
camera MatroxVision mvBlueFox121C_BF001120 PanningCameraActuator 1 63.900002
0.000000 0.000000 0.500000 15.000000 10.000000 0.000000
0.000000 0.000000 -0.045000 1.000000 0.000000 0.000000 0
63.900002 49.099998 76.900002

12-06-2009 16:21:19: NewJob 1 1 0 8.761896 1 0.000000
0.000000 0.000000 29.641186 -82.351650

12-06-2009 16:21:19: JobAssigned 1 zoomPanningCameraSensor 1 2 0 0
0 0

12-06-2009 16:21:19: NewJob 2 1 0 8.709227 2 0.000000
0.000000 0.000000 29.641189 -82.351739

12-06-2009 16:21:19: NewJob 3 1 0 8.687555 3 0.000000
0.000000 0.000000 29.641196 -82.351782

12-06-2009 16:21:19: NewJob 4 1 0 8.305409 4 0.000000
0.000000 0.000000 29.641531 -82.352317

12-06-2009 16:21:19: NewJob 5 1 0 10.350150 5 0.000000
0.000000 0.000000 29.641668 -82.351854

12-06-2009 16:21:19: NewJob 6 1 0 10.297049 6 0.000000
0.000000 0.000000 29.641704 -82.351859

12-06-2009 16:21:19: NewJob 7 1 0 8.103398 7 0.000000
0.000000 0.000000 29.642180 -82.352292

12-06-2009 16:21:19: NewJob 8 1 0 8.085177 8 0.000000
0.000000 0.000000 29.642385 -82.352060

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 5

12-06-2009 16:21:19: NewJob 9 1 0 8.441306 9 0.000000
0.000000 0.000000 29.642228 -82.351904

12-06-2009 16:21:19: NewJob 10 1 0 9.005350 10 0.000000
0.000000 0.000000 29.642133 -82.351604

12-06-2009 16:21:19: NewJob 11 1 0 13.013770 11 0.000000
0.000000 0.000000 29.641837 -82.351553

12-06-2009 16:21:19: NewJob 12 1 0 9.112381 12 0.000000
0.000000 0.000000 29.642082 -82.351363

12-06-2009 16:21:19: NewJob 13 1 0 8.926373 13 0.000000
0.000000 0.000000 29.642035 -82.351188

12-06-2009 16:21:19: NewJob 14 1 0 8.613336 14 0.000000
0.000000 0.000000 29.641131 -82.351516

12-06-2009 16:21:19: NewJob 15 1 0 8.686091 15 0.000000
0.000000 0.000000 29.641178 -82.351726

12-06-2009 16:21:19: NewJob 16 1 0 8.441768 16 0.000000
0.000000 0.000000 29.641199 -82.351992

12-06-2009 16:21:19: NewJob 17 1 0 8.101766 17 0.000000
0.000000 0.000000 29.642143 -82.352325

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 11

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 5

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 6

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 12

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 10

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 13

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 2

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 3

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 15

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 14

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 16

12-06-2009 16:21:19: Could not assign Job, pushing to back of queue Job No: 9

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 4

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 7

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 17

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 8

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 11

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 5

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 6

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 12

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 10

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 13

12-06-2009 16:21:20: JobComplete 1 1

12-06-2009 16:21:20: UpdatedResource Actuator PanningCameraActuator 1 Front panning camera actuator PhidgetStepper 123456789 2 180.000000 0.056250 45.000000 180.000000 0.000000 0.000000 3.898900 0.000000 0.000000 0.000000 0.000000 -47.000000

12-06-2009 16:21:20: JobAssigned 2 zoomPanningCameraSensor 1 2 0 0 0 0

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 3

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 15

12-06-2009 16:21:20: NewJob 18 1 0 8.431002 19 0.000000 0.000000 0.000000 29.641211 -82.352016

