F COMMERCE
andards and Technology

NIST Technical Note 1261 Volume 1

Mining Automation Real-Time Control System Architecture Standard Reference Model (MASREM)

James Albus, Richard Quintero, Hui-Min Huang, and Martin Roche

Coal

Mine

Automation

The National Institute of Standards and Technology[1] was established by an act of Congress on March 3, 1901. The Institute's overall goal is to strengthen and advance the Nation's science and technology and facilitate their effective application for public benefit. To this end, the Institute conducts research to assure international competitiveness and leadership of U.S. industry, science and technology. NIST work involves development and transfer of measurements, standards and related science and technology, in support of continually improving U.S. productivity, product quality and reliability, innovation and underlying science and engineering. The Institute's technical work is performed by the National Measurement Laboratory, the National Engineering Laboratory, the National Computer Systems Laboratory, and the Institute for Materials Science and Engineering.

The National Measurement Laboratory

Provides the national system of physical and chemical measurement; coordinates the system with measurement systems of other nations and furnishes essential services leading to accurate and uniform physical and chemical measurement throughout the Nation's scientific community, industry, and commerce; provides advisory and research services to other Government agencies; conducts physical and chemical research; develops, produces, and distributes Standard Reference Materials; provides calibration services; and manages the National Standard Reference Data System. The Laboratory consists of the following centers:

- Basic Standards[2]
- Radiation Research
- Chemical Physics
- Analytical Chemistry

The National Engineering Laboratory

Provides technology and technical services to the public and private sectors to address national needs and to solve national problems; conducts research in engineering and applied science in support of these efforts; builds and maintains competence in the necessary disciplines required to carry out this research and technical service; develops engineering data and measurement capabilities; provides engineering measurement traceability services; develops test methods and proposes engineering standards and code changes; develops and proposes new engineering practices; and develops and improves mechanisms to transfer results of its research to the ultimate user. The Laboratory consists of the following centers:

- Computing and Applied Mathematics
- Electronics and Electrical Engineering[2]
- Manufacturing Engineering
- Building Technology
- Fire Research
- Chemical Engineering[3]

The National Computer Systems Laboratory

Conducts research and provides scientific and technical services to aid Federal agencies in the selection, acquisition, application, and use of computer technology to improve effectiveness and economy in Government operations in accordance with Public Law 89-306 (40 U.S.C. 759), relevant Executive Orders, and other directives; carries out this mission by managing the Federal Information Processing Standards Program, developing Federal ADP standards guidelines, and managing Federal participation in ADP voluntary standardization activities; provides scientific and technological advisory services and assistance to Federal agencies; and provides the technical foundation for computer-related policies of the Federal Government. The Laboratory consists of the following divisions:

- Information Systems Engineering
- Systems and Software Technology
- Computer Security
- Systems and Network Architecture
- Advanced Systems

The Institute for Materials Science and Engineering

Conducts research and provides measurements, data, standards, reference materials, quantitative understanding and other technical information fundamental to the processing, structure, properties and performance of materials; addresses the scientific basis for new advanced materials technologies; plans research around cross-cutting scientific themes such as nondestructive evaluation and phase diagram development; oversees Institute-wide technical programs in nuclear reactor radiation research and nondestructive evaluation; and broadly disseminates generic technical information resulting from its programs. The Institute consists of the following divisions:

- Ceramics
- Fracture and Deformation[3]
- Polymers
- Metallurgy
- Reactor Radiation

[1]Headquarters and Laboratories at Gaithersburg, MD, unless otherwise noted; mailing address Gaithersburg, MD 20899.
[2]Some divisions within the center are located at Boulder, CO 80303.
[3] Located at Boulder, CO, with some elements at Gaithersburg, MD.

NIST Technical Note 1261 Volume 1

Mining Automation Real-Time Control System Architecture Standard Reference Model (MASREM)

James Albus, Richard Quintero, Hui-Min Huang, and Martin Roche

Robot Systems Division
Center for Manufacturing Engineering
National Institute of Standards and Technology
Gaithersburg, MD 20899

Sponsored by:
U.S. Bureau of Mines
Pittsburgh, PA 15236

Issued May 1989

NOTE: As of 23 August 1988, the National Bureau of Standards (NBS) became the National Institute of Standards and Technology (NIST) when President Reagan signed into law the Omnibus Trade and Competitiveness Act.

U.S. Department of Commerce
Robert A. Mosbacher, Secretary

National Institute of Standards and Technology
Raymond G. Kammer, Acting Director

National Institute of Standards and Technology
Technical Note 1261 Volume 1
Natl. Inst. Stand. Technol.
Tech. Note 1261 Vol. 1
60 pages (May 1989)
CODEN: NTNOEF

U.S. Government Printing Office
Washington: 1989

For sale by the Superintendent of Documents
U.S. Government Printing Office
Washington, DC 20402

CONTENTS

MINING AUTOMATION
REAL-TIME CONTROL SYSTEM ARCHITECTURE
STANDARD REFERENCE MODEL (MASREM)

VOLUME 1

1. INTRODUCTION

1.1 Scope of this Document

The Mining Automation Standard Reference Model (MASREM) described in this document was adapted by the National Institute of Standards and Technology (NIST)* under the sponsorship of the U.S. Bureau of Mines, from the NIST Automated Manufacturing Research Facility (AMRF). This document is intended as a reference document for the specification of control systems for mining automation projects. It provides the high-level design concepts to be used in the automation of a mine. MASREM implementation examples are given in Volume 2 of this document. Volume 2 describes the application of MASREM to specific mining equipment.

MASREM defines a logical hierarchical architecture for mining automation derived from a number of concepts developed in previous and on-going research programs such as the hierarchical control system developed for the Automated Manufacturing Research Facility at the National Institute of Standards and Technology {[Al 81], [Al 82], [Ba 84], [Mc 83], [Si 83]}. The AMRF control system was developed for simultaneously controlling a number of robots, machine tools, and materials transport systems in a machine shop.

The MASREM architecture defines a set of standard modules and interfaces which facilitates software design, development, validation, and test, and makes possible the integration of software from a wide variety of sources. Standard interfaces also provide the software hooks necessary to incrementally upgrade future mining automation systems as new capabilities develop in computer science, robotics, and autonomous system control.

The Bureau of Mines, with its research program at the Pittsburgh Research Center, has been of great assistance during this architectural development project in explaining mining concepts.

* Formerly the National Bureau of Standards (NBS) prior to 8/23/88

1.2 Background

The Mining Automation Standard Reference Model control system architecture is hierarchically structured into multiple levels, as shown in figures 1.1, 1.2, and 1.3, such that a different fundamental operation is performed at each level. At level one (servo/actuator level), outputs consist of electrical currents, voltages, hydrodynamic pressure, etc. At level two (primitive level), mechanical dynamics are computed. At level three (elementary-move or e-move level), strings of intermediate poses which define motion pathways that are free of collisions and kinematic limits and singularities are determined within the range of equipment mounted sensors. At level four (equipment level), tasks on objects are decomposed into strings of symbolic "elementary" moves. At level five (section level), tasks to be performed by a functional group are decomposed into simple tasks to be performed by a single machine. These tasks typically require the coordination of more than one piece of mining equipment depending on the mining method. At level six (production level), the coordinating of the production jobs and support activities for the different mine sections take place. It is also responsible for the allocation of resources to these jobs. At level seven (facility level), the managing and coordination of the performance of all mining and support operations within the mine take place. These levels are described in greater detail in section 3 of this document and Volume 2 of this document.

The MASREM architecture is also horizontally partitioned into three sections: Task Decomposition, World Modeling, and Sensory Processing. Task Decomposition includes planning and task monitoring, value driven decisions, servo control, and interfaces for operator input. World Modeling includes Computer-Aided-Design (CAD) models of objects and structures, maps of areas and volumes, lists of objects with their features and attributes, and tables of state variables which describe both the system and the environment. Sensory processing includes signal processing, detection of patterns, recognition of features, objects, and relationships, and correlation and differencing of observations vs. expectations. These functions are described more thoroughly in sections 2, 5, and 6.

The MASREM architecture described in this document incorporates many artificial intelligence concepts such as goal decomposition, hierarchical planning, model driven image analysis, blackboard systems, and expert systems. It integrates these into a framework that also includes modern control concepts such as multivariable state space control, model reference adaptive control, dynamic optimization, and learning systems. The framework also readily accommodates concepts from operations research, differential games, utility theory, and value driven reasoning {[Al 88], [Ba 81], [Bo 83], [Gl 79], [Ha 85], [Kn 86], [Pe 78], [Sh 84]}.

