Chapter 8

Computing Technology

CONTENTS

Boxes

Box *Page*

Tables

Table No. *Page*

Chapter 8

Computing Technology

INTRODUCTION

This chapter discusses the demands that ballistic missile defense (BMD) systems would place on computing technology, and the tradeoffs that would have to be considered in satisfying those demands. Initial sections discuss why BMD would need computers and how it would use them for battle management, weapons control, sensor data processing, communications, and simulation. Later sections describe the technology used to build computers and the requirements that the Strategic Defense Initiative (SDI) imposes on that technology. The chapter concludes with key issues posed by SDI computing needs. Any description of computing technology must be accompanied by a discussion of software and software technology issues; these can be found in chapter 9 and appendix A.

The Need for Automation

The rapid response times and volume of data to be processed would require the use of computers in every major BMD component and during every phase of battle. Humans could not make decisions fast enough to direct the battle. The launch of thousands of intercontinental ballistic missiles (ICBMs), some employing fast-burn rocket boosters, might permit less than 60 seconds to detect, track, aim, and fire weapons at the first boosters to clear the atmosphere. During mid-course it might be necessary to account for a million objects and to discriminate among hundreds of thousands of decoys and thousands of reentry vehicles (RVs). In the terminal stage, RVs in the atmosphere would have to be quickly located, tracked, and destroyed by interceptor missiles.

Mutual occupation of space by two defensive systems of comparable capabilities might require considerably faster reaction times than those needed to meet a ballistic missile attack alone. Countering an attack by space-borne directed-energy weapons would require response times of seconds or less to avoid the loss of critical defensive capabilities. The critical part of such a battle could well be over before humans realized that it was taking place.

Although automated decision-making is a focus of concern for the use of computers in BMD, computers would also serve many other purposes. Table 8-1 shows many of the places where computers would be used.

Integrating Sensors, Weapons, and Computers

An automated BMD system would require some degree of coordination among different computers, but there would be many places where computers would act independently of each other. Table 8-1 shows many such cases, e.g., computers incorporated into sensor systems, such as radars, to perform signal processing on data perceived by the sensor. In each case the computer may be specially designed for its job and is physically a part of the system of which it is a component.

As an example, an imaging radar would build up an image of an object such as an RV by analyzing the returns from the object over a period of seconds. The radar would process each return individually and store the results. With sufficient individual returns, the radar could analyze them to form an image of the object. A single computer incorporated into the radar would perform the processing, storage, and analysis. From the viewpoint of an external observer, such as a battle management computer residing on a different platform, the radar is a black box that produces an image of an RV. The external observer need know nothing about the computer inside the

Note: Complete definitions of acronyms and intialisms are listed in Appendix B of this report.

Table 8-1 .—Computers in a Ballistic Missile Defense System

Component	Purpose of Computers
First. and Second. Phase Systems:	
Battle Management Computers[a]	Coordinate track data (e.g, maintain a track data base and correlate data from multiple sensors); maintain status of and control defense assets; select strategy; select targets; command firing of weapons; assess situation.
Boost Phase Surveillance and Tracking Satellite (BSTS)	Process signals to transform IR sensor data into digital data representing potential booster tracks; process images to recognize missile launches and to produce crudely-resolved booster tracks; communicate with battle management computers; maintain satellite platform: guidance, station keeping, defensive maneuvering; housekeeping.
Space Surveillance and Tracking Satellite (SSTS)	Process signals to transform IR, laser range-finder, and radar sensor data into digital data representing potential tracks; process images and data for fine-tracking of launched boosters, post-boost vehicles, RVs and decoys and to discriminate RVs from decoys; point sensors; communicate with other elements of the BMD system; guidance, station keeping, defensive maneuvering, housekeeping (maintain mechanical and electronic systems).
Laser Thermal Tagger	Point the laser beam; communicate with other elements of the BMD system; maintain satellite platform: guidance, station keeping, defensive maneuvering; housekeeping.
Carrier Vehicle (CV) for Space-based Interceptors (SBI)	Monitor status of SBIs; control launching of SBIs; communicate with battle manager; maintain satellite platform: guidance, station keeping, defensive maneuvering; housekeeping.
Space-based Interceptor (SBI)	Guide flight based on commands received from battle manager; Track target and guide missile home to target; communicate with battle manager; housekeeping.
Airborne Optical System (AOS)	Process signals to transform IR data into digital data representing potential tracks; process images and data for fine-tracking of post-boost vehicles, RVs and decoys and, if possible, discriminating RVs from decoys; point sensors; communicate with other elements of the BMD system; control of airborne platform; housekeeping.
Exe-atmospheric Interceptor System (ERIS)	Guide flight; process signals and images from on-board sensor for terminal guidance and target tracking; communicate with battle manager and SSTS, AOS, and probe sensors; housekeeping.
Ground-based Terminal Imaging Radar	Process signals and images to convert radar returns to target tracks; process images and data to discriminate between decoys and RVs; control radar beam; communicate with battle managers and other elements of BMD system; housekeeping.
High Endo-atmospheric Interceptors (HEDI)	Guide missile flight based on commands received from battle manager; process signals and images from on-board sensor for terminal guidance and target tracking; communicating with battle manager; housekeeping.
Third Phase, Add:	
Ground-based Laser, Space-based Mirrors	Manage laser beam generation; Control corrections to beam and mirrors for atmospheric turbulence; steer mirrors; communicate with battle manager; housekeeping.
Space-based Neutral Particle Beam (NPB)	Manage particle beam generation; steer the accelerator; track potential targets; communicate with battle manager and neutron detector; maintain satellite platform: guidance, station keeping, defensive maneuvering, housekeeping.
Radiation Detector Satellites	Discriminate between targets and decoys based on sensor inputs; communicate with battle manager and/or SSTS; maintain satellite platform: guidance, station keeping, defensive maneuvering; housekeeping.

[a] May be carried on sensor platforms, weapon platforms, or separate platforms; ground-based units may be mobile.

SOURCE: Office of Technology Assessment, 1988, radar, or how it operates, but only the form and content of its output.

Customizing the Computer for the Application

The above "black box" design strategy is based on sound engineering principles and tends to simplify the battle management architecture, but it still involves some difficult trade-offs. One such trade-off is that between developing special-purpose computers for different sensors and weapons versus utilizing commercially available hardware. Utilizing commercially available computers may simplify the job of software development. There would be people available who have experience with existing hardware. In addition, support tools for software development on available computers already exist. As a result, software developed for commercially available computers would probably be more reliable, more efficient, and less expensive than software developed for new computers built specifically for BMD. Furthermore, software development would not have to wait for development of the hardware, reducing the risk of not meeting schedules.

On the other hand, hardware specially built for BMD is likely to be more efficient and better suited to the job, possibly offsetting efficiency losses in software. Moreover, maintainability, reliability of the hardware, and life-cycle cost would have to be taken into account. Software experts at OTA's SDI Software Workshop suggested that hardware customization v. software reliability and cost was an important trade-off that should be resolved in favor of simplifying software development. However, some SDI computing might require the use of novel hardware designs, even though this might require designing new and complex software from scratch.