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 11

12-06-2009 16:21:20: NewJob 19 1 0 8.070373 20 0.000000 0.000000 0.000000 29.642062 -82.352411

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 11

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 5

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 6

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 12

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 10

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 13

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 3

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 15

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 14

12-06-2009 16:21:20: JobComplete 2 1

12-06-2009 16:21:20: UpdatedResource Actuator PanningCameraActuator 1 Front panning camera actuator PhidgetStepper 123456789 2 180.000000 0.056250 45.000000 180.000000 0.000000 0.000000 3.898900 0.000000 0.000000 0.000000 0.000000 -39.000000

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 16

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 9

12-06-2009 16:21:20: NewJob 20 1 0 8.617713 22 0.000000 0.000000 0.000000 29.641192 -82.351836

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 11

12-06-2009 16:21:20: JobAssigned 5 zoomPanningCameraSensor 1 2 0 0 0 0

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 6

12-06-2009 16:21:20: Could not assign Job, pushing to back of queue Job No: 12

LIST OF REFERENCES

[1] S. Nof, *Handbook of Industrial Robotics*, 2nd ed.: John Wiley & Sons, 1999.

[2] H. Huang, "Autonomy Levels for Unmanned Systems (ALFUS) Framework Volume II: Framework Models Initial Version." vol. II, N. I. o. S. a. Technology, Ed., 2007.

[3] N. J. Nilsson, "A Mobile Automaton: An Application of Artificial Intelligence Techniques," in *First International Joint Conference on Artificial Intelligence*, Washington DC, 1969, pp. 509-520.

[4] C. e. a. Crane, "Team CIMAR's NaviGATOR: An Unmanned Ground Vehicle for Application to the 2005 DARPA Grand Challenge," *Journal of Field Robotics,* vol. 23, pp. 599-623, 2005.

[5] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *IEEE Robotics and Automation Magazine,* vol. 13, pp. 99-108, 2006.

[6] J. e. a. Albus, "4D/RCS Version 2.0: A Reference Model Architecture for Unmanned Vehicle Systems," N. I. o. S. a. Technology, Ed., 2002.

[7] T. Galluzzo, "Simultaneous Planning and Control for Autonomous Ground Vehicles," in *Mechanical and Aerospace Engineering*. vol. PhD Gainesville: University of Florida, 2006.

[8] J. D. Hightower, "Developement of Remote Presence Technology for Teleoperator Systems," in *14th Meeting of teh UJNR/MFP*, 1986.

[9] D. W. Gage, "Navigating Multiple Robots in Semi-Structured Environments," in *ANS 6th Topical Meeting on Robotics and Remote Systems*, Monterey CA, 1995, pp. 838-845.

[10] B. Shoop, M. Johnston, R. Goehring, J. Moneyhun, and B. Skibba, "Mobile Detection Assessment and Response Systems (MDARS) a force protection, physical security operational success," Kissimmee, FL, United states, 2006, p. SPIE.

[11] DARPA, "DARPA Urban Challenge Technical Evaluation Criteria." vol. 2009, 2006.

[12] J. N. D. Gupta, "An excursion in scheduling theory: An overview of scheduling research in the twentieth century," *Production Planning and Control,* vol. 13, pp. 105-116, 2002.

[13] Re2, "Robotics Engineering Experience." vol. 2009, 2009.

[14] ASI, "Autonomous Solutions Products and Services." vol. 2009, 2009.

[15] OpenJAUS, "OpenJAUS 3.3.0 The Next Generation of OpenJAUS Code." vol. 2009, 2007.

[16] JAUS, "JAUS Reference Architecture." vol. 2007, 2007.

[17] SAE, "JAUS Service Interface Definition Language," Society of Automotive Engineers, Aerospace Standard 2008.

[18] SAE, "JAUS Core Service Set," Society of Autmotive Engineers, Aerospace Standard 2008.

[19] SAE, "JAUS Manipulator Service Set," Gainesville: SAE International, 2009, p. 47.