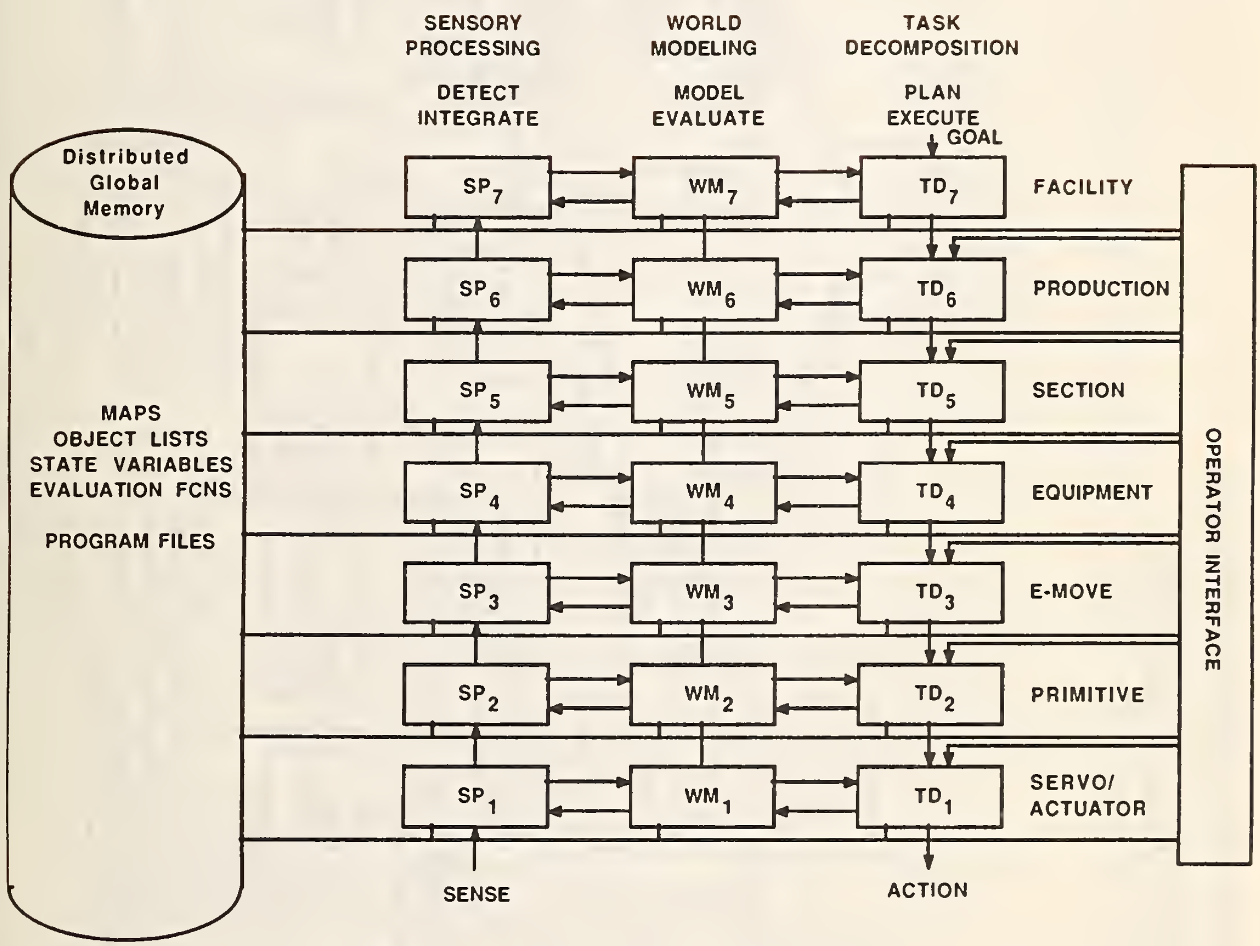


Figure 1.1: High Level Block Diagram of Mining Control System Architecture

MINE AUTOMATION HIERARCHICAL REAL-TIME CONTROL SYSTEM

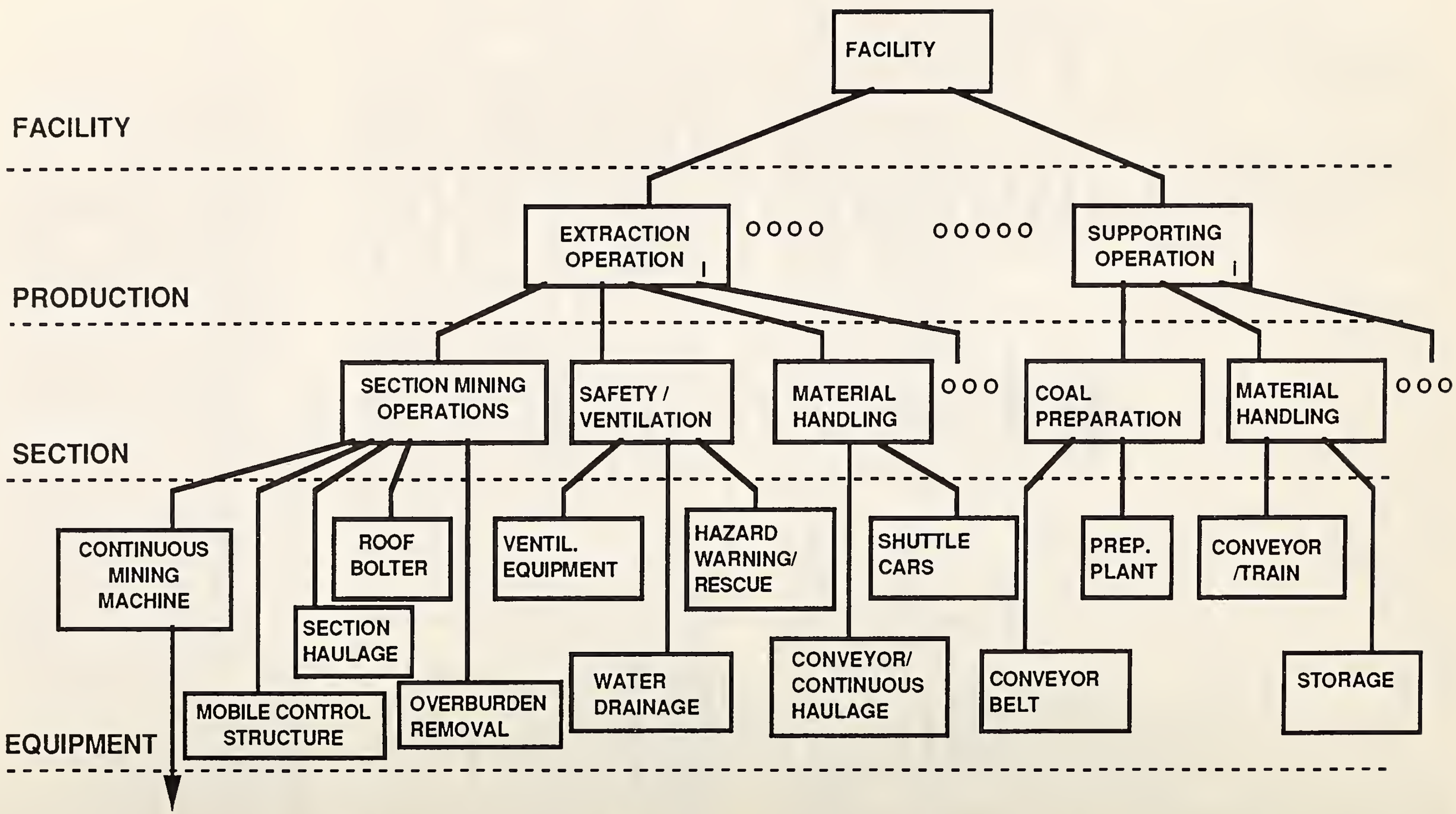
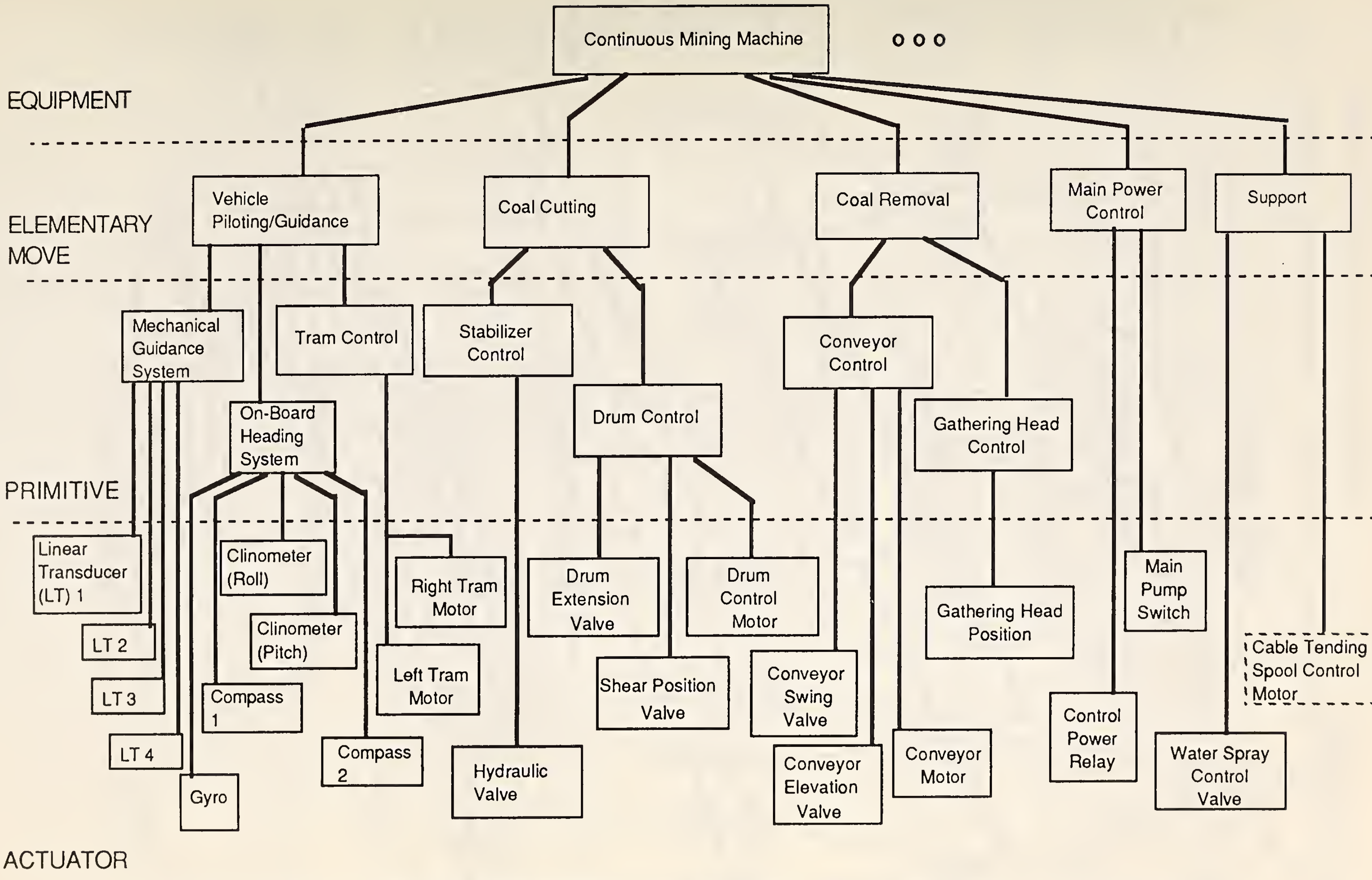


Figure 1.2: Mine Automation Hierarchical Real-Time Control System Architecture (for Higher Levels)



Note: Functional Decomposition may change if continuous haulage is used; for example, the Cable Spooling Control may appear on the haulage unit, rather than shown here as a manually controlled support function.

