NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

SOLVING DYNAMIC BATTLESPACE MOVEMENT PROBLEMS USING DYNAMIC DISTRIBUTED COMPUTER NETWORKS

by

Robert D. Bradford, III

June 2000

Thesis Advisor:	Gordon H. Bradley
Second Reader:	Arnold H. Buss

Approved for public release; distribution is unlimited.

DTIC QUALITY INSPECTED 4

20000818 065

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

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

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 2000	Master's Thesis

4. TITLE AND SUBTITLE
SOLVING DYNAMIC BATTLESPACE MOVEMENT PROBLEMS USING DYNAMIC DISTRIBUTED COMPUTER NETWORKS

5. FUNDING NUMBERS

6. AUTHOR(S)
Bradford, Robert D., III

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Office of Scientific Research, 801 North Randolph Street, Arlington, VA 22203-1977

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

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

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

This thesis develops an architecture for dynamic distributed military operations research. This architecture assumes that a network of heterogeneous computing devices connects forces throughout the battlespace. Both the raw data about the battlespace and the operations research models used to analyze this data are accessible to devices on this network. The thesis designs a system using this architecture that invokes operations research network optimization algorithms to solve problems involving movement of people and equipment over dynamic road networks. A specific application is implemented to help a medic find the nearest aid station using a shortest path algorithm. This application marshals the most current data on unit locations and road conditions (distributed across the computing network) and locates on the network an appropriate algorithm that is then used to construct a solution. The answer is returned to the user as a web page in a form appropriate for his computing device. The application is implemented with existing technologies including the Java computer language, König, a Java-based tool for representing networks and graphs, and Hypertext Markup Language, a format for shared information on the Internet. This system uses operations research tools to transform data into decisions in real-time or near real-time.

14. SUBJECT TERMS
Computer Architecture, Network Optimization, Java, Loosely Coupled Components

15. NUMBER OF PAGES
72

16. PRICE CODE

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

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239.18

Approved for public release; distribution is unlimited.

SOLVING DYNAMIC BATTLESPACE MOVEMENT PROBLEMS USING DYNAMIC DISTRIBUTED COMPUTER NETWORKS

Robert D. Bradford, III
Captain, United States Army
B.S.E., Princeton University, 1989

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
June 2000

Author: _____
Robert D. Bradford, III

Approved by: _____
Gordon H. Bradley, Thesis Advisor

_____
Arnold H. Buss, Second Reader

_____
Richard E. Rosenthal, Chairman
Department of Operations Research

ABSTRACT

This thesis develops an architecture for dynamic distributed military operations research. This architecture assumes that a network of heterogeneous computing devices connects forces throughout the battlespace. Both the raw data about the battlespace and the operations research models used to analyze this data are accessible to devices on this network. The thesis designs a system using this architecture that invokes operations research network optimization algorithms to solve problems involving movement of people and equipment over dynamic road networks. A specific application is implemented to help a medic find the nearest aid station using a shortest path algorithm. This application marshals the most current data on unit locations and road conditions (distributed across the computing network) and locates on the network an appropriate algorithm that is then used to construct a solution. The answer is returned to the user as a web page in a form appropriate for his computing device. The application is implemented with existing technologies including the Java computer language, König, a Java-based tool for representing networks and graphs, and Hypertext Markup Language, a format for shared information on the Internet. This system uses operations research tools to transform data into decisions in real-time or near real-time.

DISCLAIMER

The reader is cautioned that computer programs developed in this research may not have been exercised for all cases of interest. While every effort has been made, within the time available, to ensure that the programs are free of computational and logic errors, they cannot be considered validated. Any application of these programs without additional verification is at the risk of the user.

TABLE OF CONTENTS

I. BACKGROUND 1

A. Joint Vision 2010 1

B. Empowering Decision Makers 2

C. Putting Operations Research into this Environment 3

D. Road Movement Problems 4

II. ARCHITECTURE 7

A. The Underlying Communications Network 8

B. Client-Server Design 11

C. Model-View-Controller 14

D. Mobile Data and Algorithms 16

E. The ADDMOR Design 17

III. TECHNOLOGIES 19

A. Transport Control Protocol/Internet Protocol 19

B. Hypertext Markup Language 19

C. HTTP Server 20

D. Java 21

E. Servlets 23

F. König 24

IV. MILITARY ROAD MOVEMENT OPTIMIZER 27

A. Distributed Design 27

B. The Medic 29

C. The Expert User 33

V. OTHER PROBLEMS 39

VI. CONCLUSIONS 43

LIST OF REFERENCES 45

INITIAL DISTRIBUTION LIST 47

LIST OF FIGURES

Figure 1. A schematic of a computer network. 9
Figure 2. Schematics of thick and thin client computer architectures. 13
Figure 3. Schematic of Model-View-Controller design 15
Figure 4. A distributed computing MRMO implementation 28
Figure 5. The medic's introductory screen 30
Figure 6. The introductory screen viewed on a Palm Pilot. 31
Figure 7. The medic's solution screen 32
Figure 8. Solution screen viewed on a Palm Pilot 33
Figure 9. Expert user introduction page 34
Figure 10. The expert user's next view 35
Figure 11. The expert user's solution page 36
Figure 12. A graphical view of the solution. 37

EXECUTIVE SUMMARY

Military leaders have predicted a future battlefield that is much different than that of today. Joint Vision 2010 and the Army Vision 2010 create a framework for planning a future force to succeed on that battlefield. Information superiority is the most important tenet of Joint Vision 2010 and it can lead to dominant battlespace awareness, providing leaders a much more accurate assessment of friendly and enemy operations. Enabling technologies, including tactical computers, communications networks, and analytical tools that synthesize data from many sources on the battlefield, are key to taking advantage of information superiority.

According to Joint Vision 2010, soldiers on the battlefield will be connected on a network of digital computing devices capable of communicating with one another. With the explosion of sensors and digital devices on the battlefield, huge amounts of data will be collected and will be available to soldiers and decision makers in the battlespace. However, access to these data is not sufficient by itself to help these soldiers reach decisions. Raw data without any analysis might even hinder decisions, causing the decision maker to get lost in minutia and slowing reaction time. Raw data can even confuse and overwhelm, leading to incorrect conclusions. Trends and insights in the data are often not obvious and require sophisticated tools to uncover.

Operations research tools can help in this data-rich environment by supporting better, faster decisions in real-time and near real-time. The field of operations research specializes in using models of the real world to solve problems that provide information

and insight to help the decision maker. It has many tools to help decision makers refine raw data into decisions.

To be useful in real-time or near real-time, an operations research model or algorithm must execute on existing and future computing devices available in the battlespace, and the results of these models must be effectively presented to the decision maker to help him make decisions. Currently, most operations research models are not designed to operate in this digitally connected environment. Most are designed as stand-alone models that operate in specific locations coincident with the data. These operations research software tools, models, and algorithms must be redesigned to operate in a future environment that consists of dynamic, distributed computing devices, large numbers of sensors, huge collections of raw data, and many decision makers working at all levels of the battlespace. If the models do all of these things they will have a significant value on the battlefield of the future.

A specific set of problems that current operations research tools can help solve involves efficiently moving men and equipment around a road network. The dynamic nature of the road network presents a unique problem. Road trafficability can change with the weather and with changes in the tactical situation. Roads can be controlled by the enemy at one time and then controlled by friendly forces at another. In addition unit locations are very dynamic since units move frequently. In general unit locations change more frequently than the status of the road network. Another complicating factor is the dynamic nature of the underlying communications grid on which data flows between units. This communications grid is made up of the digital computing devices and the

wires and/or radios that connect them. Units enter and leave the communications grid with regularity. In fact, the communications grid may not even remain connected. An additional challenge is the distributed nature of the data needed to support the models and algorithms. Different computers in distinct locations can possess data about the road network; collecting and synthesizing this data into a single picture of the battlespace presents a formidable task.

This thesis develops an architecture for dynamic distributed military operations research systems that can operate on the future battlefield. Existing network and software technologies are used to construct a specific system based on this architecture to optimize the movement of people and equipment over a battlespace road network. This system is demonstrated by constructing an application that answers the battlefield question, "Where is the nearest aid station?" The dynamic nature of the road network, unit locations, and the communications network makes this question more difficult than it first appears.

The road movement system demonstrates the power of the architecture. It takes a dynamic road network and a dynamic communications network with distributed data and distributed computing power and combines them with an operations research model to solve movement problems; it processes data and extracts the information of value to decision makers in real-time and near real-time. Data is converted into better, faster decisions. This system demonstrates that existing widely used network and software technology can be used to construct battlespace decision support systems of the type envisioned in Joint Vision 2010 and Army Vision 2010.

ACKNOWLEDGMENT

The author would like to acknowledge some people who were instrumental in the completion of this thesis. Discussions with Major David Krizov, USMC, led to the initial idea for this thesis, and were important in deciding to bring operations research to the people with "muddy boots". Captain Max Moore, USA, did some initial work with servlets for a distinctly different project, and I was able to learn from much of his work. Dr. Buss has a unique way of designing software, and some of his ideas made it into my implementation. Dr. Bradley was very patient with me as I tried to develop an interesting idea into a thesis. He gave me great guidance and was very supportive of my idea. And finally, my wife Melanie has done the most to help me complete this thesis. Without her love, support, and encouragement, I would be lost.

I. BACKGROUND

A. JOINT VISION 2010

Current military leaders are predicting a future battlefield that is much different than that of today. [1,2,3] Joint Vision 2010 and the Army Vision 2010 create a framework for planning a future force to succeed on that battlefield. The five tenets of Joint Vision 2010 are: Information Superiority, Dominant Maneuver, Precision Engagement, Full Dimensional Protection, and Focused Logistics. Of these, Information Superiority is the enabling technology for the other four tenets. Information Superiority will be gained through developing enabling technologies including tactical computers, communications networks, and analytical tools to synthesize information from many sources on the battlefield.

Joint Vision 2010 states "Advances in computer processing, precise global positioning, and telecommunications will provide the capability to determine accurate locations of friendly and enemy forces, as well as to collect, process, and distribute relevant data to thousands of locations. Forces harnessing the capabilities potentially available from this system of systems will gain **dominant battlespace awareness,** an interactive 'picture' which will yield much more accurate assessments of friendly and enemy operations." [1, p.13] The Army must harness the potential capabilities of these systems if it is to achieve these goals.