[20] NATO, "Standard Intefaces of UAV Control System for NATO UAV Interoperability," NATO Standardisation Agency, NATO Standardisation Agreement March 2005 2005.

[21] t. Congress, "H.R. 1815 Section 141," U. S. Congress, Ed., 2005.

[22] J. T. Platts, M. L. Cummings, and R. J. Kerr, "Applicability of STANAG 4586 to future unmanned aerial vehicles," *2007 AIAA InfoTech at Aerospace Conference*, Rohnert Park, CA, United states, 2007, pp. 391-404.

[23] J. T. Platts, "Autonomy in unmanned air vehicles," *Aeronautical Journal,* vol. 110, pp. 97-105, 2006.

[24] J. Albus and A. Barbera, "Intelligent control and tactical behavior development: A long term NIST partnership with the army," Salt Lake City, UT, United states, 2006, pp. 397-404.
[25] R. Touchton, "An Adaptive Planning Framework for Situation Assessment and Decision-Making of an Autonomous Ground Vehicle," in *Mechanical and Aerospace Engineering*. vol. Ph.D Gainesville: University of Florida, 2006, p. 164.
[26] C. Crane, D. Armstrong, A. Arroyo, A. Baker, D. Dankel, G. Garcia, N. Johnson, J. Lee, S. Ridgeway, E. Schwartz, E. Thorn, S. Velat, J. H. Yoon, and J. R. Washburn, "Team gator nation's autonomous vehicle development for the 2007 DARPA Urban challenge," *Journal of Aerospace Computing, Information and Communication,* vol. 4, pp. 1059-1085, 2007.
[27] I. K. Foster, Carl, *The Grid: Blueprint for a New Computing Infrastructure*. San Francisco: Morgan Jaufmann, 1999.
[28] I. K. Foster, Carl, "The Anatomy of the Grid: Enabling Scalable Virtual Organizations," *International Journal of High-Performance Distributed Computing,* vol. 15, pp. 200-222, 2002.
[29] I. Foster, "The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration," Global Grid Forum, 2002, p. 31.
[30] K. Czajkowski, "Resource Co-Allocation in Computational Grids," in *Eighth IEEE International Symposium on High Performance Distributed Computing*, 1999, pp. 219-228.
[31] I. K. Foster, Carl, "A Distributed Resource Management Architecture that Supports Advanced Reservations and Co-Allocation," in *Workshop on Quality of Service*, 1999.
[32] S. Andreozzi, T. Ferrari, E. Ronchieri, and S. Monforte, "Agreement-based workload and resource management," Melbourne, Australia, 2005, pp. 181-188.
[33] A. e. a. Andrieux, "Web Services Agreement Specification (WS-Agreement)," Global Grid Forum June 2005 2005.
[34] K. Czajkowski, I. Foster, and C. Kesselman, "Agreement-based resource management," *Proceedings of the IEEE,* vol. 93, pp. 631-643, 2005.
[35] T. Hao, "A new resource management and scheduling model in Grid computing based on a hybrid genetic algorithm," Guangzhou, China, 2008, pp. 113-117.
[36] D. M. Batista, N. L. S. da Fonseca, F. K. Miyazawa, and F. Granelli, "Self-adjustment of resource allocation for grid applications," *Computer Networks,* vol. 52, pp. 1762-1781, 2008.
[37] S. A. Forum, "The Service Availability Forum and Open Specification Solutions," Service Availability Forum, 2009, p. 13.
[38] R. S. Chin and S. T. Chanson, "Distributed object-based programming systems," *ACM Computing Surveys,* vol. 23, pp. 90-124, 1991.
[39] G. P. Pannirselvam, L. A. Ferguson, R. C. Ash, and S. P. Siferd, "Operations management research: An update for the 1990s," *Journal of Operations Management,* vol. 18, pp. 95-112, 1999.
[40] K. M. M. Chandy, J., "The Drinking Philosophers Problem," in *ACM Transaction on Programming Languages and Systems*. vol. 6, 1984, pp. 632-645.
[41] C. e. a. Crane, "Team Gator Nation's Autonomous Vehicle Development for the 2007 DARPA Urban Challenge," *Journal of Aerospace Computing, Information, and Communication,* vol. 4, pp. 1059-1085, December 2007 2007.