Figure 1.2: (Continued, for Lower Levels)

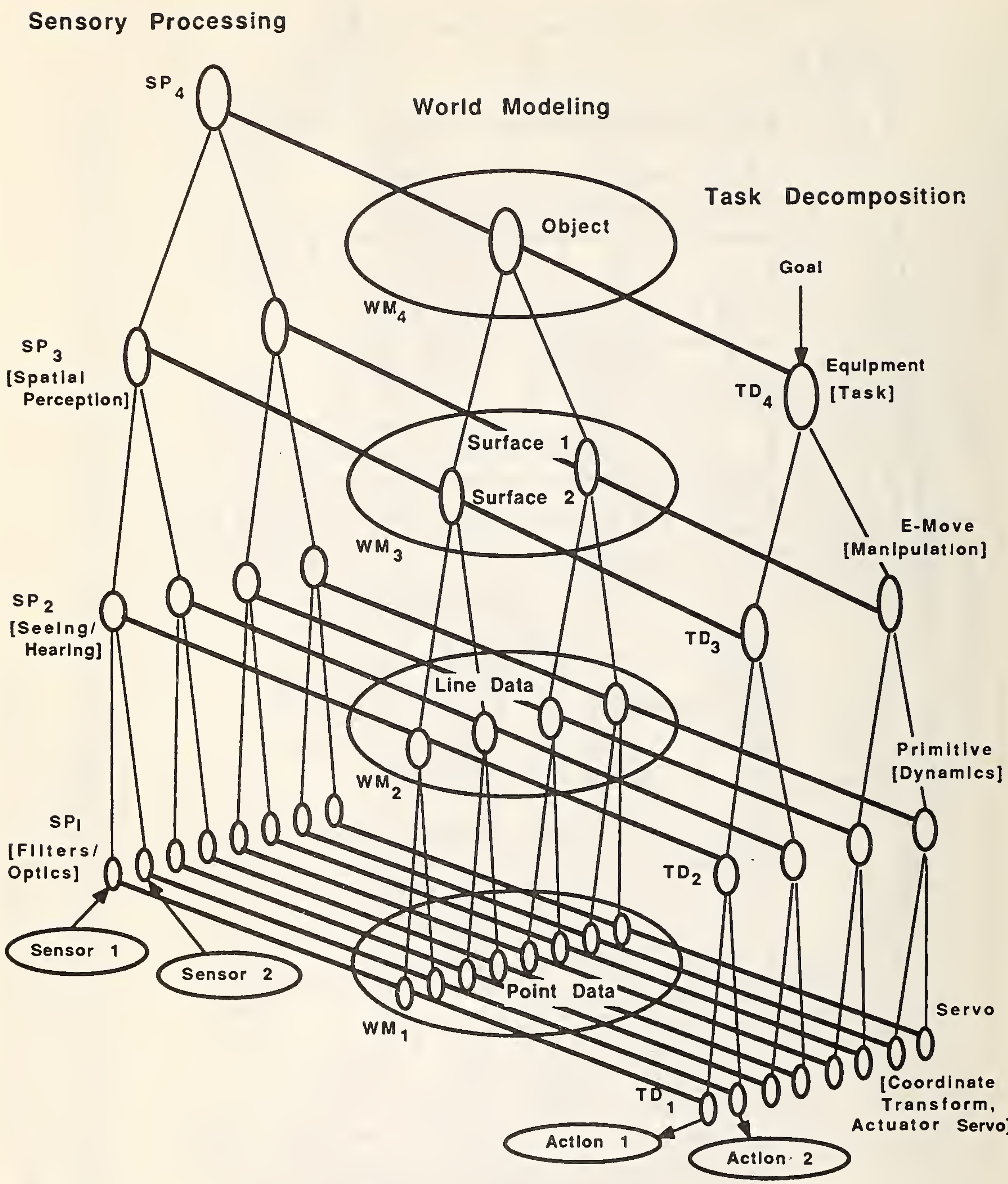


Figure 1.3: Hierarchical and Heterarchical (Horizontal) Organization of the Real-Time Control System Architecture

1.3 Hierarchical vs. Horizontal

The MASREM architecture has both hierarchical and horizontal communications.

The flow of commands and status feedback is hierarchical. High level commands, or goals, are decomposed both spatially and temporally through a hierarchy of control levels into strings and patterns of subcommands. Each task decomposition module represents a node in a command tree. It receives input commands from one and only one supervisor, and outputs subcommands to a set of subordinate modules at the next level down in the tree. Outputs from the bottom level consist of drive signals to motors and actuators.

The sharing of data is horizontal between modules at the same level. Both in terms of volume and bandwidth, there is much more information flowing horizontally between modules at the same level than flowing vertically between levels along the branches of the command tree.

1.3.1 Task Decomposition

Each task decomposition module at each level in the control hierarchy consists of a job assignment module, a set of planner modules, and executor modules. Each of these communicates voluminously with its respective world modeling module at the same level.

1.3.2 World Modeling

Each world modeling module is made up of a set of processes that maintain geometric models of the workspace, update lists of objects and their attributes, keep state variables current, generate predictions, and compute evaluation functions based on hypothesized or planned actions. Each world modeling module is constantly in communication with its respective sensory processing module.

1.3.3 Sensory Processing

The sensory processing modules are programmed to filter, detect, recognize, measure, and otherwise extract from the sensory data stream the information necessary to keep the world model at each level updated. It computes spatial and temporal correlations, differences, convolutions, and integrations; comparing predictions generated by the corresponding level modeling module with observations detected by lower level sensory processing modules.

1.3.4 Global Memory

The sharing of information between world model, task decomposition, and sensory processing modules at the same level is, however, not necessarily strictly horizontal. All input and output variables to all of the modules at all levels are globally defined. This facilitates interaction with the user, and facilitates debugging of the system. Conceptually, there is no logical restriction prohibiting any module at any level from making a query of, or obtaining information from, the world model at any level. These variables are available to any process that wishes to post a query or read a value. There, of course, may be practical limitations dictated by the physical implementation of the distributed computing hardware. For this reason the global memory, in general, will be distributed over a number of physically distinct memories. That is, it will be a distributed global memory.

1.3.5 Communications Processing

There exists a communications process which allows shared access to information in the distributed global memory. It is transparent to the computing modules and makes the distributed global memory appear to the various computing modules as if it were a single common memory.

1.3.6 Command Trees

Although the flow of commands through the hierarchical task decomposition command tree is strictly enforced (no mining machine and no command subtree will ever report to more than one supervisor at any instant in time), the command tree is not necessarily stationary. For example, a shuttle car at any given time would have but one supervisor; however, at the mining face it might have a different supervisor than when it was at the material handling area. At the production level and above, the command tree may be reorganized from time to time so as to reassign mining machines to different sections for various tasks. In the AMRF, this idea corresponds to the "virtual cell" which is described by McLean [Mc 82]. When the command tree is reconfigured, it is done instantaneously with the control structure always remaining a tree with one root node at the top where the longest term strategy is pursued and the highest level priority is determined.

2. A FUNCTIONAL SYSTEM ARCHITECTURE

The high-level block diagrams of the MASREM architecture are shown in figures 1.1, 1.2, and 1.3. The MASREM architecture is a three-legged hierarchy of computing modules, serviced by a communications system and a distributed global memory, and interfaced to operator workstations and other I/O devices.

The task decomposition (TD) modules perform real-time planning and task monitoring functions, and decompose task goals both spatially and temporally. The sensory processing (SP) modules filter, correlate, detect, and integrate sensory information over both space and time so as to recognize and measure patterns, features, objects, events, and relationships in the external world. The world modeling (WM) modules answer queries, make predictions, and determine values of evaluation functions on the state space defined by the information stored in global memory. Global memory is a database which contains the system's best estimate of the state of the external world. The world modeling modules keep the global memory database current and consistent.

2.1 Task Decomposition - TD modules (Plan, Execute)

The task decomposition hierarchy consists of TD modules which plan and execute the decomposition of high level goals into low level actions. Task decomposition involves both a temporal decomposition (into sequential actions along the time line) and a spatial decomposition (into concurrent actions by different subsystems). [Task Decomposition is explained in more detail in section 5.]

Each TD module at each level consists of three sublevels:

1) a job assignment manager JA,
2) a set of planners PL(i), and
3) a set of executors EX(i).

These three sublevels decompose the input task into both spatially and temporally distinct subtasks as shown in figure 2.1.

2.2 World Modeling - WM modules (Remember, Estimate, Predict, Evaluate)

Def. 1: World Model
The world model contains the system's best estimate and evaluation of the history, current state, and possible future states of the world, including the states of the system being controlled. The world model includes both the WM modules and a knowledge base stored in global memory where state variables, maps, lists of objects and events, and attributes of objects and events are maintained.

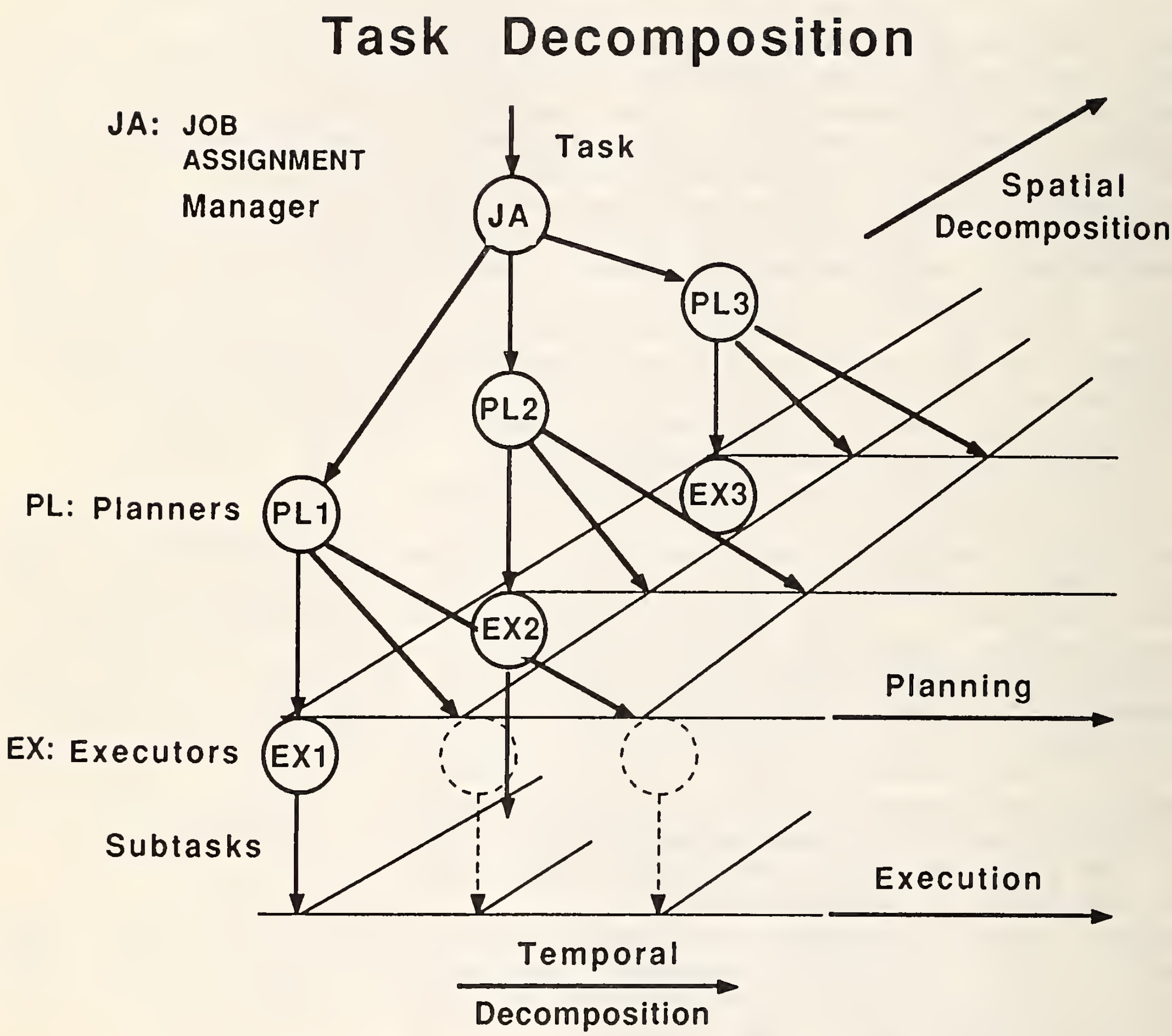


Figure 2.1: The Job Assignment Manager JA Performs a Spatial Decomposition of the Task. The Planners PL(j) and Executors EX(j) Perform a Temporal Decomposition

By this definition, the world model corresponds to what is widely known throughout the artificial intelligence community as a "blackboard" [Ba 81].