Communications and Computer Networks

Battle management requires communications among the battle managers, sensors, and weapons forming a BMD system and between the battle managers and the human operators of the system. Space-to-space, space-to-ground, and ground-to-space communications would be required. As in traditional battle management, information must be sent in useful form, on time, and securely to the place where it is needed. Also as in traditional battle management, information transmitted among battle managers concerns the location of targets and weapons, the status of resources, and decisions that have been made. Distinct from traditional battle management, information transmitted in a BMD system would all be digitally encoded and the transmissions controlled by computers. As noted in chapter 7, the rate and volume of data to be transmitted depend on the battle management architecture.

Estimates of Communications Requirements

The Fletcher Report estimated that the peak data rate needed by any communications channel in a BMD system would be about 10^7 bits per second (bps).' This estimate assumed that an entire track file would have to be transmitted, that the file would contain 30,000 tracks, and that each track could be represented in 200 bits. Except for the number of tracks in a track file, the estimate is based on conservative assumptions. Furthermore, it scales linearly with the number of tracks, i.e., a track file containing 300,000 tracks would require a peak rate of about 10^8 bps.

In more recent work, analysts have made more specific assumptions about architectures and have been able to produce more refined, but still rough, estimates. For example, one study of boost-phase communications uses a highly distributed architecture consisting of sensor satellites and satellite battle groups composed of battle management computers, sensors for booster tracking, and space-based interceptor (SBI) carrier satellites. Additional assumptions were made about numbers of targets tracked per sensor in the battle, number

'James C. Fletcher, Study Chairman and B. McMillan, Panel Chairman, *Report of the Study on Eliminating the Threat Posed by Nuclear Ballistic Missiles: Volume V, Battle Management, Communications, and Data Processing,* (Washington, DC: Department of Defense, Defensive Technologies Study Team, Oct. 1983), p. 19.

of bits per target track, non-uniform message traffic density, number of relays per message, varying message types (examples are track data, status information, and engagement data), and number of seconds per frame. The result was a peak link data rate for boost phase within the transmission rates of current technology.[2]

The Fletcher Report noted, and OTA concurs, that:

> The technology exists today to transmit 10^7 to 10^8 bits/sec over data links of the length and kind needed for a BMD system. Therefore, even with 300,000 objects in the track file, existing communication technology could handle the expected data rates. Cost and complexity will vary with the rate designed for, but the Panel concludes that communication rates, per se, will not be a limiting factor in the design of a BMD system.[3]

Communications Networks

Regardless of volume, communications would have to be secure and reliable. It would have to survive attempts by an enemy to intercept, jam, or spoof communications, at best rendering the system ineffective, at worst taking control of it for his own purposes. It would also have to survive physical damage incurred in a battle or defense suppression attack. Understanding the threats requires understanding how communications would function in a BMD system.

Current communications technology, including that proposed for BMD systems, involves establishing a network of computers, each acting as a communications node, that transmit data to each other. One example of an existing network that is widely distributed geographically is the ARPA network, initially developed by the Defense Advanced Research Projects Agency (DARPA) as an experimental network. Another example is the AT&T long distance telephone network. Both differ considerably from a space-based battle communications network, which would have:

1. more nodes and more available direct connections between nodes;
2. different delays between nodes (perhaps 5 milliseconds for the example distributed space-based network described earlier as compared to more than 25 milliseconds for the ARPA network);
3. more stringent security requirements;
4. a need to re-establish links every few minutes; and, probably
5. long repair times for individual nodes.

Nonetheless, the problems are sufficiently similar that the terrestrial networks are useful examples. Each node in the network communicates with several other nodes. Users of the network communicate by submitting messages to the network.[4] The computers controlling and comprising the network route the message from one node to another until it reaches its destination.

Some of the major issues that must be resolved in designing a communications network are:

- the physical arrangement of the nodes and the interconnections among the nodes;
- the unit of data transmission, which may be a complete message or part of a message;
- the algorithm used to decide what route through the network each unit of data transmission will take;
- the algorithm used to encode units of data transmission so that they may be reliably transmitted;
- the algorithm used by nodes for interchanging data so that the start and end of each data transmission may be determined; and
- the methods used to ensure that data communications are secure and cannot be jammed, spoofed, or otherwise rendered unreliable.

‘Personal communication, Ira Richer, The Mitre Corp.

³The Fletcher Report, *op. cit.*, footnote 1, p. 40.

[4] In the AT&T network, messages are sent across the network to establish a circuit to be used for a long distance call when a subscriber dials a long distance number. Generally, once a circuit is established, it is dedicated to a call, and communications on it may be sent in non-message form as analogue signals or may be encoded digitally into messages.

Message Transmission

Information to be sent over a digital communications network, such as used in BMD systems, is organized into messages. In some networks, known as "packet-switched" networks, for transmission purposes the messages are organized into blocks of data "packets."[7] In a packet-switched network the user submits his message to the network unaware of how the message will be organized for transmission. The software that controls the network must incorporate a method for extracting messages from packets when the packets reach their destinations.

Security of Communications

Secure network communications require that the routing algorithm be correct, that nodes cannot be fooled into sending messages to the wrong recipient, and that the physical communications links are secure from unauthorized interceptions. Since a network by its nature involves access to many computer systems, it affords potential saboteurs a chance to access many different computers. Both the ARPA network and the AT&T telephone network have been fooled on many occasions into permitting unauthorized access to the network and, in the case of the ARPA network, to computers on the network. The managers of both networks continually try to improve their protection against such access, but no workable foolproof protection techniques have been found.[8] As noted by Lawrence Castro, Chief of the Office of Research and Development at the National Computer Security Center,

> Current computer networking technology has concentrated on providing services in a benign environment, and the security threats to these networks have been largely ignored. While literature abounds with examples of hackers wreaking havoc through access to public networks and the computers connected to them, hackers have exploited only a fraction of the vulnerabilities that exist. Techniques need to be developed that will prevent both passive exploitation (eavesdropping) and active exploitation (alteration of messages or message routing.)[9]

Gaining unauthorized access to a BMD communications network would at least require communications technology as sophisticated as that used in the design and implementation of the network. Furthermore, an enemy would have to penetrate the security of the data links, which would likely be encrypted. Since network communications would be used for coordination among battle managers, and would probably involve transmission of target and health data,"[10] the worst result of compromise of the network would be that the enemy could control the system for his own uses. Disruption of communications could result in disuse or misdirection of weapons and sensors, causing the BMD system to fail completely in its mission. To achieve such disruption, it would not be necessary for a saboteur to gain control of a battle management computer, but only to feed it false data. Less subtle ways to achieve the same means might be to destroy sufficient

(continued from previous page)

IEEE Trans. on Communications, vol. COM-26, No 12, December 1978.