As the Army pursues its part of this vision, it is concentrating on developing the necessary enabling technologies in order to make the future battlefield different than that

of today. With the improvements in digital technology, computers will be much more numerous, more powerful, and much smaller than today. Current Army experiments include putting a computer on every vehicle and even on every individual soldier. These computing devices will be connected on a digital communications network that will allow information to move quickly around the battlespace. [4] Future technologies will solve the current limitations on bandwidth. The complex network of digital devices will create a huge amount of data. The difficult goal of achieving information superiority will not be met by just collecting these data. It will be achieved by taking this glut of data, more than can be processed and understood by humans in finite amounts of time, and processing it into information that can aid the decision makers.

B. EMPOWERING DECISION MAKERS

Decision makers have traditionally been thought of as people high in the organizational hierarchy whose decisions have great influence on the battlespace. In the future vision of the battlespace, this view of decision makers must be expanded to include all people who make decisions down to the lowest levels. Access to the great amounts of data should benefit not just the high ranking decision makers at the top of a hierarchical structure whose decisions have obvious far reaching effect, but also the soldiers working at the lowest level, whose decisions must be made quickly and in accordance with the overall intent of the commander. The idea of a self-synchronizing system requires that people at the lower levels have much autonomy during the execution phase of an operation, and although their decisions may not have as great or immediate an impact as

decisions made by higher ranking officers, they are based on the same data sources as the decisions made at the highest level.

A means of collecting and transmitting raw data is required in order to secure this vision and truly help decision makers. With the explosion of sensors and digital devices on the battlefield, huge amounts of data will be collected, and will be available to decision makers in the battlespace. However, access to these data is not sufficient by itself to help people reach decisions. Raw data without any analysis might even hinder decisions, causing the decision maker to get lost in minutia and slowing reaction time. Raw data can even confuse and overwhelm, leading to incorrect conclusions. Trends and insights in the data are often not obvious and require sophisticated tools to uncover. Operations research models and algorithms can help get important information out of the data. Often optimal decisions are not obvious by simply inspecting the data. Operations research models and algorithms executing on modern computers can assist in solving these difficult problems. They can provide significant value to decision makers who are pressed for time and operating in high stress environments.

C. PUTTING OPERATIONS RESEARCH INTO THIS ENVIRONMENT

Operations research tools can help in this data-rich environment by supporting better, faster decisions in real-time and near real-time. The field of operations research specializes in using models of the real world to solve problems that provide information and insight to help the decision maker. It has many tools to help decision makers refine raw data into decisions. Some examples of operations research tools include optimization models, simulation models, and network algorithms.

To be useful in real-time or near real-time, an operations research model or algorithm must execute on existing and future computing devices available in the battlespace, and the results of these models must be effectively presented to the decision maker to help him make decisions. Currently, operations research tools exist that help decision makers reach good informed decisions. However, most of these tools have been designed to run on a single computer that contains all the necessary data and algorithms to process the data. The software is often limited to a specific computer system that has a specific hardware and software configuration. Many models also output solutions in formats that are very technical, making it difficult to show decision makers model output without first processing it into an understandable format. These operations research software tools, models, and algorithms must be redesigned to operate in a future environment that consists of dynamic, distributed computing devices, large numbers of sensors, huge collections of raw data, and many decision makers working at all levels of the battlespace. If the models can do all of these things they will have a significant value on the battlefield of the future.

D. ROAD MOVEMENT PROBLEMS

A specific set of problems that current operations research tools can help solve involves efficiently moving men and equipment around a road network. Many models currently exist to help solve these problems in a static environment, but most of these models are not designed to operate in a dynamic environment. The dynamic nature of the road network presents a unique problem. Road trafficability can change with the weather and with changes in the tactical situation. Roads can be controlled by the enemy at one

time and then controlled by friendly forces at another. In addition unit locations are very dynamic since units move frequently. Unit locations in general change more frequently than the status of the road network. Another complicating factor is the dynamic nature of the underlying communications grid on which data flows between units. This communications grid is made up of the digital computing devices and the wires and/or radios that connect them. Units enter and leave the communications grid with regularity. In fact, the communications grid may not even remain connected. An additional challenge is the distributed nature of the data needed to support the models and algorithms. Different computers in distinct locations can possess data about the road network; collecting and synthesizing this data into a single picture of the battlespace presents a formidable task.

A difficult question to answer on the current battlefield is "Where is the nearest medical aid station?" This question can mean the difference between life and death when evacuating a casualty. With units moving and the status of the road network changing this problem is actually much more difficult than it at first appears. Because of the dynamic nature of the location of units, the road network, and the communications network, a real-time solution to this question is very important. A system that has real-time access to data about the roads and military units can provide the best possible answer to this question.

In such an environment, with a dynamic road network and a dynamic communications network with distributed data and distributed computing power, the challenge is to develop a network and software architecture for models and algorithms to

operate in real-time and near real-time to help decision makers convert data into decisions.

This thesis develops the requirements for this architecture in Chapter II. Chapter III describes some current technologies to be used in systems designed for this architecture. Chapter IV discusses in detail the design and development of a specific system to optimize the movement of people and equipment around a road network. Chapter V lists some other problems that could be solved using this architecture and Chapter VI provides conclusions for this thesis.

II. ARCHITECTURE

The future battlespace will require many systems that will use the vast amount of available data to support decision making by a variety of users. Operations research models and algorithms can be used to convert raw data into information that can be used by these users. Even though these systems will differ in the data, models, and algorithms they use and in the users they serve, they can all be based on a single network and software architecture that is developed here. Developing a variety of systems based on a single architecture avoids the development of systems with different designs that are incompatible.

The Architecture for Dynamic, Distributed Military Operations Research (ADDMOR) developed here is needed to help Army decision makers on the future battlefield. Systems based on ADDMOR synthesize the huge amounts of data that are collected by sensors on the network and refine it through the use of models and algorithms into a format that decision makers can use. A specific problem where a system based on ADDMOR can add great value is the efficient movement of men and equipment around the battlefield. This problem exists through the full spectrum of Army operations, from stability and support operations through major theater war. A system that can function at all of these levels can add great capabilities to decision makers and will have far reaching impact.

A. THE UNDERLYING COMMUNICATIONS NETWORK

In the Army's vision of the future, soldiers throughout the battlespace will operate digital computer devices connected to a communications network. These soldiers will perform many different functions and so will have many different hardware and software configurations on their local machines. These digital devices will communicate with each other on a network whose transfer medium may include High Frequency and FM radios, tactical satellite communications, and wire and fiber optic cable. They will all be able to communicate with each other because they will all use a low-level communications protocol such as Transport Control Protocol/Internet Protocol (TCP/IP) used by the Internet.

Figure 1 shows a diagram of a computer network. Connected computers will exchange information using the underlying communications network. This computer network will not require centralized control. It is not necessary that all communications be routed through a central computer that controls the network. The computer network does not need a homogeneous computing environment. Many different computational devices will be accessible on this computer network. They will range in power and size from personal digital assistants and cellular telephones through personal computers to large mainframes and super computers. These devices will have access to great amounts of data about the battlespace. These data could be distributed throughout the network, or they may be collected into a large data store that exists on one or more physical computers.

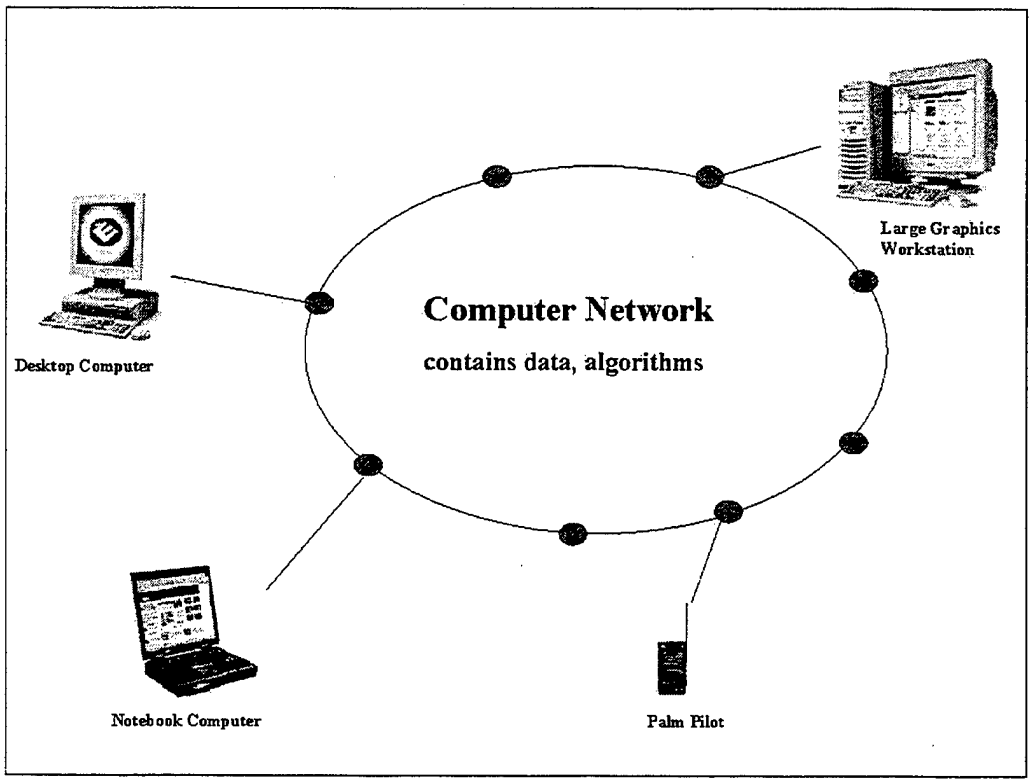

Figure 1. A schematic of a computer network.

In order to solve the specific class of problems dealing with road movement, a system must have access to certain data about the physical environment and access to operations research algorithms to work on this data. In the future battlespace, data about the road network will be maintained in a data store that is connected to the computer network. These data will include road connectivity, road trafficability, travel times on the roads for wheeled and tracked vehicles, and capacity of roads in vehicles per hour. The collection and storage of the raw data will be done routinely. Access to this data is critical to using operations research to refine the raw data into decisions. Existing and future database technology is making great advancements in the areas of real-time data collection and access, and the military will benefit from much of this technology.