The world modeling leg of the hierarchy consists of WM modules which model (i.e., remember, estimate, predict) and evaluate the state of the world.

As shown in figure 2.2, the WM modules at various levels:

2.2.1 Maintain Knowledge Base

Maintains the global memory knowledge base keeping it current. The WM modules update the knowledge base based on correlations and differences between model predictions and sensory observations. This is shown in more detail in figure 2.3.

2.2.2 Provide Predictions

Provide predictions of expected sensory input to the corresponding SP modules based on the state of the task and estimates of the external world. This is shown in figure 2.3.

2.2.3 What Is?

Answer "What is?" questions asked by the planners and executors in the corresponding level TD modules. The task executor requests information about the state of the world and uses the answers to monitor and servo the task and/or to branch on conditions to subtasks that accomplish the task goal. This is shown in more detail in figure 2.4.

2.2.4 What If?

Answer "What if?" questions asked by the planners in the corresponding level TD modules. As shown in figure 2.5, the WM modules predict the results of hypothesized actions.

2.2.5 Evaluate Situations

Evaluate the current situation and potential future consequences of hypothesized actions by applying evaluation functions to current states and to future states expected to result from hypothesized actions. The evaluation functions define a set of values over the state-space defined by state variables in the global memory. These evaluation functions can be used to compute priorities, cost-benefit values, risk estimates, and pay-off values of states of the world. Thus, working together with the world model, the planners are able to search the space of possible futures and choose the sequence of planned actions that produce the best value of the evaluation functions. The executors are able to apply value judgments to moment by moment behavioral decisions.

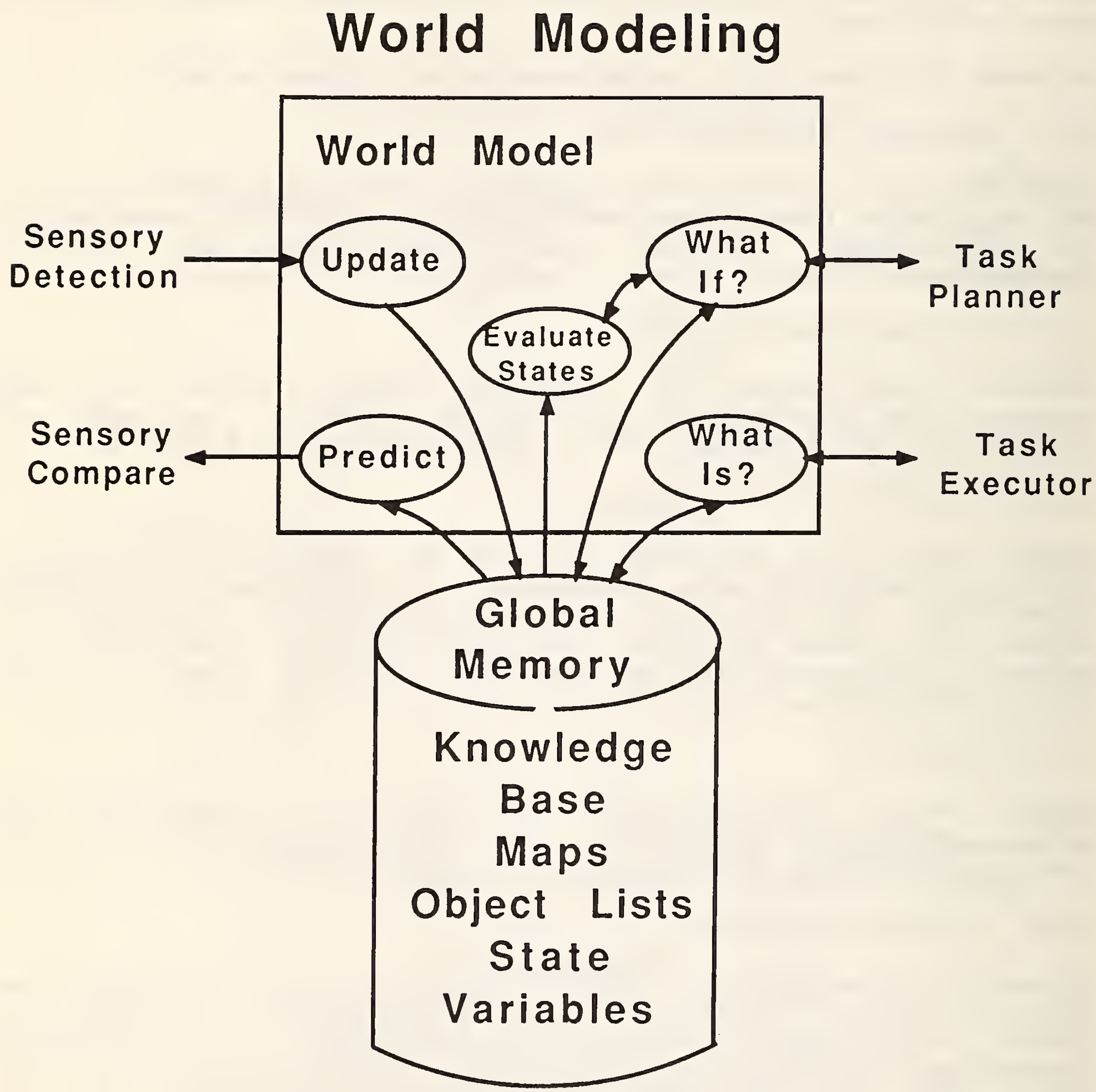


Figure 2.2: Functions Performed by WM Modules in the World Model

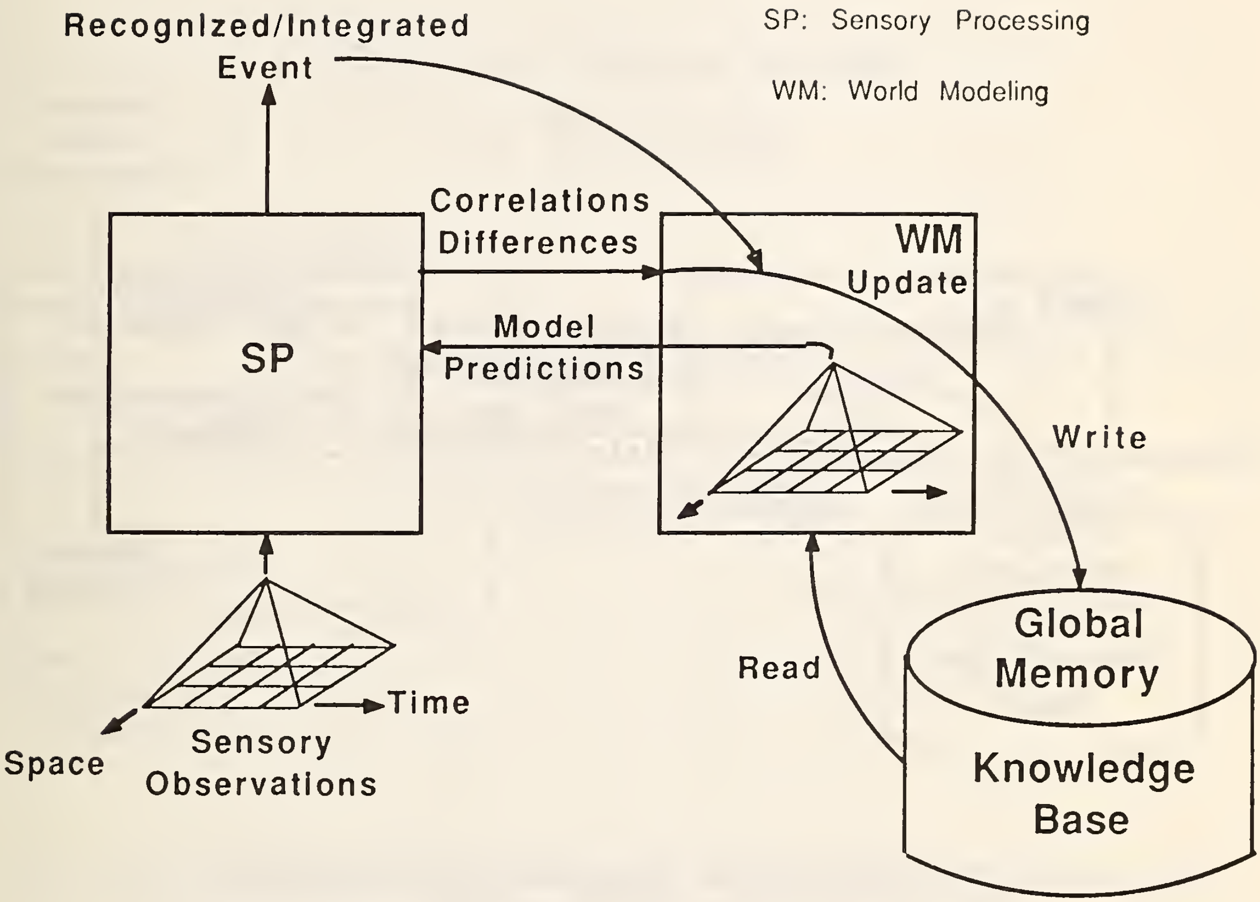


Figure 2.3: Role Of WM Module in Predicting Sensory Input and in Up-dating Knowledge Base Based on Correlations and Differences between Predictions and Observations