The reader should keep in mind that the ARPANET was designed as an experimental network, and not as a high reliability network intended for commercial use.

[7] Depending on the situation, several messages maybe combined into one packet, or one message maybe split across several packets. In either case, the benefit of packet switching is that network resources may be shared, leading to more efficient routing of messages and more efficient use of the network, The disadvantage is that the job of the routing algorithm may be complicated, and routing may become more difficult to debug.

[8] Access to a network is frequently separated from access to the computers using the network. Entrance to the ARPA network is through computers dedicated to that job, known as terminal access computers (TACs). Until recently, such access was available to anyone who had the telephone number of a TAC. Several so-called hackers have made use of TAC facilities to gain entrance to Department of Defense computer systems connected to the ARPA network, and they have been successfully prosecuted for doing so, Partly as a result of such unauthorized use of TACS, password protection has been added to TAC access procedures. The telephone companies wage constant war against people who attempt to use their long distance networks without paying.

[9] Lawrence Castro, "The National Computer Security Center's R&D Program," *Journal of Electronic Defense,* vol. 10, No. 1, January 1987.

[10] The health of a resource, such as a sensor satellite or weapon satellite, is how well the resource is able to perform its mission and what reserves are available to it. Example measures of satellite health are battery power and efficiency of solar cells. For a BMD satellite, such as a carrier vehicle for SBIs, additional data specific to the function of the satellite, such as number of SBIs remaining, would be included.

Since most of these issues are resolved in software, the solutions chosen have a strong effect on the complexity of the software and the reliability of the battle management system as a whole. The more critical of these issues are discussed in the following sections. In almost every case, the trade-off is that adding sophistication to the algorithm(s) used to solve the problem results in software that is more complicated and more difficult to debug.

Network Topology

The arrangement of interconnections among nodes is known as the "network topology." In attempts to improve the efficiency and reliability of networks, numerous topologies have been tried. As an example, until recently the AT&T long distance telephone switching system used a hierarchical topology to establish a circuit to be used for a long distance call.[5] Nodes were organized into levels. Messages requesting the circuit were sent from a lower level to a higher level, then across the higher level and back down to a lower level. If all messages must pass through one or two nodes, then under heavy loads those nodes may form bottlenecks that decrease network performance. If the nodes break down under the traffic load, the network cannot not function at all. As a result, most networks employ algorithms that decide what route each message will take through the network. The route may vary according to the prevailing load conditions and the health of the nodes in the network.

Routing in Networks

In geographically distributed networks with many nodes, the routing algorithm is a sophisticated computer program. Frequently, network performance degrades as a result of incorrect assumptions or errors in the design and implementation of the routing algorithm. Finding and correcting the reason for degraded performance requires knowledge of the network status, including traffic loads at nodes and health of nodes. Since traffic load in particular varies second-by-second, debugging network routing software is a difficult and time-consuming job.

One can only have confidence in relatively bug-free operation by permitting the network to function under operational conditions long enough to observe its performance under varying loads. Stress situations, e.g. especially heavy traffic conditions, tend to cause problems. In operation such conditions are relatively infrequent; they are also hard to reproduce for debugging purposes. Nevertheless, for a dedicated network such as a BMD communications system, it may be easier to simulate heavily loaded conditions than for a commercial network.

Either software failures, such as an error in a routing algorithm, or hardware failures may cause catastrophic network failure. In December, 1986, the east coast portion of the ARPA network was disconnected from the rest of the network because a transmission cable was accidentally cut. Although the ARPA network had evolved over more than 15 years, an opportunity for a single-point catastrophic failure remained in the design.

Sometimes the interaction of a hardware failure and the characteristics of a particular routing algorithm CM cause failure. In 1971 normal operations of the ARPA network came to a halt because a single node in the network transmitted faulty routing information to other nodes. Transmission of the faulty data was the result of a computer memory failure in the bad node. Based on the erroneous data, the routing algorithm used by all nodes caused all messages to be routed through the faulty node. The routing algorithm was later revised to prevent the situation from recurring, i.e., the software was rewritten to compensate for certain kinds of hardware failures.[6]

[5] To help alleviate bottlenecks in the system, AT&T is now moving toward anon-hierarchical system where nodes can communicate directly with each other rather than going through a hierarchy. Note that decisions to change the structure of the long distance system are made as the result of observing its behavior over extensive periods of use by millions of subscribers.

[6] For a more complete description of this problem, see J. McQuillan, G. Falk, and I. Richer, "A Review of the Development and Performance of the ARPANET Routing Algorithm, "

(continued on next page)

communications nodes that routing algorithms become overstressed and fail, or to destroy sufficient nodes that battle managers can no longer communicate with each other. The former attack requires that the enemy have some knowledge of the routing algorithms used; the latter may require considerable expenditure of physical resources such as antisatellite missiles.

Even passive observation of a BMD communications network could reveal enough about the battle management and communications algorithms used by the network to permit an enemy to devise means of circumventing those algorithms and thereby rendering the defense partially or totally ineffective. To prevent an enemy gaining such knowledge by observation of communications, encryption of communication links and techniques for disguising potentially revealing changes in message traffic would have to be incorporated a network design.

Although encryption and other technology could make passive exploitation quite difficult, a saboteur could perhaps gain access to the communications software and hardware. Analysis of the sabotage questions, however, beyond the scope of this study.

Achieving secure, reliable, adequate communications requires the conjunction of at least two technologies. The technology for physical communications, such as laser communications, needs to provide a medium that is difficult to interceptor j am and that can meet the required transmission bandwidth. The network technology must provide adequate, secure service for routing messages to their destinations.

SIMULATIONS AND THE NATIONAL TEST BED

Preceding sections have discussed the role of computers during battle. Computing technology would also play a key role in preparation for battle and in maintaining battle-readiness. Computer simulations (box 8-A) would be needed:

- to anticipate threats against the system,
- to model different ways in which the system might work,
- to provide a realistic environment in which system components may be tested during their development, and
- to test the functioning of the system as a whole, both before and after deployment.

Simulations and Systems Development

Simulators are useful during all stages of the development of complicated systems.

- During the early stages of the development of a system, simulators may predict the behavior of different system designs. An example is simulators that predict stresses on parts of a bridge for different bridge designs.
- During the middle stages of development, simulators may test individual components of a system by simulating those parts not yet built or not yet connected together. An example is a simulator that reproduces the behavior of the different parts of an aircraft before the aircraft's systems are integrated. A radar simulator can feed data to the radar data processor before the radar itself has been finished.
- During testing, a simulator can be used to reproduce the environment in which the system will operate. Avionics computers and their software are tested before installation by connecting them to an environmental simulator that reproduces the flight behavior of the aircraft's systems to which the computers will be connected when installed in the airplane.
- After deployment, simulators test the readiness of systems by mimicking the environment-including stress conditions

Box 8-A.—Simulations

A simulation is a system that mimics the behavior of another system. The difference between the simulation and the system being mimicked (called the target), is that the simulation does not accurately reproduce all of the behavior of the target. Behavior not accurately reproduced is either unimportant to the users of the simulation, unknown to the builders of the simulation, or too expensive to reproduce. Many simulators operate by solving a set of mathematical equations that predict the behavior of the target system under the desired conditions. This process is known as modelling the behavior of the target, and such a simulator is often called a model. Others may do no more than supply a previously determined sequence of values on demand or at fixed time intervals.