Data about military units will also be maintained in a data store. These data will include unit identification, unit location, unit size, and unit type. This data store may be physically located somewhere distinctly different from the data about the road network. It will also be accessible on the computer network.

In addition to data, algorithms will be accessible on the computer network. In this class of movement problems, these algorithms can operate on graphs representing the roads and units in order to optimize movement on the road network. Examples of different graph algorithms that can help decision makers change data into decisions about road movement include shortest-path, maximum flow, minimum cost flow, and others.

In the future environment, people connected to the computer network will need to solve problems and make decisions. These people will have varying levels of knowledge of operations research and its tools. Some users will be relatively naive, while others will be operations research analysts with a great knowledge of models and algorithms. The difference between the naive and expert user is their knowledge of operations research models and algorithms. For example, a naive user may be a medic in an ambulance who only needs to know the shortest path to a medical unit. The information he needs is a destination and how long it takes to get to that destination. An expert user may be an analyst at a Tactical Operations Center who is trying to plan the next position for a medical unit to move to. He may want to solve an algorithm to minimize the longest shortest path to any maneuver unit, and position his unit at that location.

Different types of users is will use different computational devices to connect to the communications network. Current network devices include mainframe and personal

computers, hand-held digital devices and personal digital assistants, and even cellular telephones and pagers. ADDMOR systems should allow for a full range of computational devices, and be expandable to devices that are not yet even imagined. Future military operations will emphasize joint and coalition operations with forces that may have many different levels of organic technology, and many different hardware and software configurations. In this environment different users will have vastly different hardware running different systems software, and it will be unwise to mandate anything but the lowest level of software standards. Users should be able to operate in this environment as long as the user communicates on the network using an agreed upon standard low-level communications protocol. ADDMOR systems should allow any kind of hardware configuration and must operate on any and all conceivable platforms in order to be useful. These systems should also be able to scale up to larger networks, and they should be capable of evolving with technology as devices on the network move to different hardware and software configurations.

Another challenge of working in the future battlespace will be the dynamic nature of the communications network. Computational devices will enter and leave the network frequently. Connections will change rapidly as units move within their zone of coverage. Message routes will change as units move physically and as different parts of the network become congested. All of these characteristics make robustness a design imperative.

B. CLIENT-SERVER DESIGN

Computer network architecture has moved through a few design paradigms in recent years that affect the operation of the network. Traditional mainframe computer

networks operated in a client/server mode with all of the computer processing taking place on the mainframe, and the client, (the user's device), acting simply as a dumb terminal. This paradigm is not sufficient for work on a busy connected network, since the mainframe or server can be overwhelmed when executing simultaneous processes for multiple users. It also does not use the capabilities available with the more powerful client or user devices. With the growth of the Internet and the power of connected computing devices, systems have moved toward a thick client network architecture. In this case the network server acts as a file server, sending applications to the client computing devices where the programs are executed. In a thick client design most of the processing takes place on the client, and the server acts primarily as a file and data storage device. This network architecture places significant demands on data transfer, and suffers significant degradation when bandwidth between computers is limited and multiple users are trying to move large applications and data files to their own computers at the same time. [5]

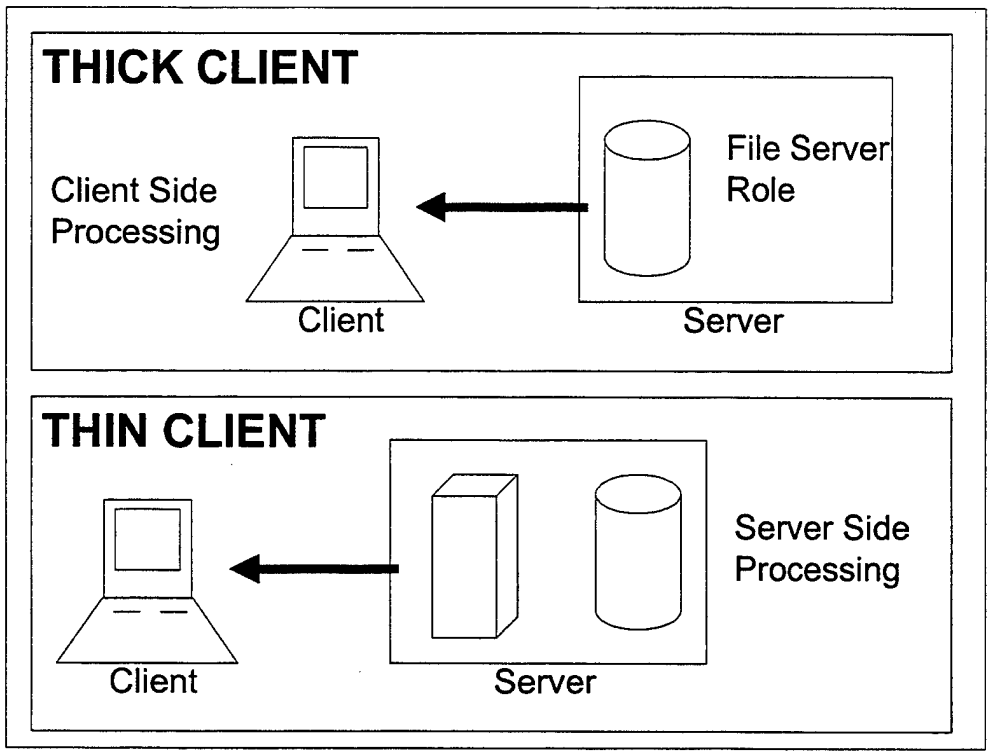

Figure 2. Schematics of thick and thin client computer architectures. After Reference [5]

The current trend is toward a thin client network architecture. This design framework is between the client/server architecture and the thick client architecture. In this case most but not all of the processing is done at the server; some processing is still executed at the client. One key advantage over the thick client architecture is that less data must be transferred across the network. This is an important benefit because current network limitations on bandwidth can slow the transfer of large applications across the network. Another advantage is that thin client design does not depend on powerful client side computers. This design places only minimal requirements on client side computing

power and memory. This thin client design should operate with many levels of future computing devices.

In the future, connected environment units will sometimes operate autonomously, without the benefit of a network. For a system to be useful it should be able to operate in a stand-alone mode. In this mode, functions should appear to the user to operate the same as if the system were connected to a network. Some functionality may be limited due to lack of access to different algorithms and data, but the look and feel of the system should not change. The system should be able to optimize its operations to use different computational assets as these different assets are connected to the communications network. This could give the system different levels of functionality depending on what computer devices are connected to the communications network.

C. MODEL-VIEW-CONTROLLER

An important design concept for systems is that the view of the world that the user gets is separated from the model used to represent the world in the computer. A controller between the model and the view of the user takes inputs from the user and forms them into a problem the model can solve. It then takes the solution from the model and converts it into a form the user can see and understand. The model does not need to know who the user is, his level of technical expertise, or the capabilities of his computational device. Similarly, the user does not need to know how the underlying model works. Whenever new algorithms are implemented, the controller can give those capabilities to the user without the user having to know how they are implemented. This separation of the user from the model makes it simple to change the model without

having to change the view of a user. It also makes it easier to customize a view for different user types without having to understand the underlying model. [6] Figure 3 shows the how the model and view interact through a controller.

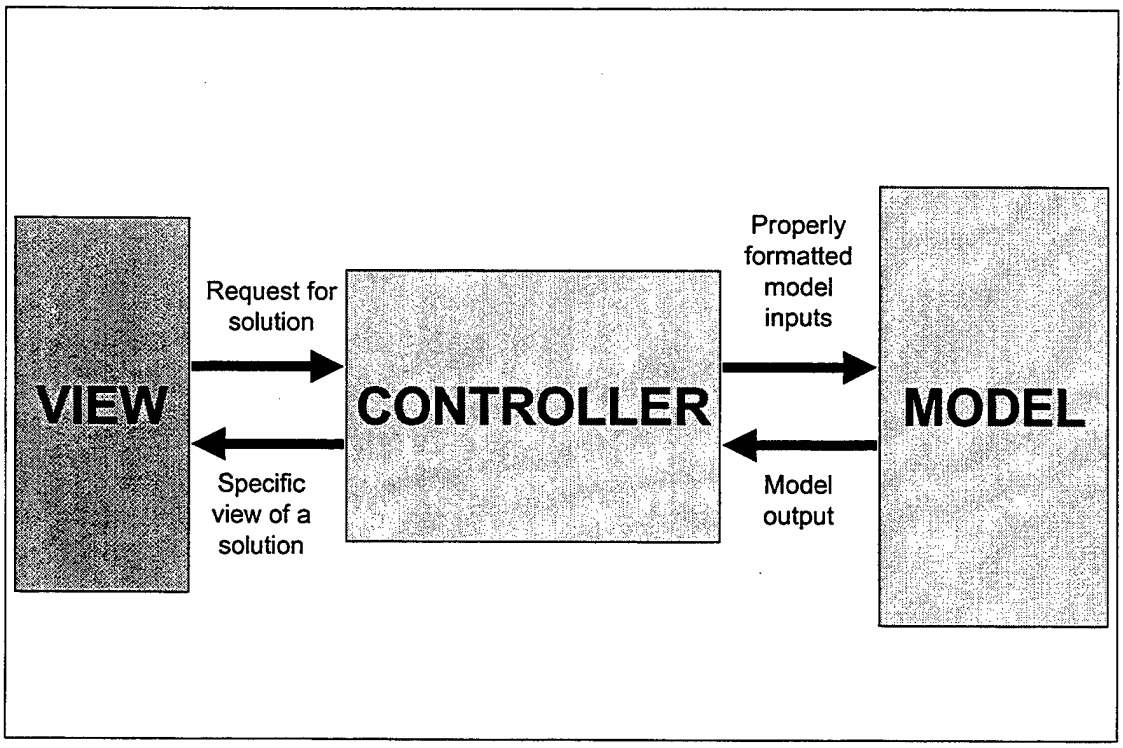

Figure 3. Schematic of Model-View-Controller design

With the separation of the model from the user it is very simple to update the algorithm without affecting the user. If a faster implementation of a specific algorithm is written, it can be plugged into the system without any change in how the system works for the user. The user does not need to know anything about the underlying system and how it works. Similarly, if a system designer thinks of a new user who only needs specific capabilities of the system, he can create a new view that interfaces with the controller. This view can be specific to this new user and may look totally different than

anything previously imagined. The underlying model will not change and the people creating the model need not have any knowledge of this new user and his new view. This capability to use the one model for many different user types each with a different view of the system is valuable. Separate models are not required for different types of users.