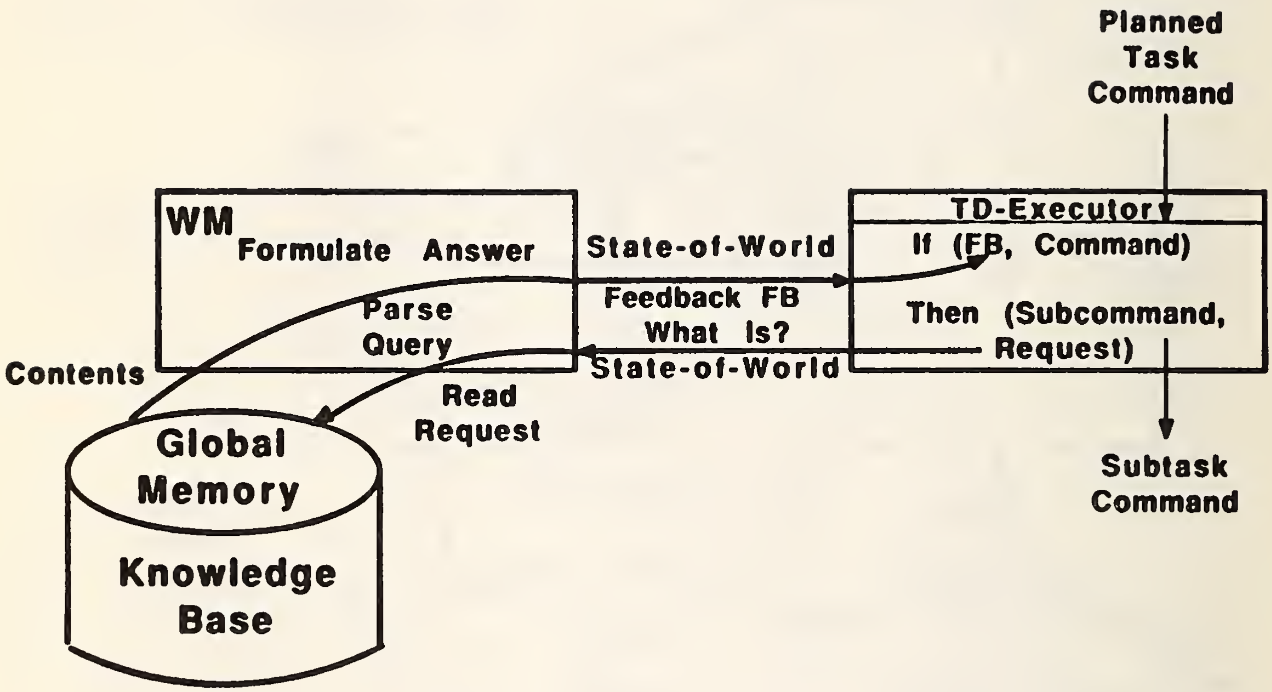


Figure 2.4: Role of WM Modules in Responding with Feedback FB to TD Module Executor "What Is?" Questions

Role of World Model in Planning

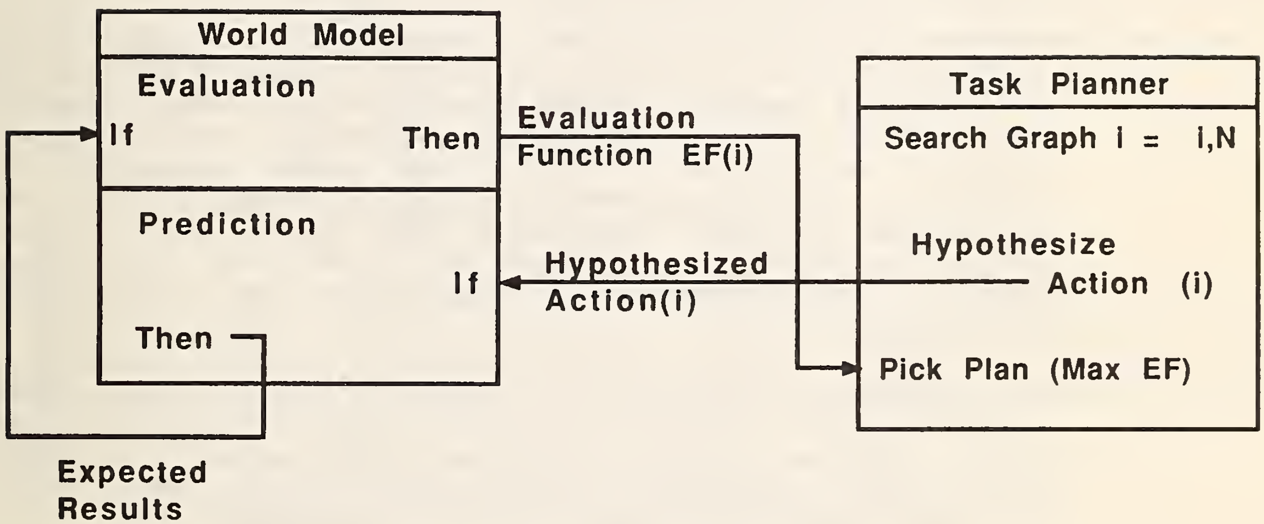


Figure 2.5: Role of World Model in Planning. Hypothesized Actions Are "What If?" Questions.

2.3 Global Memory

Def. 2:
Global memory is the database wherein is stored knowledge about the state of the world including the internal state of the control system.

2.3.1 Contents of Global Memory

The knowledge in the global memory consists of:

a) Maps which describe the spatial occupancy of the world. A map is a spatially indexed database showing the relative position of objects and regions. At different levels the maps have different resolution. The maps at different levels may be represented in a pyramid structure. Map overlays may also contain value functions such as utility, cost, risk, etc. to be used in path planning and safety.

b) Lists of objects, features, relationships, events, and frames containing their attributes. This database is indexed by name. Object and feature frames contain information such as position, velocity, orientation, shape, dimensions, mass, and other features of interest. Event frames contain information such as start and end time, duration, type, cost, payoff, etc. Recognized objects and events may also have confidence levels, and degrees of believability and dimensional certainty.

   At different hierarchical levels, object frames have different levels of detail and spatial resolution, and event frames have different levels of temporal resolution. Typically, class labels of the lower level are considered as primitives for the higher level.

c) State variables which identify particular situations. The state variables in global memory are the system's best estimate of the state of the world, including both the external environment and the internal state of the TD, WM, and SP modules. Data in global memory is available to all modules at all levels of the control system.

2.3.2 Implementation of Global Memory

Global memory is not necessarily implemented as a physically contiguous single block of memory. Global memory will normally, in practice, be distributed over a variety of media in physically disparate locations. Parts of global memory may be on dual-ported RAM located on TD, WM, or SP module processor boards on a multiprocessor bus. Other parts may be on disk or bubble memory at a variety of physical locations, on a mining machine, or at a control station for a section. What is important is that the variables in global memory are globally defined, that they can be called symbolically by the computational modules that either read or write them, and that they can be accessed with acceptable delay. It is recognized that each level of the hierarchy has substantially different requirements for delay because each runs roughly one order of magnitude slower than the level below. This must be taken into consideration during the implementation of global memory so that information can flow as required. There should be an on-line data dictionary so that numerical values can be bound to symbolic names during execution.

Of course, it is important that there exist automatic protection mechanisms to prevent array variables from being simultaneously read and written, and to prevent corruption of data by program bugs such as indexing outside of arrays, etc. When global memory is distributed such that multiple copies of data exist, then there must be mechanisms for assuring consistency between multiple copies. Implementation must also take into account the timing requirement of the computational modules that make use of the data for real-time control. Typically, every global variable has only one process that writes it, while many processes may read it. This greatly simplifies the problem of preventing inadvertent corruption of global memory.

2.4 Sensory Processing - SP modules
(Filter, Integrate, Detect, Measure)

The sensory processing leg of the hierarchy consists of SP modules which recognize patterns, detect events, and filter and integrate sensory information over space and time. As shown in figure 2.6, the SP modules also consist of three sublevels which:

1) compare observations with predictions
2) integrate, correlate and difference over time
3) integrate, correlate and difference over space

These spatial and temporal integrations fuse sensory information from multiple sources over extended time intervals. Newly detected or recognized events, objects, and relationships are entered by the WM modules into the world model knowledge base in global memory, and objects or relationships perceived to no longer exist are removed. The SP modules also contain functions which can compute confidence factors and probabilities of recognized events, and statistical estimates of stochastic state variable values.

2.5 Operator and Programmer Interfaces
(Control, Observe, Define Goals, Indicate Objects, Edit Programs and Data)

The control architecture defined here has operator and programmer interfaces at each level in the hierarchy.

Sensory Processing

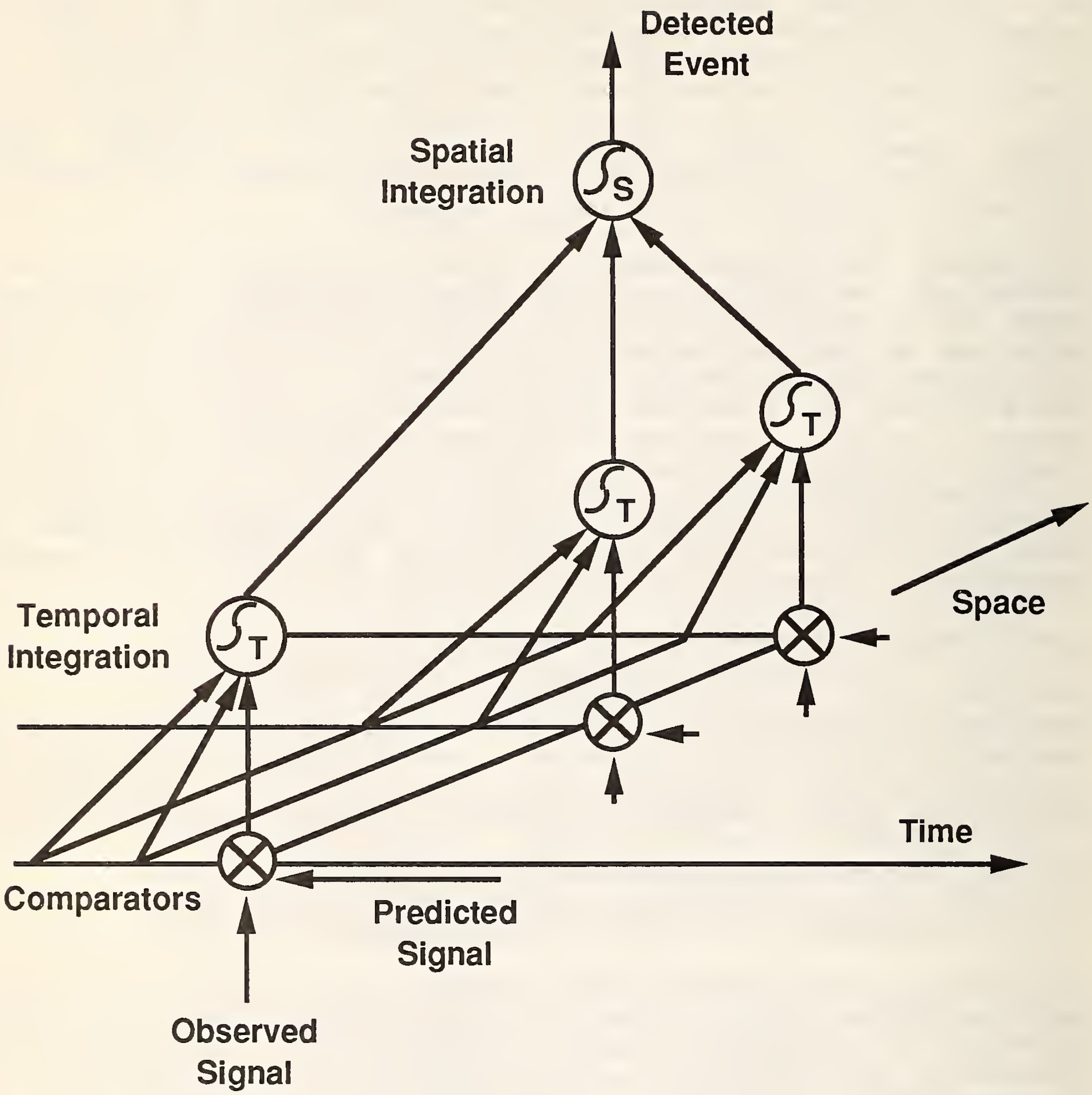


Figure 2.6: Each Sensory Processing SP Module Performs a Comparison and both Temproal and Spatial Integration