Airplane flight simulators are good examples of simulators. Flight simulators used for pilot training reproduce flight conditions well enough to help train pilots how to fly, but not to grant them licenses. No one would trust a pilot all of whose flight time was logged on a simulator. Flight simulators are just not sufficiently accurate reproductions of flight conditions to ensure that the pilot knows what it feels like to fly a real plane. However, a pilot who already has a license may use a simulator to qualify for another aircraft in the same class as his license, e.g., a pilot qualified for a DC-10 could qualify to fly a Boeing 747 based only on simulated flights.

Constructing an accurate simulation requires that the target behavior be well understood and that there be some method for comparing the behavior of the simulator with the behavior of the target. In cases where the physical target behavior is unavailable for comparison, simulator behavior may be compared to other simulators modeled on the same target, or to predictions made by mathematical models of the target. (In cases where the simulator itself is a model, a different model may be used for comparison. If a different model is unknown another simulator already known to be reliable, or hand calculations, maybe used.) A simulator that models the trajectory of a missile in flight can be checked against actual missiles and the equations of motion that are known to govern such trajectories. A simulator that models the behavior of the Sun can only be compared to observed solar behavior, and may be quite inaccurate when used to predict behavior under previously unknown conditions.

–for which it is critical that the system operate correctly. Such simulators are often build into the system and contain means of monitoring its behavior during the simulation. The design of the SAFEGUARD anti-ballistic missile system of the early 1970s incorporated a simulator called the system exerciser" to permit simulated operation of SAFEGUARD during development and after deployment.

Current Battle Simulation Technology

As faster, deadlier, and more expensive weapons, such as guided missiles, have been added to arsenals, the demand on simulation technology to analyze their effects has increased. For example, in the early 1970s, single engagement simulations modeled such events as defending against a single missile attacking a single ship. Such simulations can now be run 30 times slower than real time, i.e., 30 seconds of processor time devoted to running the simulation corresponds to 1 second in an actual engagement. However, the demand is now to develop simulators that can model many missiles against many ships.

Work at the U.S. Army's Strategic Defense Command (USASDC) Advanced Research Center (ARC) is representative of current BMD simulation technology. In late 1986, ARC researchers completed a set of mid-course BMD battle simulations. The simulations empoyed 6 Digital Equipment Corporation VAX 11/780 computers coupled by means of shared memories. Four of the computers could simulate battle managers, one simulated surveillance sensors (all of the same type) and weapons (ground-launched homing interceptors of the Exoatmospheric Reentry vehicle Interceptor

System), and one simulated 32 other engaged platforms."

The ARC researchers ran three battle management design cases:

1. 36 battle managers that communicated among each other, known as the *distributed* case;
2. a single centralized battle manager; and
3. 36 autonomous battle managers that did not communicate with each other, known as the *autonomous* case.

The centralized and distributed cases assumed 236 and 237 interceptors respectively, and the autonomous case assumed 660. The maximum threat simulated was 1,000 objects, which required 7 hours to run. The centralized and distributed cases took 3 and 4 hours respectively to run against a threat of 216 RVs. The simulation took 15 months to develop, and included about 150,000 lines of code, much of it in the Pascal programming language. (Code for the battle managers was replicated for some simulations; the replication is not included in the 150,000 lines.)

Perhaps the largest stumbling block in running larger scale and more realistic simulations for the ARC is the lack of computing power. SDIO expects the EV88 experiment sequence, running through fiscal year 1990, to conduct larger scale simulations involving the ARC, the Airborne Optical Adjunct, prototype space-based BMD components, and the National Test Bed. This series of experiments will require considerably more computing power than is now in place at the ARC.

Simulation experts agree that computing power is currently the major limitation in performing large scale simulations. However, other factors complicate the situation. Where equipment or environments are not well-understood or include many random variables, the accuracy of simulations is difficult to verify. This is the case, for example, in simulations of sea conditions surrounding missile v. ship engagements.

Some military simulation experts noted to OTA staff that every time they performed simulated threat assessments without prior access to the real equipment being modeled, the behavior of the real equipment surprised them. They strongly emphasized that it was only when a simulation could be compared to an actual experiment that the verisimilitude of the simulation could be checked.[12] The implication for BMD is that actual Soviet decoys and missiles would have to be examined and observed in operation to simulate their workings accurately. Similarly, the battle environment, including nuclear effects—where appropriate—and enemy tactics, would have to be well understood to conduct a battle simulation properly.

The National Test Bed

The SDIO is sponsoring the development of a National Test Bed (NTB)—a network of computers and a set of simulations to execute on those computers. A threat model is to simulate the launch of Soviet missiles and display their trajectories after launch. Another model would simulate a complete BMD battle to exercise a deployed BMD system.

The NTB would be utilized in all phases of the development and deployment of a BMD system. It should permit experimentation with various system and battle management architectures, battle management strategies, and implementations of architectures. It would be the principal means of testing BMD system components and subsystems as well as the entire BMD system, thereby providing the basis for their reliability.

Preliminary design work studies for the estimated $1 billion NTB were completed in December 1986.[13] Initially, the NTB is to be a network of computers, each simulating a different aspect of a BMD engagement. The number of computers linked for any particular engage-

[11] Depending on the architecture being simulated, the other platforms were either battle managers or sensors.

[12] Experience cited here is drawn from discussions with scientists from the Naval Research Laboratory's Tactical Electronic Warfare Division about simulations of Naval warfare.

[13] Major James Price, SDIO's assistant NTB director, described the NTB as a $1 billion program through 1992 in an interview reported in Defense Electronics in February, 1987.

ment would vary depending on the completeness and depth of detail required. Initial capabilities would not permit simulation of a full battle involving hundreds of thousands of objects.

A major use of the NTB would be to conduct experiments with different BMD technologies and strategies. The currently visualized NTB would link sensors, weapons, or battle managers to simulations that reproduce the data they would handle during a battle. The object could then be tested under varying conditions. The results of such experiments would be quite sensitive to the verisimilitude of the simulations. Accordingly, it is important that there be a way to verify the accuracy of the simulations used in NTB tests and experiments.