D. MOBILE DATA AND ALGORITHMS

One other challenge facing battlefield systems is the mobility of the road networks data, the units data, and the algorithms. These may either be stored on several different devices around the network, or in a centralized database. The user may know where this data is stored; in fact it may be on his machine. However, a system should not require the user to have any information about where this information resides on the computer network. A good design will allow for the data to be located in many places around the network.

Computationally intensive models and algorithms should take advantage of the computational power in the computer network regardless of where it is located. A system should allow the hard work of actually solving the model to be moved to the most appropriate computational device available on the network. For more difficult, time consuming algorithms, the system could send the solution to a super-computer to be solved, while simple problems could be solved on less capable devices. This ability to use existing power in the system to solve problems enables the complexity of problems to increase as computational devices on the computer network become more powerful. The decision about which device to use to solve the problem might be invisible to the user. He does not necessarily need to know about the capabilities of the computer system, and

in many cases will not want to know. He just wants to know that at the instance he submits his problem, the system will marshal the best possible combination of data, algorithms, and computing devices to solve his problem.

E. THE ADDMOR DESIGN

As described above, the ADDMOR design consists of:

1. A communications network of heterogeneous devices based on a low level broadly supported protocol,
2. A thin client network architecture with a model-view-controller design, and
3. Data, models, and algorithms distributed over the communications network that can migrate over the network as it changes.

In the last several years a number of hardware, network, and software technologies have emerged that allow systems based on ADDMOR to be developed. The next chapter describes these technologies and Chapter IV describes the Military Road Movement Optimizer, a decision support system for the movement of people and equipment around the battlefield that is based on the ADDMOR architecture.

THIS PAGE INTENTIONALLY LEFT BLANK

III. TECHNOLOGIES

Design decisions must be made in order to build a specific system implementing the ADDMOR that meets all of the preceding requirements.

A. TRANSPORT CONTROL PROTOCOL/INTERNET PROTOCOL

In order to implement a useful, scalable and evolvable system that can help transform the large amounts of data on the network into decisions on the battlefield, users must be able to communicate with one another. The system requires a low level communications protocol, so that users can communicate with each other and with the data and algorithms on the network. Currently most digital networks use TCP/IP, a protocol developed to route messages effectively around the Internet and fully functioning on a wide variety of communications networks and computer devices. The military is developing a Tactical Internet that will also use TCP/IP to send information around the network [7, 8].

B. HYPERTEXT MARKUP LANGUAGE

Given this communications protocol and a desire to have a thin-client architecture, systems need a common format for transmitting information back and forth between the user and the controller. This basic format should be the same for the entire range of users and applications. The well-known interface for users to interact with is the Internet is a web browser. A browser is capable of reading files in the Hypertext Markup Language (HTML) format. HTML is a language used to encode formatted text so it can be transferred over the Internet, and displayed on a variety of devices. It is a very low-level standard in use all over the Internet and the World Wide Web. Browsers are

available for most current devices including personal computers and personal digital assistants, and should be available for new devices in the future. By making the only requirement of a user that he have a web browser that can process an HTML message, ADDMOR systems allow for users without powerful machines. Users' devices do not need a lot of computing power to display solutions to problems. As network computing moves to smaller and smaller devices like cellular telephones and personal digital assistants, HTML browsers are some of the first applications developed for these devices. A system that takes advantage of this fact and interacts with users using HTML will be able to expand to these future devices with minimal effort. HTML allows for a truly thin-client architecture.

Another advantage of HTML is the small size of files stored in this format. Since the data that does not include graphics is stored basically as text with a few formatting tags, it is not large in size. An HTML file consisting of 3 text characters is 212 bytes in length, while the same file saved in the Microsoft Word document format is 19,456 bytes in length. This is almost two orders of magnitude larger. The size of files is especially important when sending the information around a digital computer network. Smaller files can be sent more quickly and use less communications bandwidth.

C. HTTP SERVER

The thin client architecture using HTML requires that at least one device on the network is operating as a Hypertext Transfer Protocol (HTTP) server. These servers distribute web pages across a digital network. With the growth of the World Wide Web, many HTTP server implementations are widely available. Apache is a quite powerful

HTTP server that was developed and is maintained by the Apache Software Foundation. Apache is a robust commercial grade, open-source implementation of a server that is freely available for download on the Internet. It is also the HTTP Server in widest use today, being used by over 60% of all web servers[9]. The group that has developed Apache believe that a free implementation of an HTTP server is vital to the health and growth of the World Wide Web, so they have pledged to always distribute Apache free of charge. [10]

D. JAVA

The requirement that systems be able to operate on many different computer configurations led to the decision to implement them in Java. Java is a powerful platform independent computer language developed by Sun Microsystems. A program written and compiled in Java can be transported from one computer platform to a totally different computer platform without having to rewrite or recompile any of the underlying code. The only commonality between the two devices is that they both must have a Java Virtual Machine, a specific software implementation that translates Java byte code into machine instructions specific to that hardware device. Platform independence is an enabling property in an environment where the hardware components on the communications network are changing as quickly as the network itself. The use of Java allows the system to be independent of the hardware configuration of the communications network; the program is inherently portable to other machines connected to the network. This is a very important idea, since the future operating environment will include other armed services in joint operations, other countries' militaries in coalition operations, and other

governmental agencies and non-governmental organizations in stability and support operations.

Another advantage of using Java is its capabilities to interface with databases. The JDBC Data Access(tm) API developed by Sun has many powerful tools for accessing, querying, and updating databases from other Java applications. This means that if the data about the roads and units is stored in databases, Java has built in tools to help access that data and use it in different models. Java's JDBC capability allows programs to access data in virtually any existing or future database. It takes advantage of the low level Standard Query Language (SQL) that almost all current database technology implement to allow access to data in different databases.

Java was originally developed as a computer language to harness the power of the Internet. Because of this it has many built in features that facilitate distributed computing operations. It has a very powerful and robust security model that can allow selective access to information and computer resources. Authentication and encryption are both implemented in Java. These tools are especially important in a military environment where access to information must be controlled.

Java also has the ability to dynamically load programs from across a communications network. An executing Java class or program can query another computer on the network to find out what compiled programs it has available. The executing program can then download and execute one of those programs, without stopping the current execution. This is a very powerful idea that greatly extends the

functionality. The first program does not need to know which programs are available ahead of time; it can choose from the list of currently available programs. [11]

E. SERVLETS

Servlets are a tool developed using Java that allows users on a client machine to run applications on the server. A servlet runs completely on the server side of the network rather than on the client computer. A servlet generates HTML for the client, usually a web browser. The user or client does not need to have a Java Virtual Machine on his device, and in fact needs to have no knowledge of the existence of Java on the server. The Java servlet API provides powerful tools for programs so that they can operate on the server and can serve HTML back to the user. The servlet is a fully functioning Java program located on a server that runs when a user points his browser to the web address of the servlet. The servlet has all of the previously mentioned powerful capabilities of Java including platform independence, database connectivity, distributed computing, security, and dynamic loading. Because it is implemented in Java using the servlet API, it does not suffer from many of the shortcomings of other server side applications that use the Common Gateway Interface (CGI) implementations. Also servlets tend to be more efficient than CGI scripts. [5]

In order for a servlet to work, the machine it is operating on must have a web server, such as the HTTP server developed by Apache, and a servlet engine. A popular servlet engine that has been specifically designed to operate with the Apache server is Apache JServ, developed by the Java Apache Project.[12] JServ allows servlets written in Java to operate on the network and to serve web pages to users on the network. It is a

powerful, award-winning software implementation that is widely available on the Internet at no cost.

The HTTP server allows users to connect to the system with a HTML browser, and the servlet engine allows servlets to execute, giving all of the capabilities of Java. In this design, user devices must only be able to process HTML encoded text. Their view of the system is just the HTML page they are presented with. A servlet acts as the controller between the user's view and the operations research models. A user enters information into an HTML form. This information can either be entered by the user, or can be built in and even hidden in the form. The servlet parses this form after it is submitted, and sends the input parameters to a model interface that determines which model to use, and how it should operate. The model executes, and sends results to the model interface that sends the solution to the servlet based on the input parameters. The servlet then send this solution back to the user as a HTML encoded text.

If a user wants to operate autonomously as a stand-alone system not connected to the computer network, he can still use the servlet. He must have an HTTP server, a servlet engine, the specific servlet he wants to use, and a web browser on his machine. The user's view of the servlet is the same as if he were accessing it remotely.

F. KÖNIG

One existing software tool in Java that the proposed system uses to solve network algorithms is König. [13] König represents graphs as Java objects and can solve many different graph algorithms on these graphs. It has the ability to store various properties of the nodes and the arcs that make up the network. Because it is implemented in Java,

these properties can be information that is stored and accessed from existing databases. A graph can be created from information that is stored in different data formats on different computers. König can then dynamically load algorithms, if necessary from different computers, and execute these algorithms on that graph in order to get solutions that can assist decision makers. From an operations research standpoint, König provides the power to model networks and to analyze the raw data about the network using existing algorithms. A König graph is the specific network model that is used in this implementation, but others could be used in other implementations.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. MILITARY ROAD MOVEMENT OPTIMIZER

A specific system based on ADDMOR was implemented to support decision making about the movement of military equipment and personnel across a road network. The Military Road Movement Optimizer (MRMO) uses the ADDMOR architecture described in Chapter II and the current technologies described in Chapter III to provide users with a system that helps them make road movement decisions.