[42] C. a. S. Eden, JC, *Managerial and Organizational Cognition, THeory, Methiods and Research*. London: Sage Publications, 1998.
[43] R. a. P. Bettis, CK, "The dominant logic: Retrospecitve and extension," *Strategic Management Journal*, pp. 5-14, 1995.
[44] R. K. Mobley, *Plant Engineer's Handbook*: Elsevier Butterworth-Heinemann, 2001.
[45] J. N. D. Gupta, "Management Science Implementation: Experiences of a Practicing O.R. Manager," *Interfaces*, vol. 7, pp. 84-90, 1977.
[46] S. Hildebrandt, "CHANGING ROLE OF ANALYSTS IN EFFECTIVE IMPLEMENTATION OF OPERATIONS RESEARCH AND MANAGEMENT SCIENCE," *European Journal of Operational Research*, vol. 5, pp. 359-365, 1980.
[47] Y. Xia and J. Wei, "Automated resource management framework for adapting business service capability," Piscataway, NJ 08855-1331, United States, 2007, pp. 3-9.
[48] K. Ecker, "A framework for decision support systems for scheduling problems," *European Journal of Operational Research*, vol. 101, p. 10, 1997.
[49] M. R. Garey, and Johnson, D.S., *Computers and Intractability: A Guide to teh Theory of NP- completeness*. San Francisco: Freeman, 1979.
[50] A. M. Kelemenis and D. T. Askounis, "A coherent framework for the development of a human resource decision support system," Bangkok, Thailand, 2008, pp. 228-233.
[51] C. K. Sun, V. Uraikul, C. W. Chan, and P. Tontiwachwuthikul, "Integrated expert system/operations research approach for the optimization of natural gas pipeline operations," *Engineering Applications of Artificial Intelligence*, vol. 13, pp. 465-475, 2000.
[52] N. J. Nilsson, *Artificial Intelligence: A New Synthesis*: Morgan Kaufmann Publishers, 1998.
[53] R. A. Brooks, "ROBUST LAYERED CONTROL SYSTEM FOR A MOBILE ROBOT," *IEEE journal of robotics and automation*, vol. RA-2, pp. 14-23, 1986.
[54] R. C. Arkin, "Motor schema-based mobile robot navigation," *International Journal of Robotics Research*, vol. 8, pp. 92-112, 1989.
[55] P. Bonasso and M. Freed, "Layered cognitive architectures: Where cognitive science meets robotics," Arlington, VA, United states, 2004, pp. 82-84.
[56] F. R. Noreils and R. G. Chatila, "Plan execution monitoring and control architecture for mobile robots," *IEEE Transactions on Robotics and Automation*, vol. 11, pp. 255-266, 1995.
[57] D. W. Payton, "ARCHITECTURE FOR REFLEXIVE AUTONOMOUS VEHICLE CONTROL," San Francisco, CA, USA, 1986, pp. 1838-1845.
[58] M. B. Dias, "TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments," in *The Robotics Institute*. vol. Doctor of Philosophy Pittsburgh: Carnegie Mellon University, 2004, p. 139.
[59] R. Zlot and A. Stentz, "Market-based multirobot coordination for complex tasks," *International Journal of Robotics Research*, vol. 25, pp. 73-101, 2006.
[60] A. Saffiotti, "Fuzzy logic in autonomous robotics: Behavior coordination," Barcelona, Spain, 1997, pp. 573-578.
[61] R. Liscano, A. Manz, E. R. Stuck, R. E. Fayek, and J.-Y. Tigli, "Using a blackboard to integrate multiple activities and achieve strategic reasoning for mobile-robot navigation," *IEEE expert*, vol. 10, pp. 24-36, 1995.