Computers in Support of BMD System Development

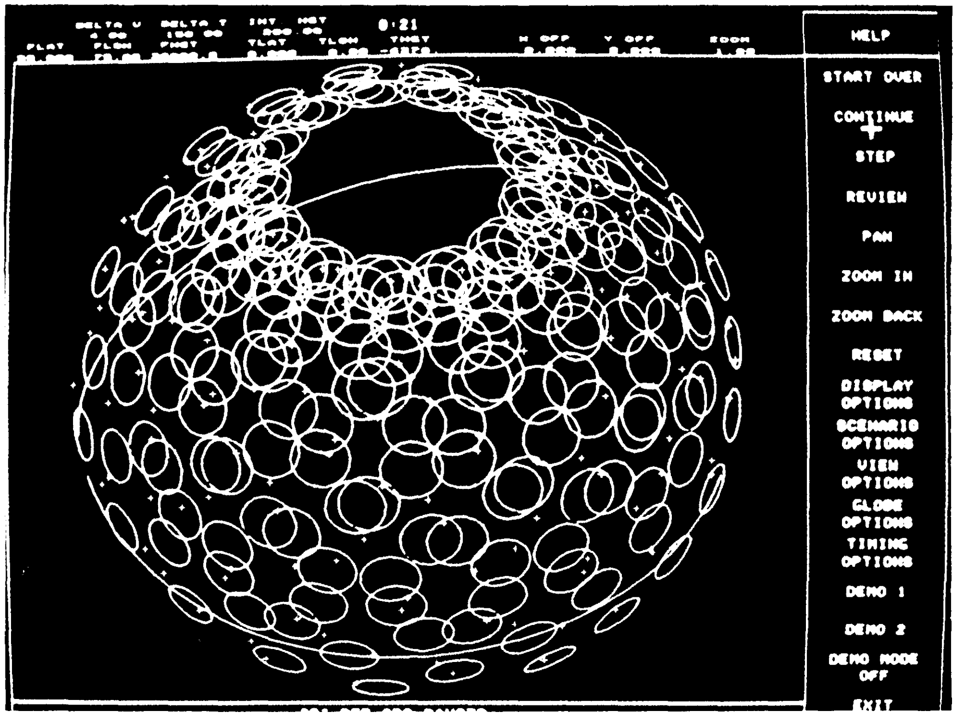
A BMD system to counter the Soviet ballistic missile threat might be the most complicated system ever built. It would involve the use of many different technologies, the automated interplay of thousands of different computers, sensors, and weapons, and the development of more software than has been used in any single previous project. Accordingly, managing the development of such a system would require considerable computer support to track progress, to identify problems, and to maintain the status of components under development, in test, and deployed.

Computers would also be used to design, generate, and test system hardware and software. Engineers and managers are likely to be geographically dispersed and would need to transfer information from one computer to another. The interaction among people would only be effective if there were a means for effective interaction among the computer systems that they use. Previous sections of this chapter have concentrated on the role of computers in the operation and testing of a BMD system. But it is clear that effective computing technology would be needed not just in a strategic battle, but long before system deployment and throughout the lifetime of the system.

Computing Technology Trade-offs

Chapter 7 and the preceding sections have portrayed some of the trade-offs involved in using computers for ballistic missile defense. The following list summarizes those trade-offs.

- *Processing power required v. volume of data communications among battle managers.* Sharing information among battle managers relieves them of some of the tasks that they might otherwise have to perform, and decreases the processing load on each of the battle managers, but increases the data communications rate requirements and also requires that communications be secure and reliable.
- *Performance v. volume of data communications.* Sharing data among battle managers allows the system to operate more efficiently, but, as in the previous trade-off, greater dependence on communications requires greater communications capacity, reliability, and security.
- *Performance v. degree of automation.* Permitting human intervention during a battle degrades performance under some conditions, but may permit recovery from failures caused by the inability of an automated system to recover from unanticipated and undesired events.
- *Processing power required v. battle management organization.* A distributed organization would require less processing power from each computer but more communications than centralized battle management, which requires placing a considerable concentration of processing power in one computer system.
- *Software complexity v. battle management organization.* A hierarchical battle management architecture simplifies the software design but may leave the system less survivable because of the possibility of command layers being disabled. A decentralized battle management structure would increase the complexity of the communications software and might require more weapon resources, but might result in a more survivable system.
- *Software expense and reliability v. hardware customization.* Customizing hard-



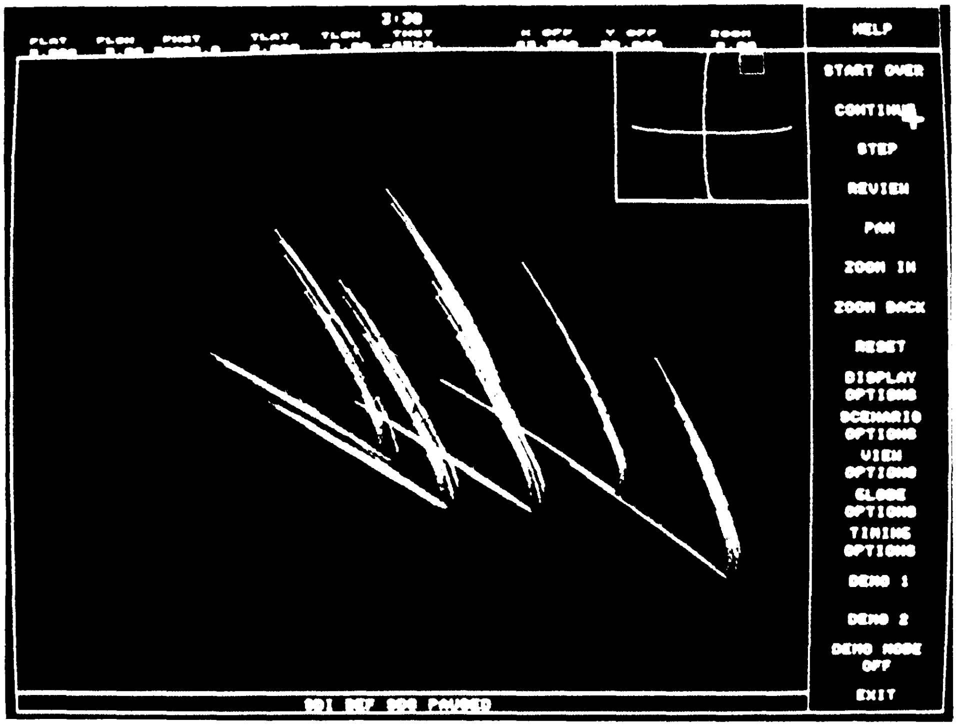


Photo credit: Electronic Systems Division, US Air Force Systems Command

Computer simulations are to play a key role in the development and testing of SDI systems. The photographs here are of video displays screens generated by a ballistic missile defense battle simulation program developed at the MITRE Corporation. The circles in the scene above depict areas of coverage for a system of space-based interceptors. The scene below indicates the tracks of ICBM boosters a few minutes after launch.

ware to perform efficiently at specific tasks could improve hardware capabilities, but might result in longer software development schedules and decreased software reliability because of lack of experience with and lack of development tools for the hardware.