A. DISTRIBUTED DESIGN

The power of the MRMO is that it is designed for a distributed network. Figure 4 shows how different parts of the system can exist on distinct hardware devices connected by a digital network. Each distinct box in the figure represents functions that can be on separate computers connected to the network. The user's digital device is running a web browser. His view of the system is provided to him via HTML pages. He accesses a servlet by connecting to a computer acting as a server by pointing his web browser to the address of the servlet. This server is running an HTTP server and a servlet engine. The server passes the user's request to the servlet running on the same machine. The job of the servlet is to interface with the operations research model of the road network and to control the view the user has of this model. Each servlet creates HTML pages specific to the type of user and his requirements. All of the MRMO servlets take information input by the user and send it to a single application that is the model interface. The model interface retrieves data about the road network and creates a König graph from that data. Road network data can exist on any computer connected to the network. It can be stored

in a flat data file or in a database. The servlet retrieves data about units in the battlespace and adds that data to the König graph model of the road network. Again, unit data can exist anywhere on the computer network. The model interface then finds the appropriate algorithm to execute on this graph. Algorithms can also be distributed throughout the network. The result of executing the algorithm is returned to the servlet, which formats this result in a format appropriate for the specific user and returns them to the user as an HTML page.

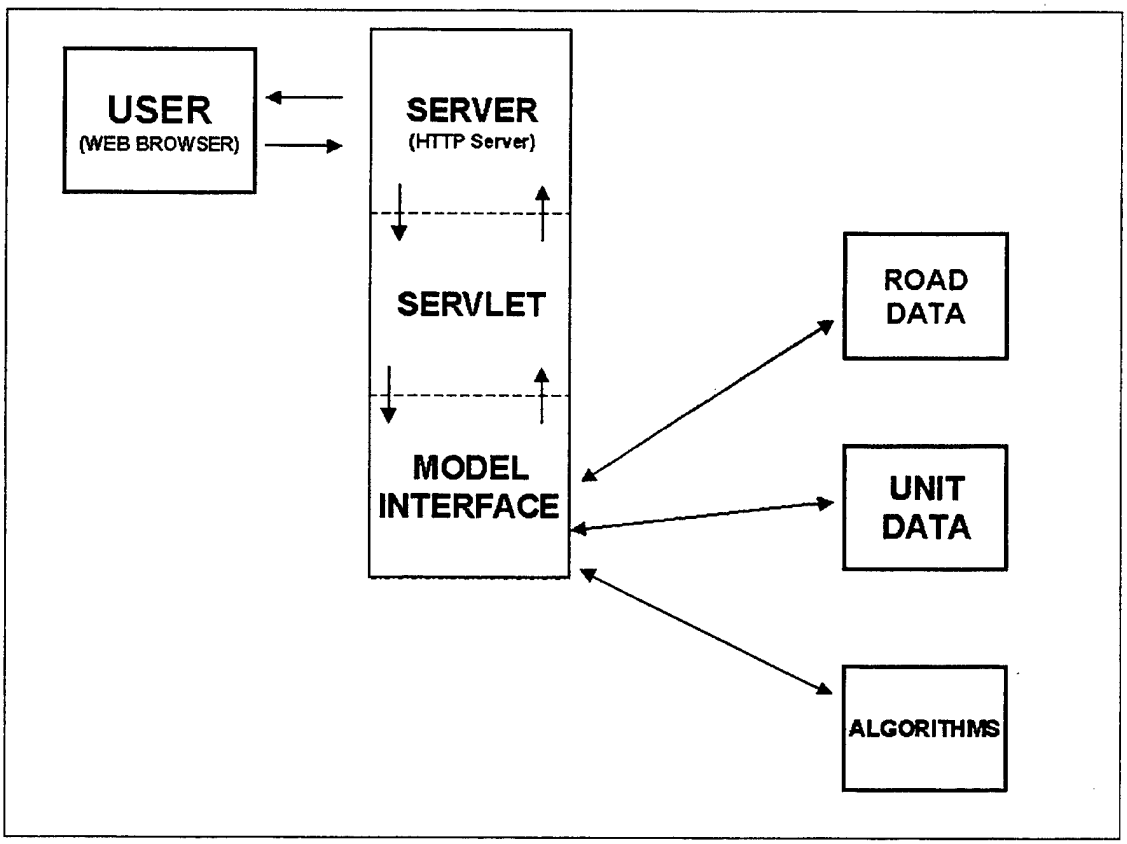

Figure 4. A distributed computing MRMO implementation

The MRMO system consists of several servlets that allow different types of users to interact with this model in order to help them make decisions. Each of these servlets

uses the same model interface to interface with the model. Each servlet formats the model's output into HTML pages tailored for the specific user that servlet is designed for.

B. THE MEDIC

One servlet interacts with a company medic on the battlefield. This medic is moving around the battlefield with a specific unit. When that unit takes casualties the medic needs to know the location of the nearest aid station so that he can evacuate those casualties to the next level of medical care. Currently the company gets information about the status of road networks and medical unit locations from voice radio transmissions and stores that information on a map. If in the heat of battle an updated unit location is not heard or if the map is updated incorrectly, these data will be wrong. The procedure to find the nearest aid station is to look at the map and then estimate the distance to the aid station. This system is fraught with error and can benefit greatly from a system designed to harness the power of the ADDMOR.

Figure 5 contains a view of the introductory HTML page that the servlet constructs for a medic in Microsoft's Internet Explorer web browser. The medic enters the name of the unit he is located with and hits the 'SUBMIT' button. This HTML page contains other inputs for the model as hidden text that the user cannot see. This hidden text includes the name of the algorithm and the values of the parameters for that algorithm. By including this information as hidden text, the user does not need to know anything about the algorithm.

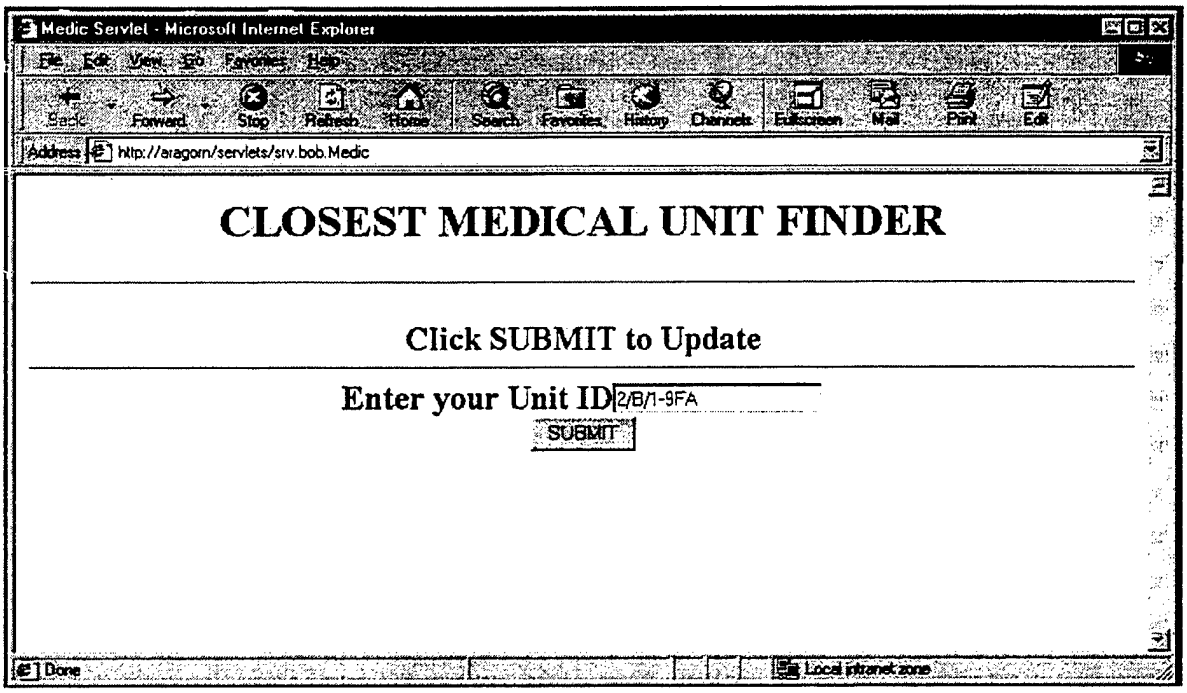

Figure 5. The medic's introductory screen

Figure 6 shows how the HTML page looks when viewed on a small personal digital assistant. The Palm Pilot is 4.7 inches tall by 3.2 inches wide. It weighs only 6 ounces; it is very portable and is an example of the types of small digital devices on the connected battlefield. Because this servlet is designed to return simple text files without pictures or huge displays, it is able to work well within the limitations of the personal digital assistant's small size and display.

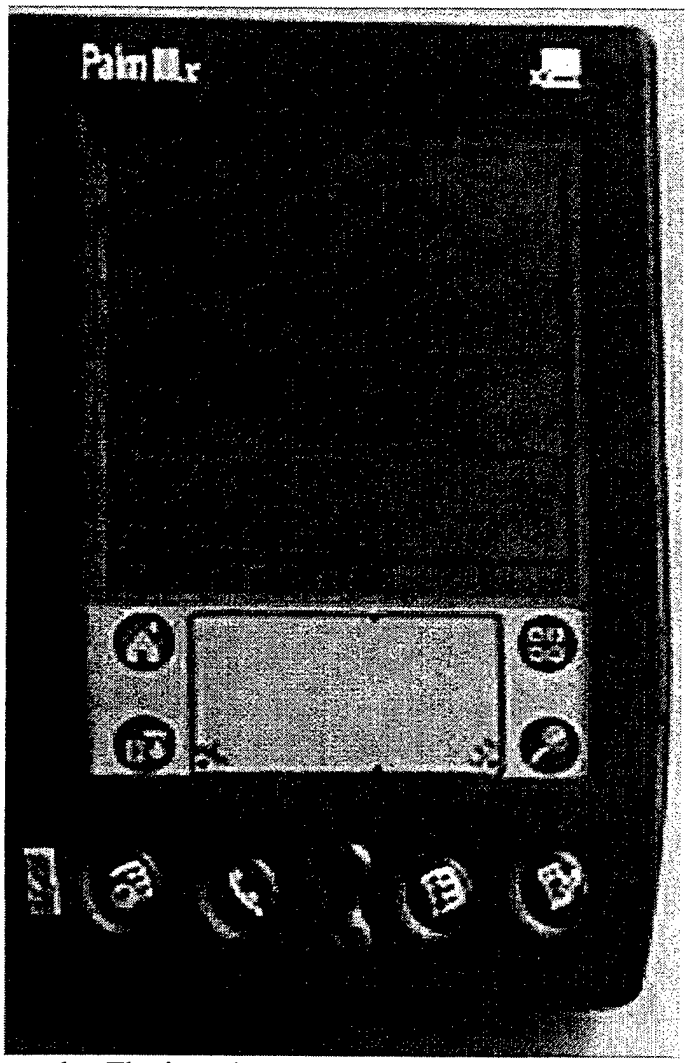

Figure 6. The introductory screen viewed on a Palm Pilot.