[62] D. Kent, "Storing and Predicting Dynamic Attributes in a World Model Knowledge Store," in *Mechanical and Aerospace Engineering*. vol. Doctor of Philosophy Gainesvill: University of Florida, 2007, p. 190.
[63] S. Aoki, J. Nakazawa, and H. Tokuda, "Spinning sensors: A middleware for robotic sensor nodes with spatiotemporal models," Kaohsiung, Taiwan, 2008, pp. 89-98.
[64] B. Benneweis, "ISOBUS, An Introduction to ISO 11783," 2008.
[65] N. J. Nilsson, "Shakey the Robot," *SRI International*, 1984.
[66] J. S. Albus, "Outline for a theory of intelligence," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, pp. 473-509, 1991.
[67] F. Alhalabi, B. N. R. Aubry, M. Maranzana, L. Morel, and L. Jean, "Centralized vs. decentralized QoS management policy," Damascus, Syria, 2008.
[68] B. Merritt, "Vision Based Robotic Convoy," in *Department of Mechanical and Aerospace Engineering*. vol. MS Gainesville: University of Florida, 2009.
[69] R. R. C. Competition, "Introduction to the Robotic Range Clearance Competition." vol. 2010, 2009.

BIOGRAPHICAL SKETCH

Gregory Garcia was born and raised in Saint Petersburg, Florida. He graduated from the Saint Petersburg High School International Baccalaureate Program in 2001. Mr. Garcia received his Bachelor of Science degree in Mechanical Engineering from the University of Florida in Gainesville, Florida in April, 2005. He received a Master of Science degree in Mechanical engineering from the University of Florida in August, 2008. He is currently pursuing his Doctorate degree and serving as a Graduate Research Assistant at the Center for Intelligent Machines and Robotics (CIMAR) in the Mechanical and Aerospace Engineering Department of the University of Florida. Mr. Garcia was an integral part of Team Gator Nation and Team CIMAR, whose autonomous ground vehicles, Urban NaviGATOR and NaviGATOR, competed in the 2007 DARPA Urban Challenge and 2005 DARPA Grand Challenge, respectively. Mr. Garcia participated in the Joint Architecture for Unmanned Systems (JAUS) Working Group, a DoD-sponsored body promoting interoperability of robotic systems and components. He is currently an active committee member of the Society of Automotive Engineers (SAE) International AS-4 Unmanned Systems Standards Body, an international group aimed at developing standards documents for all unmanned systems operating domains. His current research focuses on intelligent decision-making for autonomous ground vehicles.

A COGNITIVE RESOURCE MANAGEMENT FRAMEWORK FOR AUTONOMOUS GROUND VEHICLE SENSING

Gregory A. Garcia
(352) 262-6925, GregAGarcia@gmail.com
Department of Mechanical and Aerospace Engineering
Dr. Carl D. Crane, III
Doctor of Philosophy, Mechanical Engineering
May 2010

Autonomous ground vehicles are entrusted with performing high complex tasks that necessitate the concurrent coordination of multiple system resources; increased capabilities management is needed to meet these requirements in an effective and efficient manner. This dissertation presents a novel framework which formalizes the discovery, modeling, monitoring, and configuration of a distributed collection of disparate resources onboard an autonomous ground vehicle. This framework was implemented and tested on a large, JAUS compliant unmanned ground vehicle. It supports the management of articulated camera, laser, and radar sensors but is extensible to other sensor and platform types and system architectures. This research provides the foundation upon which new sensing, planning, and scheduling methodologies pertaining to intelligent resource utilization can be implemented and evaluated with the hope of promoting the advancement of highly autonomous platforms capable of sophisticated interactions with unknown, dynamic environments.