These trade-offs represent important architectural issues that strongly affect the computing technology needed for BMD. For most of them, the SDI system architects have not yet explored the alternatives in sufficient detail to be able to quantify choices. As a result, there are still only crude estimates of the speeds and sizes of the computers needed, and the rates at which data would have to be communicated among the elements of a BMD system.

PROCESSOR TECHNOLOGY

Table 8-1 shows many of the places where computers would be used in the fighting components of a BMD system and the jobs they would perform. Rough estimates of the required memory capacities and speeds are in included in the classified version of this report.[14] Estimates of computer performance requirements for various BMD functions are shown in table 8-2.

Processing requirements are highly dependent on the system and battle management architectures, and on the threat. Without detailed architectural specifications and a precise specification of the algorithms to be used, estimates of speed and memory requirements accurate to better than a factor of 10 probably cannot be made.

Because of the variety of jobs they would perform, BMD computers would vary considerably in speed and memory capacity. Special purpose computers would probably execute some computing tasks, such as signal processing. General-purpose computers faster than any now existing would probably be needed for computationally stressful tasks such as discrimination of RVs and decoys in mid-course. All space-based computers would have to be radiation-hardened beyond the limits of existing computers.[15]

In addition to their use in the fighting components of the system, computers would also be used:

1. in simulators;
2. to help design, test, exercise, and train people in the use of the system; and
3. to assist in supporting the system throughout its lifetime.

Capabilities of Existing Computers

The processing power of a computer is determined by the operating speed of its components and the way they are interconnected (see box 8-B). Processing and memory components are built from semiconductor chips, whose speed is limited by the number and arrangement of circuits that can be placed on a chip. Developments in chip design and production technology, including advances from large scale integrated circuits (LSI) to very large scale integrated circuits (VLSI), have increased processor speeds for general purpose computers by a factor of three to four approximately every 2 years for about the past 10 years. Much of this progress has been the result of refinements in chip design and production. As a result, some existing supercomputers, such as the Cray XMP series or Cray 2, may be close to satisfying most SDI data processing needs, except that such machines are not packaged in a suitable form.

[14] For many of the system elements shown in table 8-1, estimates for processing speed and size are not available. The most computationally intensive tasks are probably signal processing for the IR and optical sensors incorporated into BSTS, SSTS, and AOS, especially for the mid-term and far-term architectures.

“Radiation hardening to within an order of magnitude of SDI requirements for some critical components of computer systems has been demonstrated. A complete computer system that is space qualified and radiation-hardened to within an order of magnitude of SDI requirements for spaceborne computers has yet to be built.

Table 8-2.—Computing Performance Requirements

SDS functions	Present state of the art	DEM/VAL objective	First Increment SDS requirement	Follow on SDS requirement	Risk reduction programs
Space based					
General purpose *processing* hardware					
• Command defense • Maintain positive control • Assess situation • Select and implement mode • Coordinate with others and higher authority • Maintain readiness • Reconfigure and reconstitute . Engagement management • Weapon guide and home • Assess kill • Data distribution	• Throughput: 1 MIPS (space qualified)	• Throughput: 500 MIPS • Memory: 500 MBYTES • Architecture: heterogeneous • Technology: VHSIC (CMOS), GaAs, SOI . MTBF: 10 years	• Throughput: 10-50 MIPS • Memory: 1000 MBYTES • Architecture: heterogeneous . Technology: VHSIC (CMOS), GaAs, SOI • MTBF: 10 years	• Throughput: 50-150 MIPS . Memory: 1000 MBYTES • Architecture: heterogeneous • Technology: VHSIC (CMOS), GaAs, SOI • MTBF: 20 years	• VHSIC • DARPA strategic computing • SD I BM/C^{3} • MCC
Special purpose computing hardware					
• Sense and bulk filter • Track • Type and d incriminate • Data distribution	. Throughput: 350 MFLOPS (space qualified)	• Throughput: 2000 MFLOPS • Memory: 500 MBYTES • Architecture: heterogeneous . Technology: VHSIC (CMOS), GaAs, SOI • MTBF: 10 years	• Throughput: 500 MFLOPS • Memory: 1000 MBYTES • Architecture: heterogeneous . Technology: VHSIC (CMOS), @i/k, SOI • MTBF: 10 years	. Throughput: 1-10 MFLOPS . Memory: 1000 MBYTES . Architecture: heterogeneous . Technology: VHSIC (CMOS), GaAs, SOI . MTBF: 20 years	• DARPA • SDI BM/C^{3} • SD I sensors • Commercial
Common hardware characteristics					
	• Space qualified • Hardness (unshielded) Total dose: 10^{4} rad Upset: 10^{-9}/sec Survive: 10° rad/sec Neutrons/cm*: 10^{19}	• Space qualified • Hardness (unshielded) Total dose: 3x 10^{7} rads Upset: 10^{-11}/sec Survive: 10^{11} rads/sec Neutrons/cm’: TBD • Shielded: none • Fault tolerant • Secure	• Space qualified • Hardness (unshielded) Total dose: 3 x 10^{7} rads Upset: 10^{-11}/sec Survive: 10^{11} rads/sec Neutrons/cm^{2}: TBD • Shielded: 5 x JCS • Fault tolerant • Secure	• Space qualified • Hardness (unshielded) Total dose: 3x 10^{7} rads Upset: 10^{-11}/sec Survive: 10” rads/sec Neutrons/cm’: TBD . Shielded: 10x JCS • Fault tolerant • Secure	• DARPA • SDI BM/C^{3} • SD I sensors
Software					
• All above	• FORTRAN, JOVIAL	• Size Element: 0.5 MSLOC Total: 1.5-3 MSLOC • Fault-tolerant • Secure . Ada, COMMON LISP, C • SA/PDL	• Size Element: 1 MSLOC Total: 5 MSLOC • Fault-tolerant • Secure • Ada, COMMON LISP, C	• Size Element: 2 MSLOC Total: 5-10 MSLOC . Fault-tolerant • Secure • Ada, COMMON LISP, C	• AdaJPO • SPC • SE I • SDI BM/C^{3} • DARPA • STARS
Ground based					
Genera/ *purpose processing hardware*					
• Command defense • Maintain positive control • Assess situation • Select and implement mode . Coordinate with others and higher authority	• Throughput: 30-100 MIPS • Technology: bipolar LSI	• Throughput: 500 MIPS • Memory: 500 MBYTES • Architecture: heterogeneous . Technology: VHSIC (CMOS) • MTBF: 1 year	• Throughput: 10-50 MIPS • Memory: 1000 MBYTES • Architecture: heterogeneous • Technology: VHSIC (CMOS) . MTBF: 1 year	• Throughput: 10-50 MIPS . Memory: 1000 MBYTES • Architecture: heterogeneous • Technology: VHSIC (CMOS), GaAs, SOI • MTBF: 1 year	• VHSIC • DARPA strategic computing • SDI BM/C^{3} • MCC