Figure 7 shows the solution page returned to the medic by the servlet listing the name, location, and distance to the nearest aid station. The time that the data was sent to the model is also returned to the user. In order to update this result the medic clicks the 'BACK' button on his HTML browser and then can resubmit the input data to solve the algorithm again. Each time the medic clicks 'SUBMIT', the controller creates a current graph of the road network and units by reading from a data store (in the current implementation, two flat text files). The model invokes an algortihm to compute the

shortest path from the entered source node to all other nodes in that graph. This intermediate solution is then searched for all units that have a unit type equal to "medical", and the medical unit with the lowest valued shortest path distance is selected. All properties of that unit are returned to the servlet by the model. The servlet for the medic is designed to display the name, location, and shortest path distance to the nearest medical unit.

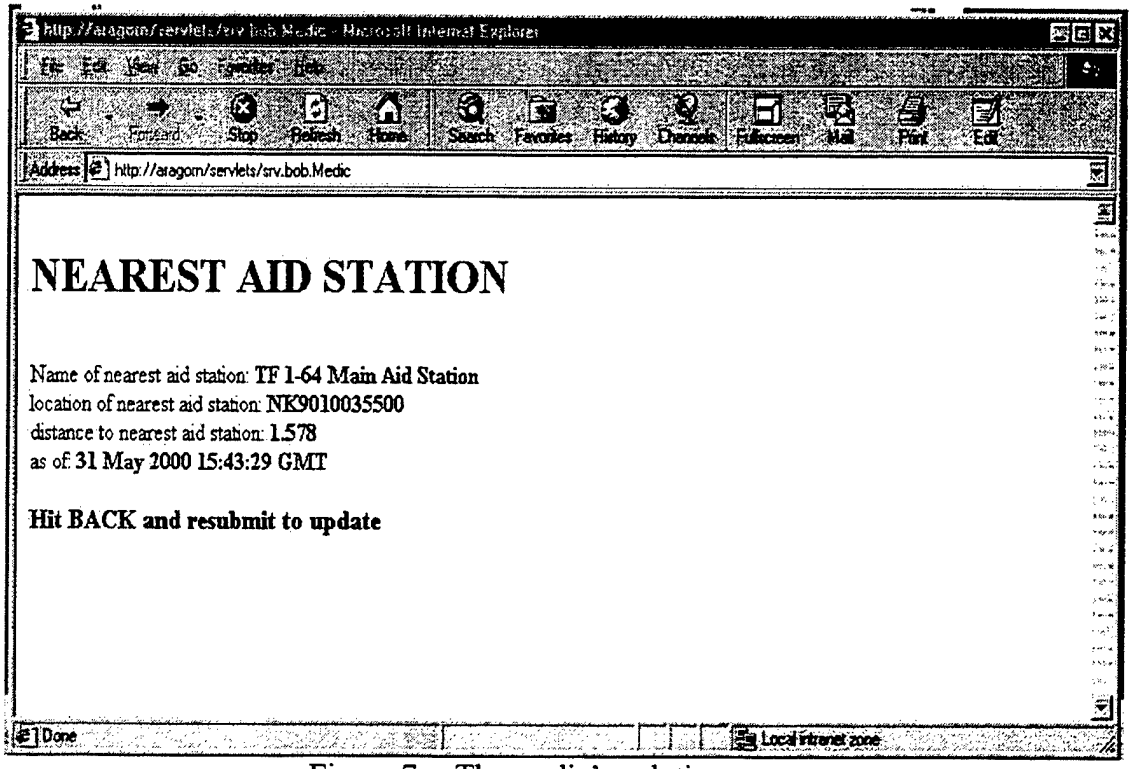

Figure 7. The medic's solution screen

Figure 8 shows the same solution viewed on a Palm Pilot. Since the servlet was designed to return a very simple web page using plain HTML, it is easily viewed with a very simple device with a small display. This is perfect for users who cannot afford to carry large display devices with them due to weight and size constraints.

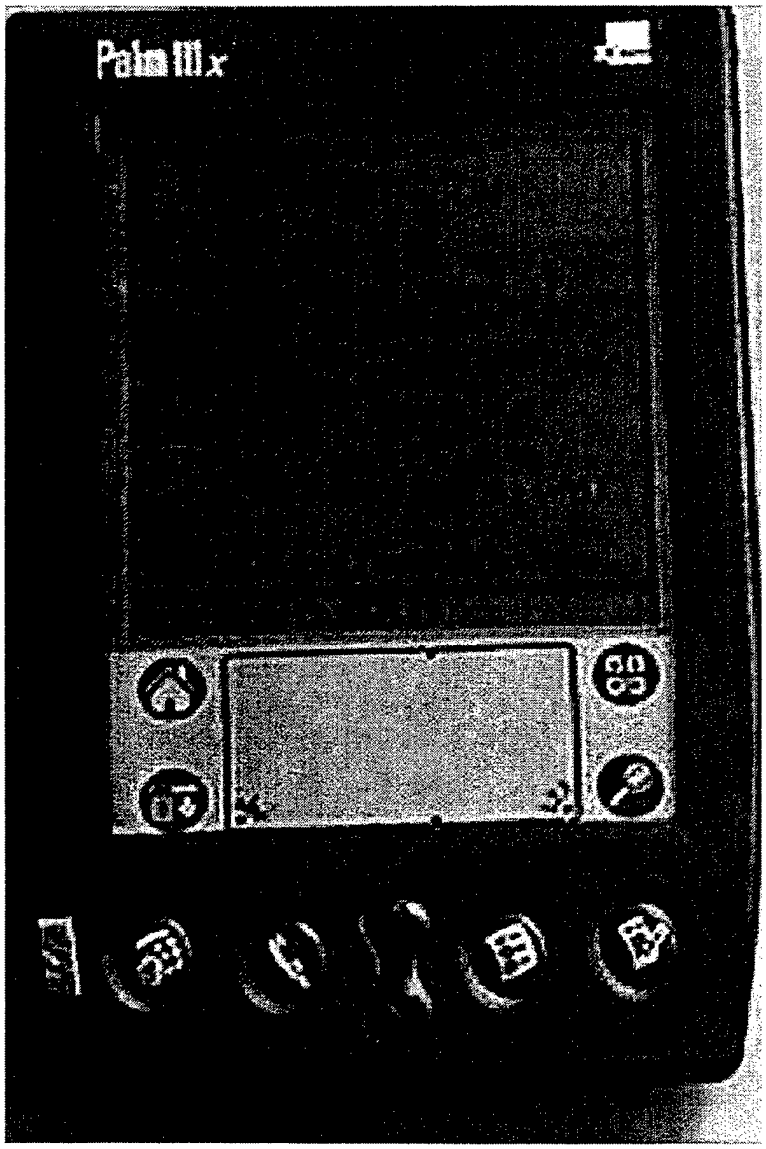

Figure 8. Solution screen viewed on a Palm Pilot

C. THE EXPERT USER

Another MRMO servlet is tailored for an expert user. An expert user has a better knowledge of operations research models and graph algorithms. He is located at a command post and has access to a more powerful computational device.

Figure 9 shows the view that an expert user of MRMO sees. This HTML form allows him to select the type of device he is operating on and the algorithm he wants to solve from pull-down menus. When he clicks 'SUBMIT', a new HTML page is dynamically created based on the algorithm he selected. It will have entry fields for parameters specific to the algorithm he selected. Note that the display in this figure is viewed using a Netscape Navigator browser window, as opposed to the Microsoft Internet Explorer browser used by the medic. As mentioned in Chapter III, these servlets can be accessed from any web browser, and are not tied to a specific software vendor.

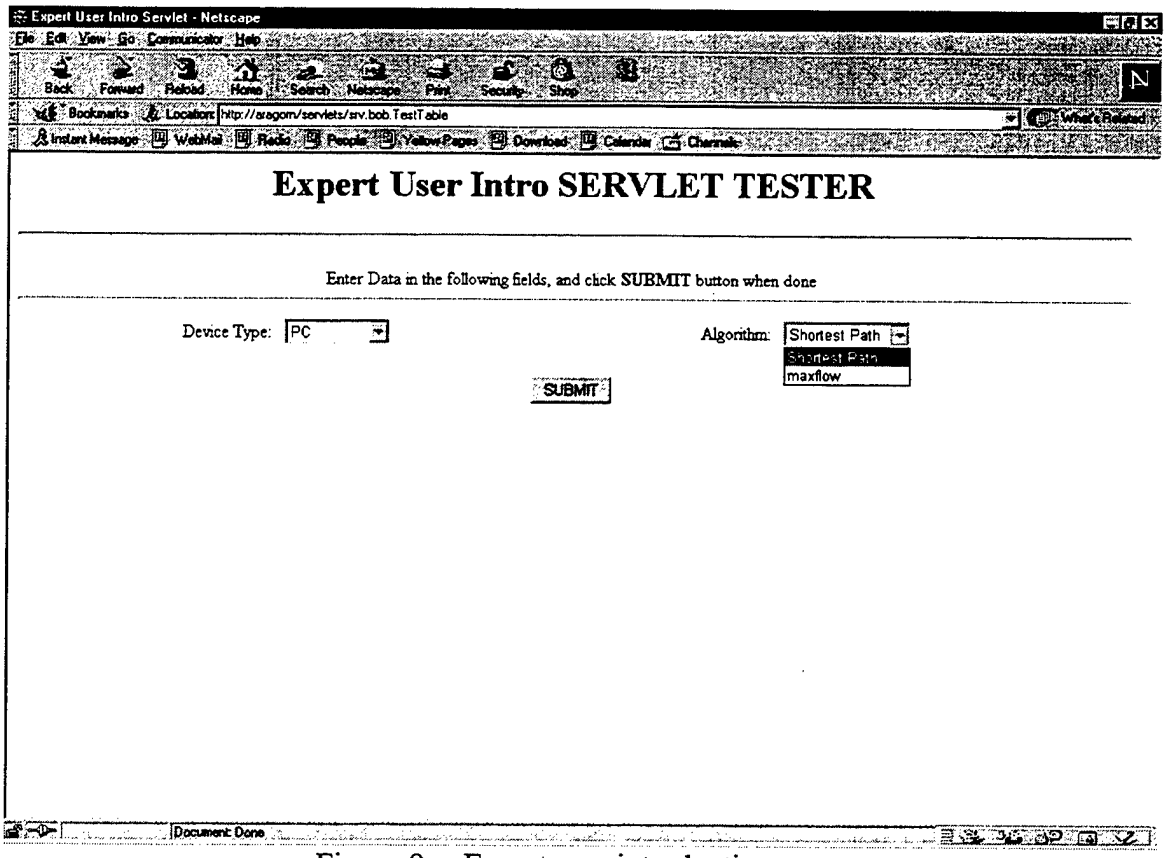

Figure 9. Expert user introduction page

Figure 10 is the page displayed when the expert user selects a shortest path algorithm. At the top of the page is the name of the algorithm that the servlet will call in