2.5.1 Operator Interface

The operator interface provides a means by which human operators, at the control station can observe, supervise, and directly control the mining equipment. Each level of the task decomposition hierarchy provides an interface where the human operator can assume control. The task commands into any level can be derived either from the higher level TD module, or from the operator interface or some combination of the two. Using a variety of input devices such as a joystick, mouse, trackball, light pen, keyboard, voice input, etc., a human operator can enter the control hierarchy at any level, at any time of his/her choosing (within restrictions imposed by synchronization and data integrity constraints), to monitor a process, to insert information, to interrupt automatic operation and take control of the task being performed, or to apply human intelligence to sensory processing or world modeling functions. (Operator interrupts will not literally be allowed at "anytime," but at frequent points in time where operator interrupts can be synchronized to coincide with state clock increments or subtask completion events.)

The operator interface terminal provides the input devices (joystick, mouse, trackball, light pen, keyboard, or voice input) whereby the human operator can input the information needed for designating tasks at that level. The terminal may also provide output devices (alphanumeric and graphic CRT's, printers, warning lights, warning sounds, etc.). These output devices provide feedback to the operator and indicate the state of the equipment and the result of the operator's intervention. The operator interface processor provides the necessary translators and string generators to format human inputs and system outputs into the proper format, and to verify, validate, and synchronize them with ongoing processes at the appropriate level or levels. The operator interface processor also provides the synchronization mechanisms necessary so that automatic operations can be resumed from the point in time and space where the human operator leaves off, or restart automatic operations from the point where the human interrupted.

The sharing of command input between human and autonomous control need not be all or none. The combination of automatic and operator modes can span an entire spectrum from one extreme, where the operator takes complete control of the system from a given level down so that the levels above the operator are disabled, to the autonomous mode where the operator loads a given program and puts the mining machine on automatic. In between these two extremes is a broad range of interactive modes where the operator supplies some control variables and the autonomous system provides others. For example, a human might control the orientation of a camera while a machine automatically translates the same camera through space. It is also within the state of the art to compute control inputs by a function which multiplies human and automatic input variables by relative percentages and sums the result so that both human and autonomous inputs share in influencing the position, velocity, and force. Even in cases where the operator takes complete control, some of the higher level safety and fault protection functions should remain in operation.

2.5.2 Operator Control Interface Levels

If the human operator enters the task decomposition hierarchy in the middle of level 1 (at the input to the servos or actuators), he/she must use a replica master, or individual joint position, rate, or force controllers.

If the human enters the task decomposition hierarchy above level 1 (input to servo/actuator level), he/she can use a joy stick to perform resolved motion force/rate control or he/she can use function buttons to actuate or deactivate subsystems or movements.

If the human enters above level 2 (input to Primitive Level), he/she can simply indicate safe motion pathways, and the mining control system will compute dynamically efficient incremental movements.

If the human enters above level 3 (input to e-move Level), he/she can graphically or symbolically define key positions, or using a menu, call for elementary cutting head or machine transport movements (e-moves) such as <navigate(A,B,C, ...)>, <sump(angle, depth_sump)>, etc. This may be done using an interactive graphics display with a joystick, mouse, track ball, light pen, or voice input.

If the human enters above level 4 (input to Equipment Level), he/she can indicate objects and call for tasks to be done on those objects, such as <continuous miner align to the face>, <continuous miner straight cut>, <shuttle car unload>, etc. This may be done using cursors and graphic images overlaid on television images.

If the human enters above level 5 (input to Section Level), he/she can reassign mining machines to different mine sections, insert, monitor, or modify plans that describe equipment task sequences, define coal preparation, etc.

If the human enters above level 6 (input to Production Level), he/she can reconfigure all mining priorities, change mining requirements, enter or delete jobs, and change the mining operations schedule.

The operator control interface thus provides mechanisms for entering new instructions into the various control modules or program selection or execution sequences. This can be used on-line for real-time supervisory control, or in a background mode for altering autonomous mining plans before autonomous execution reaches that part of the plan. The operator control interface can also provide look-ahead simulation of planned moves so as to analyze the consequences of a prospective motion command before it is executed.

2.5.3 Operator Monitoring Interfaces

The operator interfaces allow the human the option of simply monitoring any level. Windows into the global memory knowledge base permit viewing of maps of a section, geometric descriptions and mechanical and electrical configurations of mining machines, lists of recognized objects and events, object parameters, and state variables such as positions, velocities, forces, confidence levels, tolerances, traces of past history, plans for future actions, and current priorities and utility function values. These may be displayed in graphical form; for example, using dials or bar graphs for scalar variables, shaded graphics for object geometry, and a variety of map displays for spatial occupancy. Time traces can be represented as time line graphs, or as stick figures with multiple exposure and time decay. State graphs with windows into nodes and edges can be used to display the state of the various modules in the control system and the conditions required for state transitions.

Sequences of past actions or plans for future action can be represented as state graphs, with windows into nodes to display the state of the various modules in the control system at different times, and windows into edges to display the conditions required for state transitions. Geography and spatial occupancy can be displayed as a variety of maps, vectors, or stick figures, or shaded graphics images. Object geometry can be represented as wire frames or 3-dimensional solid objects. The operator may also have a direct television image of the mining machine's environment with graphics overlays which display the degree of correlation between what the mining machine believes is the state of the world and what the human operator can observe with his/her own eyes.

2.5.4 Sensory Processing/World Modeling

The operator interface may also permit interaction with the sensory processing and/or world modeling modules. For example, an operator using a video monitor with a graphics overlay and a light pen or joystick might provide human interpretative assistance to the vision/world modeling system. The operator might interactively assist the model matching algorithms by indicating with a light pen which features in the image (e.g., edges, corners) correspond to those in a stored model. Alternatively, an operator could use a joystick to line up a wireframe model with a TV image, either in 2-D or 3-D. The operator might either move the wireframe model to line up with the image, or move the camera position to line up the image with the model. Once the alignment was nearly correct, the operator could allow automatic matching algorithms to complete the match and track future movements of the image.

The human operator can thus monitor, assist, and if desired, interrupt autonomous operation at any time (within the restrictions noted above), for any reason, at any desired level, to take control, to stop the mining machine, to slow it down, to back it up, or to substitute the human's judgment by directly entering commands or other information to replace what the robot had otherwise planned to do.

2.5.5 Programmer Interface

The programmer interface allows a human programmer to load programs, monitor system variables, edit commands and data, and perform a broad range of debugging, test, and program modification operations.

There are a variety of levels of programmer interface corresponding to various levels of programming skill and system change authority. The lowest level allows only monitoring of system variables. The next higher level permits debugging tests to be performed while applications programs are running. The third level allows on-line editing of data variables and applications program code. The fourth and highest level allows editing of the mining machine operating system itself.

Both the operator and programmer interface formats could be defined in ASCII strings, so that information flowing in either direction can be easily read by either man or machine. This convention greatly facilitates debugging and system integration. However, other information formats which follow an object oriented approach are also applicable.

2.6 Safety System

The mining machine control system should incorporate a safety system which can prevent the mining machine system from entering forbidden volumes, both in physical space and in state space. This safety system should always be operational so as to prevent damage to the mining machine or surrounding structures or humans during all modes of operation: teleoperation, autonomous, and shared.

The safety system should have access to all the information contained in the world model of the control system, but should also maintain its own world model, updating it with redundant sensors. The safety system should periodically query the control system to test its state and responsiveness. Conversely, the control system should also periodically query the safety system to test it. Observed states should be constantly compared with predicted states and differences noted. If either system detects an anomaly in the other, error messages should be sent and appropriate action taken.

3. LEVELS IN THE CONTROL HIERARCHY

The MASREM architecture described here for the Mining Automation system is a seven-level hierarchy, as shown in figure 3.1. At each level in this hierarchy a fundamental operation is performed.

Level 1 The SERVO/ACTUATOR LEVEL typically transforms coordinates from a machine or tool coordinate frame into actuator coordinate frames. This level servos positions, velocities, accelerations, and forces. This level also converts state variables into electronic signals to command actuators.

Level 2 The PRIMITIVE LEVEL decomposes strings of intermediate position commands (primitive commands) and computes inertial dynamics, generates smooth trajectories, and servos various subsystems in convenient position parameters. Paths defined by the input to level 2 (primitive commands) are assumed to be obstacle free.

Level 3 The ELEMENTARY-MOVE LEVEL decomposes elementary move commands (e-moves) into strings of intermediate position commands (or trajectory knot points). The e-move level deals with the local world. The local world is the area surounding the equipment which is within the range and resolution of its sensors. The e-move level selects obstacle free paths based on real-time sensor data. e-moves are typically defined in terms of motion of the subsystem being controlled (i.e., cutter mechanism, conveyer, camera platform, etc.) through a space defined by a convenient coordinate system. e-move commands may consist of symbolic names of elementary movements, or may be expressed as keyframe descriptions of desired relationships to be achieved between system state variables. e-moves are decomposed into strings of intermediate position commands (primitive commands) which define motion pathways that have been checked for clearance with potential obstacles and which avoid limits and kinematic singularities.