Table 8-2.—Computing Performance Requirements—continued

SDS functions	Present state of the art	DEM/VAL objective	First increment SDS requirement	Follow on SDS requirement	Risk reduction programs
. Maintain readiness • Reconfigure and reconstitute • Engagement management . Weapon guide and home . Assess kill • Data distribution					
Special purpose computing hardware					
• Sense and bulk filter . Track . Type and discriminate . Data distribution	. Throughput: 100-1000 MFLOPS vector processing	• Throughput: 2000 MFLOPS • Memory: 500 MBYTES • Architecture: heterogeneous • Technology: VHSIC (CMOS) • MTBF: 1 year	• Throughput: 1-3 GFLOPS • Memory: 1000 MBYTES • Architecture: heterogeneous • Technology: VHSIC (CMOS) • MTBF: 1 year	• Throughput: 1-10 GFLOPS • Memory: 1000 MBYTES • Architecture: heterogeneous • Technology: VHSIC II (CMOS), GsAs, SOI • MTBF: 1 year	• DARPA • SDI BM/C^3 • SDI sensors . Commercial
Common hardware characteristics					
			• Fault tolerant Redundant Performance monitor Fault location • Hardness 3 PSI plus associated effects	• Fault tolerant Redundant Performance monitor Fault location • Hardness 5 PSI plus Associated effects	• DARPA • SDI BM/C^3 • SD I sensors
Software					
• All above • Readiness, test, health and status report	• FORTRAN, JOVIAL	• Size Element: 0.5 MSLOC Total: 1.5-3 MSLOC • Fault-tolerant • Secure · Ada, COMMON LISP, C • SA/PDL	• Size Element: 1.6 MSLOC Total: 3 MSLOC • Fault-tolerant • Secure • Ada, COMMON LISP, C	• Size Element: 2.3 MSLOC Total: 5 MSLOC • Fault-tolerant • Secure • Ada, COMMON LISP, C	• Ada J PO • SPC . SE I • SDI BM/C^3 • DARPA • STARS

SOURCE: Strategic Defense Initiative Organization, U.S. Department of Defense, 1987

Box 8-B.–MIPS, MOPS, and MEGAFLOPS

The processing power of a computer is often expressed as the rate at which it can execute instructions, measured in instructions per seconds, or ips. A computer that can execute a million instructions per second is a 1 mips machine. Although mips give a crude measure of the speed of a computer, there is too much variability in the time it takes to execute different instructions on the same machine and in the instructions used by different machines for mips to be a true comparative measure of processing power.

Complex instructions may take four or five times longer to execute than simple instructions on the same machine. A complex instruction on one machine may have the same effect in two-thirds the time as three simple instructions on a different machine. To simulate operating conditions, a mix of different instructions are often used in measuring computer performance. Such measurements are sometimes characterized as operations per second, or ops, rather than ips. A computer that can execute a million operations per second is called a 1 mops machine. BMD signal processing needs have been estimated to be as much as 50 billion ops (50 gigops).

One class of instructions, known as floating point instructions, are important in numerical calculations involving numbers that vary over a wide range, but are very costly in terms of execution time. A common option on computers is an additional processor, sometimes known as a floating point accelerator, specialized to perform floating point operations. The speed of computers designed to perform numerical floating point operations efficiently is usually measured in floating point operations per second, or flops. A computer that can execute a million floating point instructions per second is a 1 megaflops machine.

To compensate for differences in instruction sets and instruction effects on different computers, standard mixes of instructions are used to compare the performance of different computers. For applications involving widely-ranging numerical calculations, such as track correlation, floating point instructions are included in the mix. The variation in machine performance between machines may be a factor of three or four, depending on the mix, the machines involved, and other factors.

For purposes of estimating processing power needs for SDI BMD, the requirements are not yet known to better than a factor of about 10, which dominates differences in performance on different instruction mixes. Accordingly, estimates in this report will generally be given in terms of mips or mops.

If progress can be continued at the same rate as in recent years, sufficiently powerful processors to meet the most stressing requirements of SDI BMD should be available in about 10 years. An obstacle to satisfying BMD processing power requirements is that the processors with the largest requirements are those that would have to be space-based and therefore radiation hardened. Special development programs would be needed to produce adequate space qualification and radiation hardening for the new processors.

New Computer Architectures

Current chip production technology may soon reach physical limitations, such as the number of off-chip connectors and the size of the features used to construct circuits on the chip. Increases in processor speeds may then have to await new chip production technology or new ways of building processors, e.g., optical techniques. An alternative to increasing computer speeds without improving component speeds is to find better ways of interconnecting components, i.e. better computer ar*chitectures,* and better ways of partitioning computing tasks among computers. Computers constructed by interconnecting many small computers in ingenious ways, such as the Hypercube computers developed at Cal Tech and later produced by Intel as the iPSC machine, are just now appearing on the market.[16]

[16] C.L. Seitz, “The Cosmic Cube,” Communications ***of the ACM***, January 1985.

The iPSC is estimated to run at 100 mips and 8 mflops, but is well-suited only for scientific computing tasks that can be organized to take advantage of the iPSC's architecture. Whether or not such architectures will be useful for the most computationally-intensive BMD tasks will depend on what algorithms are used.

Novel computer architectures, despite their potential processing power, have the drawback that the software technology base needed to capitalize on their potential must be developed. New software is needed to run programs on new computers, to help users decompose their problems to utilize the machine's potential, and to convert existing software to execute on the machine. As an example, to meet Department of Defense (DoD) standards, a computer such as the iPSC would need a compiler for Ada™ (the DoD's standard programming language for weapon systems) and an operating system compatible with Ada™. Although advances in computing hardware have come rapidly, software development is notoriously slow and costly.

Space Qualification and Radiation Hardening

Space-qualified general purpose computers lag ground-based computers in processing power by a factor of 20 or more. The fastest space-qualified-but not radiation-hardened—processors today achieve processing rates of about 1 mips.[17] Adequate radiation hardening of the computers imposes a more significant penalty in cost than in processing speed. The most promising technology for meeting both speed and radiation hardening requirements currently uses gallium arsenide (GaAs) rather than silicon in the manufacture of chips. Although GaAs is more radiation resistant, high defect densities reduce manufacturing yields, making chip production costlier. The higher defect densities also impose smaller chip sizes and fewer electronic circuits per chip. The consequent lower overall level of integration may require processors to have more components and be less reliable. Researchers in chip production say that current problems with manufacturing yields and circuit densities are temporary and will be solved. As Milutinovic states,

> . . . many problems related to materials are considered temporary in nature, and one prediction states that the steady-state cost will be about one order of magnitude greater for GaAs than for silicon.[18]

Space-based computers must be able to withstand long-term cumulative doses of radiation and neutron flux, short bursts of a few highly-energetic particles (known as transient events), and electromagnetic pulses (EMP) resulting from nuclear detonations. Although shielding may protect semiconductors against all three phenomena, it incurs a corresponding weight penalty. Gallium arsenide is a promising material for semiconductors because it is more resistant to cumulative radiation and neutron flux damage than silicon. Resistance of GaAs to transient events is dependent on the particular chip design.