the model. On the left are names of the parameters that must be input to the model. In the center are text boxes where the user can enter values for these parameters. In this case the user has typed specific entries into the text fields. After each text field is a list of possible entries for that field, or a prompt for a specific entry. As mentioned earlier, this page is created dynamically based on the user's entries in the previous page. The parameter names and prompts are read from a file at the time the page is created. When new algorithms are added to the model, the model developer must simply modify this one file and the view will be updated.

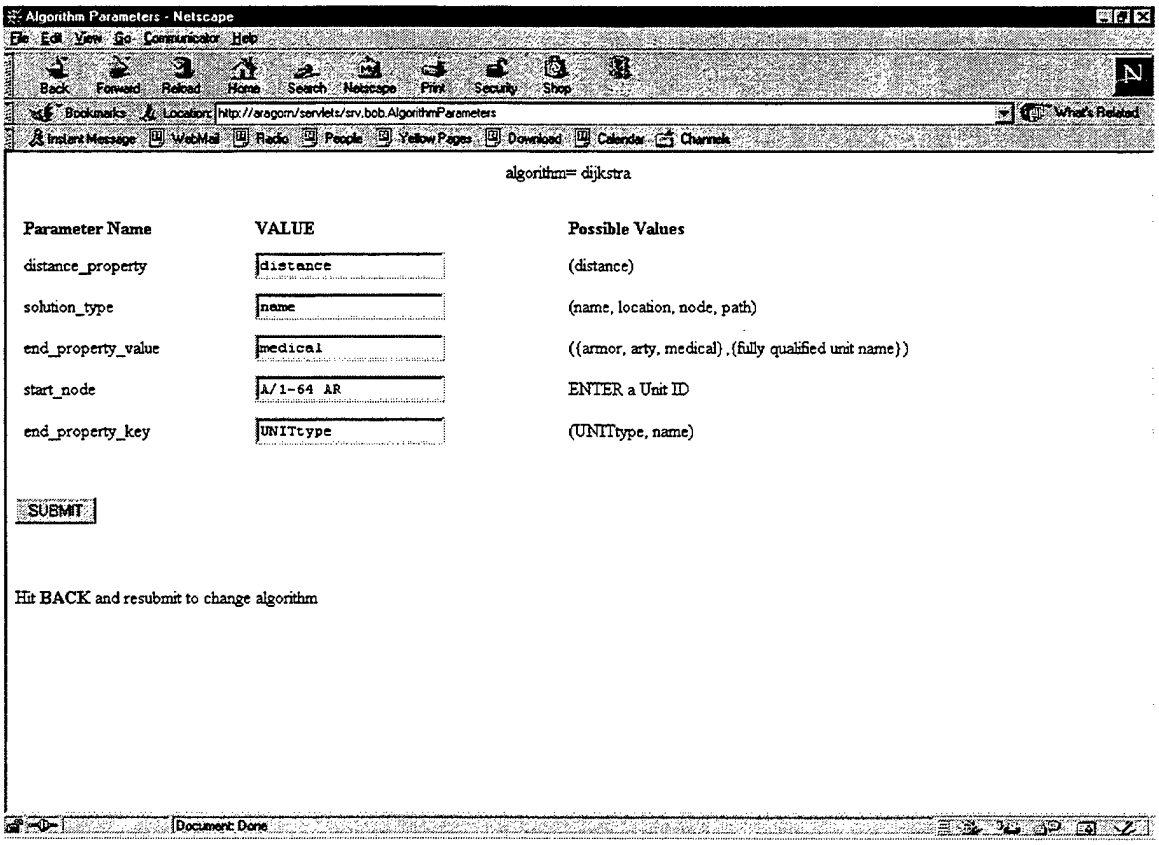

Figure 10. The expert user's next view

The expert user clicks 'SUBMIT' and the servlet puts his entries into the proper format for the model. The model runs and returns a solution to the servlet. The servlet converts that generic solution into the specific solution type the user requested and in a format appropriate for the user's computational device.

Figure 11 shows the expert user's solution page. This solution shows the user the parameters he entered for the problem and the solution to his request. In this case the user asked for a path from his location to the nearest medical unit. It is displayed as a sequential list from his location through a series of road intersection checkpoints, to the final unit.

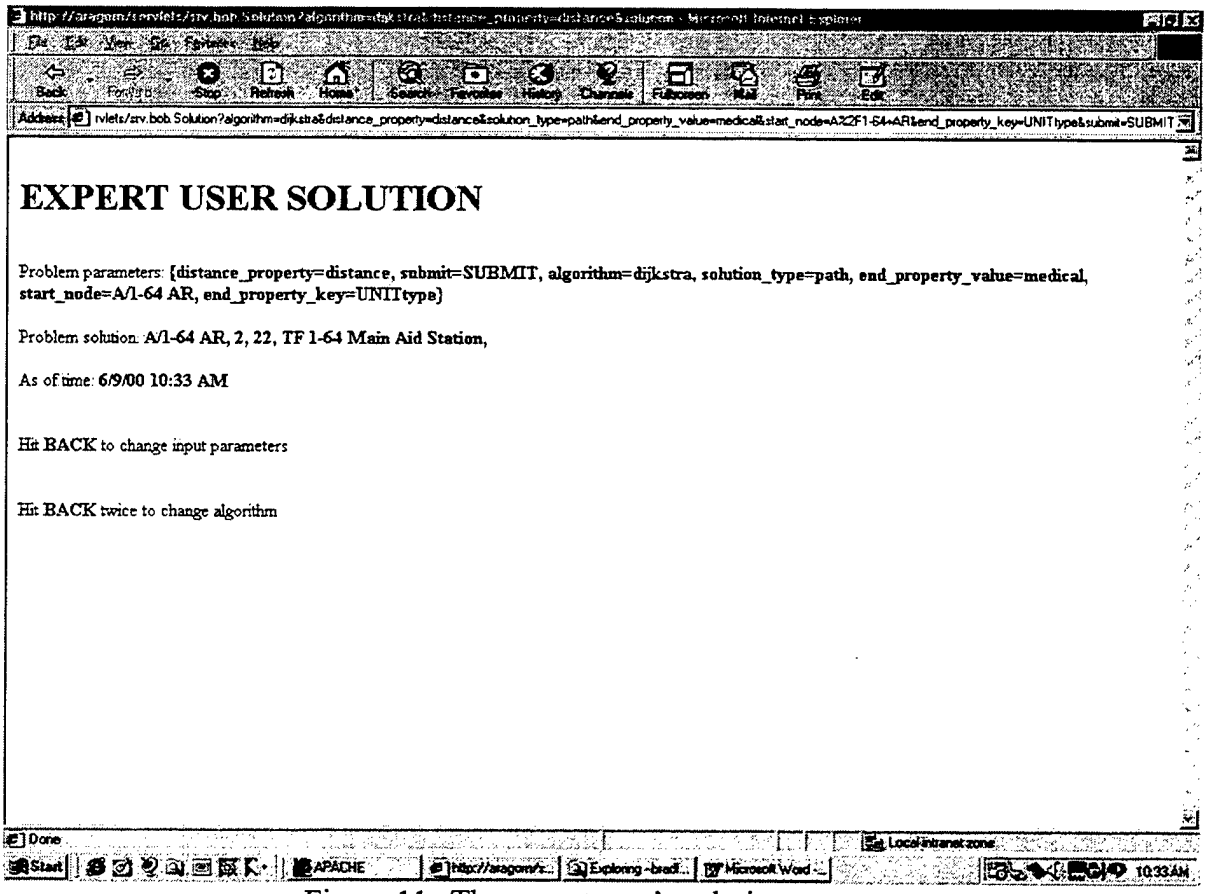

Figure 11. The expert user's solution page

As in the medic's solution, the solution is identified by the time that the data was drawn from the data store. The user knows that if changes to the road network or unit locations occur, he must resolve the algorithm. Resolving the algorithm is a simple matter of using the 'BACK' button in his web browser, and clicking the 'SUBMIT' button again. If he wants to change the algorithm, he must simply return to the introductory servlet page.

If the expert user's device is able to display maps and geographically referenced data, the servlet can return an overlay that shows the route from the user to the nearest aid station. In Figure 12, the route from the artillery unit at the top right of the map to the medical unit in the lower left is displayed on the map in red.