Level 4 The EQUIPMENT LEVEL decomposes equipment task commands specified in terms of actions performed on machines into sequences of e-moves defined in terms of machine motions. Equipment tasks typically define actions to be performed by a single mining machine system on one object at a time. Tasks defined in terms of actions on objects are decomposed into sequences of e-moves defined in terms of manipulator or vehicle subsystem motions. This decomposition checks to assure that there exist motion freeways clear of known obstacles (that exist in the world model and may be beyond sensor range) between positions given by the e-move commands. It also schedules coordinated activities of mining machine subsystems, such as the conveyer, gathering head, and cutting mechanism. (Coordination at this level consists of scheduling the start and end of e-moves, not of instant-by-instant real-time synchronization of movements. This type of tight movement synchronization is accomplished by sharing of system state variables through global memory at levels 1 through 3.)

Level 5 The SECTION LEVEL decomposes section plans for a mine. The level 5 job assignment manager decomposes section tasks into equipment and vehicle job assignments. This decomposition typically assigns to each piece of mining equipment a prioritized list of tasks to be performed in coordination with another piece of equipment and according to a schedule. Efficient equipment assignments are selected to maximize the effectiveness of the group's activity. Level 5 equipment planners schedule group task lists into coordinated sequences of equipment and vehicle tasks. The planners use the section level world model map to compute equipment trajectories and transit times. They also estimate costs, risks, and benefits of various equipment task sequences.

Level 6 The PRODUCTION LEVEL decomposes the production plan into section commands. Production plans typically specify distribution of mining equipment and support equipment needed by various sections. The level 6 decomposition typically dispatches commands to material handling, mining equipment, etc., and schedules the activities to maximize the plan. To a great extent level 6 section plans could be generated off line, either by human planners, or by automatic or semiautomatic planning methods.

Level 7 The FACILITY LEVEL accepts orders for coal from customers and plans allocation and scheduling of production facilities to meet demand. It provides estimates of delivery dates, cost, etc., to meet demand. Each order becomes a job which is decomposed into production plans and schedules. These in turn become commands to each production area. The order is typically specified by a list of objectives, priorities, requirements, and time line constraints.

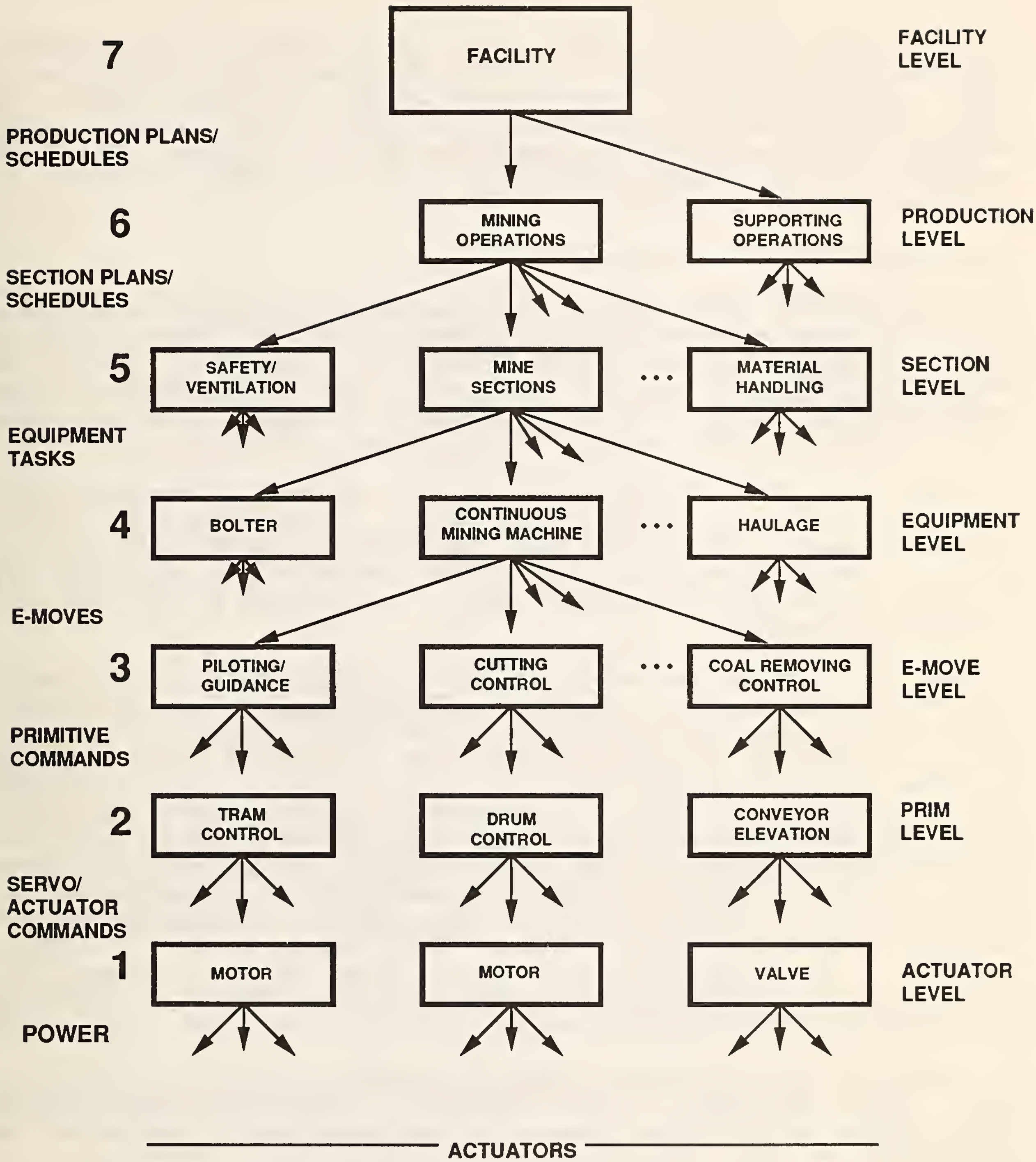


Figure 3.1: Proposed Seven Level Hierarchical Control System

4. COMMUNICATIONS

The TD, WM, and SP modules at all levels of the MASREM architecture can be viewed as state machines which periodically read input variables, compute some function of their input and state, write output variables, and go to a new state. This requires a communications mechanism by which output variables computed by the various modules at time t=i become available as input variables at time t=i+1.

4.1 Communications Timing

One possible implementation of the MASREM architecture would be a discrete time system in which a state clock is incremented at short intervals. The actual state clock interval should be selected based on application specific minimum response time requirements. Between each state clock increment, there would exist a data transfer/compute cycle as shown in figure 4.1.

As soon as the state clock is incremented, communication processes move data to the global memory from all output buffers that are ready, and thereafter, to those input buffers that are ready. The routing of data by these communication processes may be controlled by the request data in the output buffer of the computation module.

During the compute period, all state variables in global memory are effectively frozen, and represent a snapshot in time of the state of the world at the time of the state clock transition. The TD, WM, and SP functions can read from their input buffers, or from global memory, and compute functions on both local and global variables. Output is stored in output buffers until the next increment of the state clock. Any process that does not finish computing by the end of the compute period will continue until it does finish. Its output buffers will not be moved by the communication processes until the process is finished and the output buffers contain new data. Each output buffer therefore carries a ready flag which is set to busy when the computation process begins and is set to ready when the computation process is finished and the output buffers contain fresh data.

A variety of mechanisms for message passing through global memory have been studied. For example, computing modules may communicate with global memory by defining local mailboxes as described in [Mc 83, Mi 84]. Mailgrams are posted in the mailboxes or read from the mailboxes by the local processes. Delivery of the mailgrams is accomplished by a data administration system which periodically picks up messages from mailboxes that have new information, and deposits the messages at their specified destinations.

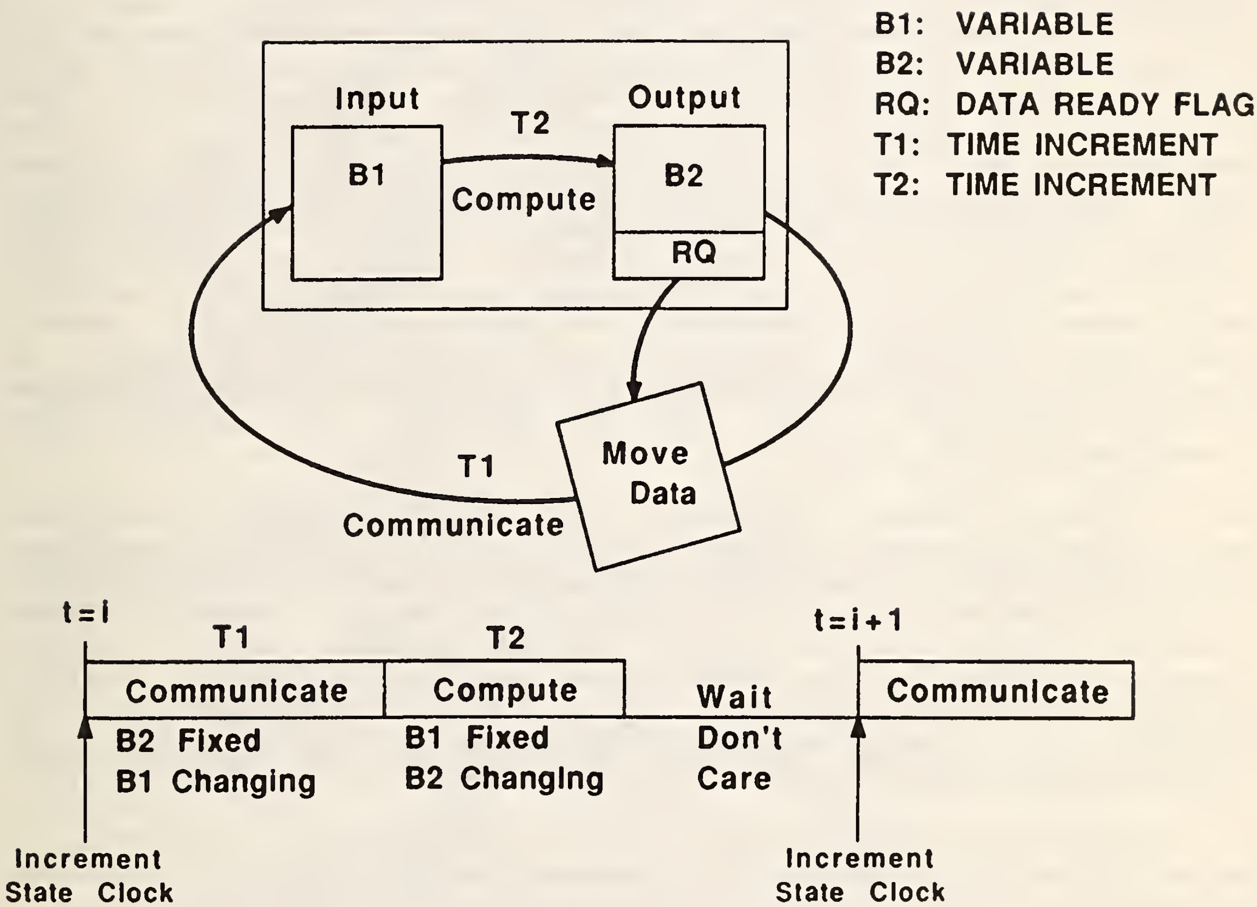


Figure 4.1: Executor Timing for Communicate-Compute-Wait Cycle

Timing requirements for process synchronization vary at different levels of the hierarchy. At level one, synchronization within a few milliseconds is typically important. At level two, synchronization within tens of milliseconds is usually adequate. At level three, synchronization within tenths of a second; level four, within seconds; level five within tens of seconds; at level six, synchronization within minutes is sufficient; and level seven synchronization within tens of minutes may suffice. This is illustrated by the example timing diagram of figure 4.2.