It may be possible to harden space-based computers to survive the radiation of a nuclear weapons battle environment. But it is important to consider the effects of such an environment on *software* as well as on hardware. A transient radiation-caused upset might interrupt the current operation of computer hardware, leading either to a resetting of the processor or to the changing of a bit in memory or in the internal circuitry of the processor. The processor may continue to function, but the state of the computation maybe altered, causing an error in software processing, i.e., a system failure.

Consider as an analogy the effects of a single digit error on the computation of an entry for an income tax form. The error may be so small as to be hardly noticeable, and it may even make no difference because the tax scales

[17] The Sperry 1637 and Delco MAGIC V avionics processors achieve a rate of about 1 mips, but neither are radiation-hardened nor have they been used in space applications. The Rockwell IDF 224 and Delco MAGIC 362S space-qualified processors achieve a rate of about 600 kops for instruction mixes that do not include floating point operations.

[18] Veljko Milutinovic, "GaAs Microprocessor Technology, " *Computer,* October 1986, pp. 10-13.

are incremental, not continuous. On the other hand, a larger error in a single digit may have a considerable effect on the amount of tax paid. In either case, the error may propagate through later entries on the form until it is noticed and corrected. Unless the taxpayer checks his entries for reasonableness, he may not find the error. The IRS may find the error by duplicating the taxpayer's calculations, or by performing consistency and reasonableness checks.

The effects of transient events on computing accuracy are difficult to predict. Designing software to cope with such events is a formidable problem, requiring one to forecast all possible symptoms of upsets and provide error-recovery measures for them.[19] It is also difficult to simulate the occurrence of transient events realistically enough to test the software design. There is little experience with software-intensive systems operating under conditions likely to produce transient events.

[19] TWhe design problem maybe simplified somewhat by grouping possible symptoms into classes so that all events in a particular class may be handled in the same way. Grouping events into classes and devising the appropriate response for each class is a very difficult design problem.

CONCLUSIONS

A BMD system to counter the Soviet ballistic missile threat might be the most complicated artifact ever built. It would involve the application of many different technologies; the automated interplay of thousands of different computers, sensors, and weapons; and the development of more software than has been used in any single previous project. An advanced BMD system would require computers in every fighting element of the system and in many supporting roles.

The degree of automation demanded entails not only advances in software technology (addressed in chapter 9) but also advances in secure computer networking, processing power, and radiation hardening of electronics. The extent and importance of simulations—in developing, exercising, and otherwise maintaining the system, as well as in training people in its use—would require an advance in simulation technology.

Because several difficult architectural trade-offs have not yet been sufficiently addressed, the scope of the advances needed cannot be well predicted. Until an architectural description is available that clearly specifies battle management structure and allocates battle management functions both physically and within that structure, better predictions will not be possible.

Further discussion of the computing technology issues involved in producing an automated BMD system follows.

Reliable, Secure Communications

Common to all BMD systems that require human intervention at any stage is the need to provide secure, rapid communications between the human and the battle management computers. If part of the system is in space, then most likely there would be a need for space-to-ground communications. Battle management requires communications among the battle managers, the sensors, and the weapons forming a BMD system. The computers forming the communications network would digitally encode and control all the transmissions.

Achieving secure, reliable, adequate communications would call for simultaneous advances in at least two technologies. First, hardware technology, such as laser communications, needs to provide a medium that is difficult to intercept or jam and that can meet the required transmission bandwidth. Second, network technology must provide adequate, secure, survivable service for routing messages to their destinations. When damaged, the network must be able to reconfigure itself without significantly disrupting communications. Such performance would take sophisticated network control software-probably beyond the current state of the art. Proposed solutions to these problems are either untried or have only been tried in ground-based laboratory situations.

Simulations

Simulations would play a key role in all phases of a BMD system's life cycle. The SDIO is building a National Test Bed (NTB) to facilitate the development and use of BMD simulation technology. A full-scale NTB should permit experimentation with different system and battle management architectures, different battle management strategies, and different implementations of architectures. It would be the principal means of testing and predicting component, subsystem, and system reliability. Initially, the NTB would be a link among computers, each simulating a different aspect of a BMD engagement. The number of computers linked for any particular engagement would vary with the completeness and depth of detail required. Initial capabilities would not permit simulation of a full battle involving hundreds of thousands of objects. Battle simulations on a scale needed to represent a full battle realistically have not been previously attempted. It would be crucial, but very difficult, to find a way of verifying the accuracy of such simulations, when and if they are developed.

Technology and Architectural Trade-offs

Many difficult trade-offs have yet to be adequately addressed in the design of a BMD system to meet SDI requirements. Novel design ideas or advances in computing technology may decrease the importance of some of these trade-offs. However, no architecture has yet been specified sufficiently to permit clear trade-off studies. Issues that should be addressed include:

- simplifying software at the cost of adding computational burden to the hardware,
- simplifying battle management software by structuring it hierarchically at the expense of survivability,
- increasing survivability by decentralizing battle management at the expense of increasing communications complexity,
- customizing hardware for specific applications at the expense of increased software development cost and decreased software reliability,
- simplifying the problem of communications security at the cost of decreasing the possibilities for human intervention during battle,
- increasing the amount of human control during battle at the expense of fighting efficiency, and
- improving fighting efficiency at the cost of increasing the complexity and volume of communications (and, thereby, the risk of catastrophic communications failure).

None of these trade-offs is easy to make and few can be quantified. Compounding the difficulty is that many of the system elements—e.g., the Boost-phase Surveillance and Tracking System and Space Surveillance and Tracking System sensors, SBIs and associated CVs, high-powered lasers, and neutral particle beams-are still in the research or development stages. Moreover, no previous system has ever required the automated handling of many different devices and different kinds of devices as would an SD I missile defense. Nonetheless, tentative conclusions on some trade-offs have been reached. Most trade-offs could be properly explored by use of an appropriate simulation, such as might be provided by a full-scale National Test Bed.

Computational Requirements

Processing requirements are highly dependent on the system design, the battle management architectures, and the threat. Because detailed architectural and algorithmic specifications for an SD I BMD system are not yet available, estimates of speed and memory requirements accurate to better than a factor of 10 probably cannot be made. However, progress in processing speed has been rapid historically. If it continues at the same pace, it should yield sufficiently powerful processors to meet SDI needs within 10 years or less. Such processors might still have to be space qualified and radiation hardened.

An additional problem in providing radiation-hardened computing hardware is the lack of experience in building software tolerant of radiation-induced faults. There is little experience with complex, large-scale software systems that must operate efficiently despite the occurrence of radiation-induced transient effects in the hardware.