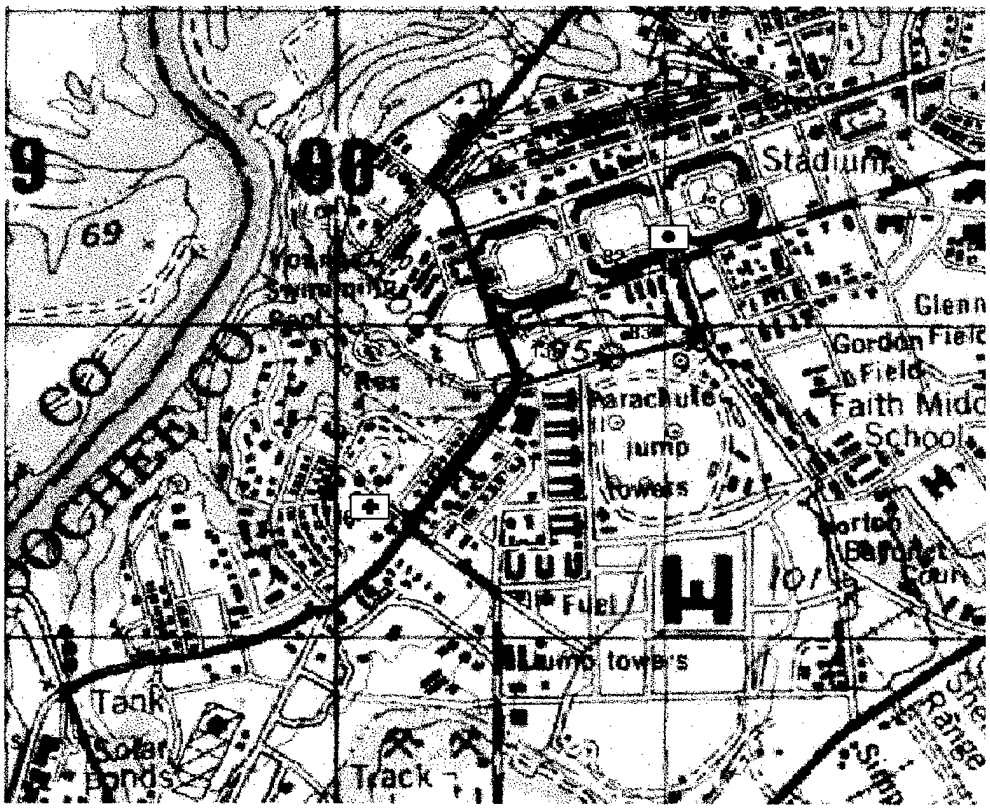

Figure 12. A graphical view of the solution.

Each of these interfaces, or views, that the user gets of the system use the same model and the same controller. The entries and hidden text from the HTML form are put into an HTTP request by the servlet. When the user clicks 'SUBMIT', the controller creates a hashtable from his entries on the form. (A hashtable is a data structure that maps keys to values with keys being are strings listing the parameter names, and values being strings identifying the parameter values.) A user can access any parameter value if he knows the parameter name.

The controller then instantiates a model interface with the hashtable of parameters. The model interface connects to the current information about the roads and units and builds a graph based on this information. The time the data was accessed is put into a hashtable that will be returned to the controller. A specific algorithm is executed on this graph based on the value of the algorithm parameter that the user entered. Based on the solution type parameter, the model interface formats the solution in the form that the user asked for. The model interface then puts the solution into the hashtable, and returns this hashtable to the controller servlet.

The servlet takes the hashtable containing the solution, and builds an HTML page for the user that is in a format appropriate for the user's computational device. The format of this page depends on the type of display the user's device has.

V. OTHER PROBLEMS

Systems based on an ADDMOR design can help decision makers in many military environments. Following are some specific military problems, and how these tools can help solve them.

An artillery battery supporting a Brigade attack occupies a new position and fires a mission. They immediately are attacked by enemy counter-battery artillery fire and begin to take casualties. They must evacuate these casualties as quickly as possible to a medical aid station or soldiers will die. The decision the battery first sergeant must make is where to move the casualties. Medical service in the army is an area support function. The first sergeant should obviously take the soldiers to the aid station that is closest. Since the battle is dynamic, the aid stations are moving to support the brigade. The battery first sergeant is not always monitoring the correct radio frequency to hear the current aid station locations. A system such as MRMO can quickly solve his problem of finding the nearest aid station. It also avoids the problem of operating from old or incorrect data. By operating from a centralized database, the first sergeant's decision can be optimized using the most current data.

A division arriving into theater must move over an existing road network to the forward battle area. The decision for the division's plans section is how to optimize this movement in order to minimize the time required to move the division to the forward area. An ADDMOR system can help the plans section optimize this movement. A system that implements a minimum cost flow algorithm on the road network could

optimize the movement. The movement can be planned before the division arrives in theater. As conditions on the ground change, the planners can resolve the problem, and issue fragmentary orders to the units to keep the movement optimized.

The Corps Support Command needs to send an ammunition convoy forward to drop ammunition at four different Ammunition Transfer Points (ATPs) in the Division Rear Area. The Corps logisticians must make two decisions. First, where should the ATPs be located, and second, what route should the convoy take to see deliver to all four. The first of these problems may require minimizing the longest route a customer of that ATP must travel to receive ammunition. Or it may require minimizing the average route, or minimizing the route of a specific customer unit. An ADDMOR system that is set up to solve this longest shortest path algorithm recursively will greatly help the planners locate their ATPs. The second problem, what route to take, stands out to an operations research analyst as a classic traveling salesman problem. What is the best route to take to go to all of these points and return to the Corps rear area? A system that implements a traveling salesman algorithm can help the logisticians plan the best route for the trucks to take. As the battle progresses, road conditions change, and units move; the problem can be solved again. The system could be customized to solve these two problems sequentially, so that it returns the ATP locations and the route to them. It could also allow a planner to set some or all of the ATP locations as input to the algorithm.

A Joint Targeting Team is planning to interdict an enemy road network. This flow interdiction problem may have the objective of minimizing residual movement along the road network between some specific points, or it may be trying to maximize the time to

travel between two points in the road network. The planners must decide where they should attack the road network to maximize damage to the enemy. An ADDMOR system designed to solve this interdiction problem could be of great value to the targeting cell. As attacks occur, and repairs are made to the road network, the problem could be solved again to plan further attacks.

The architecture of an ADDMOR system is extremely relevant with today's military emphasis on joint, coalition, and interagency operations. It can add benefits over the full spectrum of combat. In humanitarian assistance scenarios, operating with other government agencies, foreign governments, and international relief agencies, this architecture allows for all people working on the project to come with their own computer systems and devices. If they can share access to their computer networks, they can all have access to the same powerful operations research tools to help them make decisions.

Systems based on the ADDMOR design can use models other than network optimization. They could also be used to run a discrete-event simulation to help a decision maker make a decision. An example of an operations research model that could be accessed using this architecture is the convoy simulation developed at the Naval Postgraduate School by Norbert Schrepf [15]. This is a stochastic discrete event simulation that models the movement of convoys over fixed road networks. Another model that could benefit from this architecture is the model for coordinating inland area search and rescue developed by Timothy Castle [16]. This model creates a probability map for the likelihood of finding the target. The probability map is updated as negative

search results are obtained using Bayes Theorem. As data about the search changes, the solution can be updated and a new optimal distribution of search assets returned to the user.

VI. CONCLUSIONS

Future military visions are based on using the digital battlespace to obtain information superiority and dominant battlespace awareness. One important aspect is converting data into information that is valuable for decision makers. Operations research provides many tools to help distill raw data into information that helps people make decisions. A challenge for the future environment will be to design systems that combine the large amounts of data collected about the battlefield with powerful operations research models in real-time and near real-time to provide useful information for decision makers.

The road movement system developed here takes a dynamic road network and a dynamic communications network with distributed data and distributed computing power and combines them with an operations research model to solve road movement problems. The system harnesses the power of currently available, free technologies to convert real-time data into decisions. Because it was designed using low-level communications protocols, the system is accessible from different hardware platforms. This capability of the road movement system was shown by accessing it from a personal computer and a Palm Pilot personal digital assistant. The proven ability to access data on the network and to execute an operations research algorithm on that data, all from small digital devices in real-time and near real-time, demonstrates the potential operations research has to offer decision makers on the future battlefield.

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF REFERENCES

1. Chairman of the Joint Chiefs of Staff, *Joint Vision 2010: America's Military: Preparing for Tomorrow*, Chairman of the Joint Chiefs of Staff, 1996

2. "Army Vision 2010," [http://www.army.mil/2010/default.htm], 1999

3. Cebrowski, Vice Admiral Arthur K., and Garstka, John J., "Network-Centric Warfare: Its Future and Origin," [http://copernicus.hq.navy.mil/divisions/n6/n60/it21/cebrowski.htm], 1997

4. "Army Digitization Master Plan, 1996," [http://www.ado.army.mil/Br&doc/docs/ADMP/admpframes.html]

5. Williamson, Alan R. *Java Servlets by Example*, Manning Publications, 1999

6. Buschmann, Frank, and others, *Pattern-Oriented software architecture: a system of patterns*, John Wiley & Sons, 1996

7. Blair, Obediah, "Update on tactical Internet and Force XXI battle-command for brigade and below," [http://www.gordon.army.mil/regtmktg/AC/SUMR98/TI.htm], 1998

8. "Tactical Internet Communications –C2 On-the-Move," [http://www.gordon-army.mil/tsmtr/ti], 14 Sep 99

9. Netcraft Web Server Survey, [http://www.netcraft.com/survey/], May 2000

10. The Apache Software Foundation, [http://www.apache.org/]

11. "JAVA,"[http://java.sun.com/products/jdk/1.2/] 1999

12. The Java Apache Project, [http://java.apache.org/], 2000

13. Jackson, L., "König," [http://www.trac.nps.navy.mil/jacksonl/König/], 1999

14. Schrepf, Norbert, *Visual Planning Aid For Movement of Ground Forces in Operations Other than War,* Master's Thesis, Naval Postgraduate School, Monterey, California, March 1999

15. Castle, Timothy, *Coordinated Inland Area Search and Rescue (SAR) Planning and Execution Tool*, Master's Thesis, Naval Postgraduate School, Monterey, California, September 1998

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center 2
   8725 John J. Kingman Rd., STE 0944
   Fort Belvoir, Virginia 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, California 93943-5101

3. Dr. Neal Glassman 2
   Air Force Office of Scientific Research
   801 North Randolph Street
   Arlington, VA 22203-1977

4. Professor Gordon H. Bradley, Code OR/Bz 4
   Department of Operations Research
   Naval Postgraduate School
   Monterey, California 93943-5000

5. Professor Arnold H. Buss, Code OR/Bu 2
   Department of Operations Research
   Naval Postgraduate School
   Monterey, California 93943-5000

6. LTC Joel R. Parker, Code OR/Jp 1
   Department of Operations Research
   Naval Postgraduate School
   Monterey, California 93943-5000

7. CPT Robert D. Bradford, III 2
   7715 Sudbrook Square
   New Albany, Ohio 43054