4.2 Communications Through Global Memory

Although there are many methods for implementing a real-time multiprocess communications system, it is conceptually useful to think of passing variables through a global memory. Assume for example, that the MASREM control hierarchy is supported by a global memory in which all state variables, including all input and output variables, are globally defined. It is highly desirable that the global memory have a large direct address space; for example, many applications typically require a minimum of a 32-bit, 4-gigabyte address space. Then communication can consist simply of each computing module reading its inputs from global memory and writing its output back into global memory. Each computing module needs only to know where in global memory its input variables are stored, and where in global memory it should write its output variables. The read and write functions in the system SP, WM, and TD modules then define the communication interfaces.

The global memory approach not only readily supports interprocessor communications, it also provides a clean interface for the operator/programmer workstation. The operator displays the variables needed from the locations in global memory. If the operator wishes to take control of the system, he/she writes command variables to the appropriate locations in global memory. The control modules that read from those locations need not know whether their input commands derived from a human operator, or from the next higher level in the autonomous control hierarchy.

If an operator wishes to monitor or modify a data variable, a sensor input, or a drive signal output, he/she can simply execute the equivalent of a "PEEK" or "POKE" into the global memory.

The global memory also supports modular development of software. Any system module can be replaced with a functionally equivalent module by merely respecting the address definitions for the input and output data.

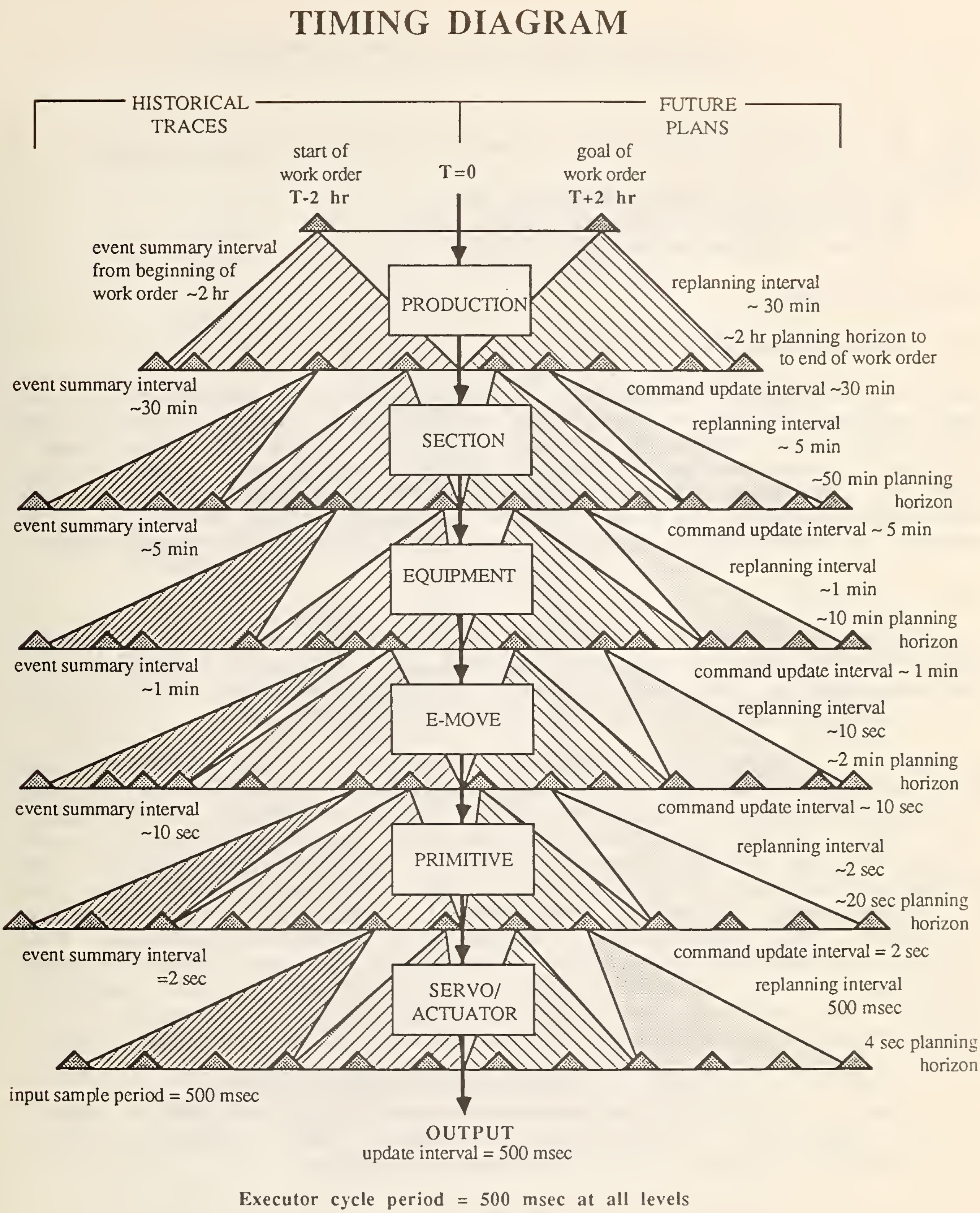


Figure 4.2: A Timing Diagram for the Mining Control System Illustrating the Planning and Sensory Processing Time Scales at Each Level.

5. DETAILED STRUCTURE OF THE TD MODULES

The TD module at each level consists of three parts as shown in figure 5.1:

1) a job assignment manager JA,
2) one or more planners PL(s), and
3) one or more executors EX(s).

For each level:

5.1 Job Assignment

The job assignment manager JA is responsible for partitioning the task command TC into s spatially or logically distinct jobs to be performed by s physically distinct planner/executor mechanisms. At the upper levels the job assignment module may also assign physical resources against task elements. The output of the job assignment manager is a set of job commands JC(s), s=1, 2, ..., N where N is the number of spatially, or logically, distinct jobs.

5.2 Planners

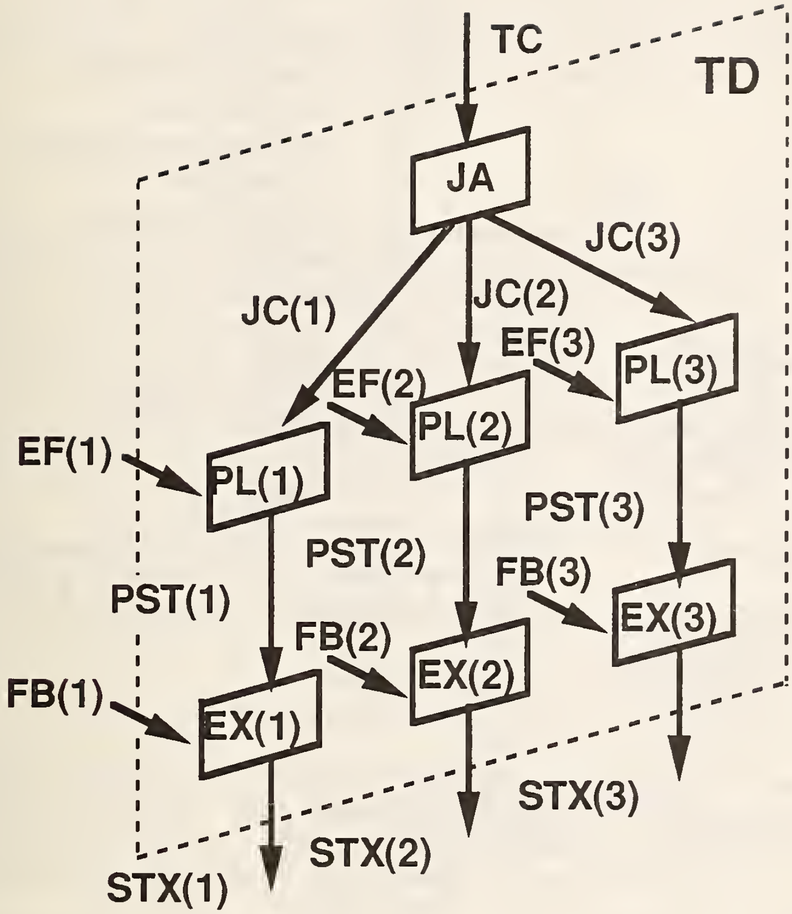
For each of these job commands JC(s), there exists a planner PL(s) and a executor EX(s). Each planner PL(s) is responsible for decomposing its job command JC(s) into a temporal sequence of planned subtasks PST(s,tt) as shown in figure 5.2.

Planning typically requires evaluation of alternative hypothetical sequences of planned subtasks. As shown in figure 2.5 the planner hypothesizes some action or series of actions, the world model predicts the results of the action(s) and determines the value of some evaluation function EF(s,tt) on the predicted resulting state of the world. This evaluation function is sometimes called a cost-benefit analysis or objective function. The hypothetical sequence of actions producing the evaluation function EF(s,tt)max that gives the best value is then selected as the plan PST(s,tt) to be executed by the executor EX(s). We may express the plan PST(s,tt) as the result of a function PL(s) operating on the parameters JC(s) and EF(s,tt)max, i.e.

$$PST(s,tt) = PL(s)\ [JC(s), EF(s,tt)max]$$

where tt is the time sequence index for steps in the plan. tt may also be defined as a dummy time variable, or a running temporal index in planning space.

$$tt = 1, 2, ..., th$$



TC: Task Command
TD: Task Decomposition
JA: Job Assigment
JC: Job Command
PL: Planner
PST: Planned Subtask
FB: Feedback
EF: Evaluation Function
STX: Executor Output
EX: Executor

Figure 5.1: The TD Module at Each Level Has Three Parts: A Job Assignment Module JA, Planners PL, and A Set of Executors, EX.

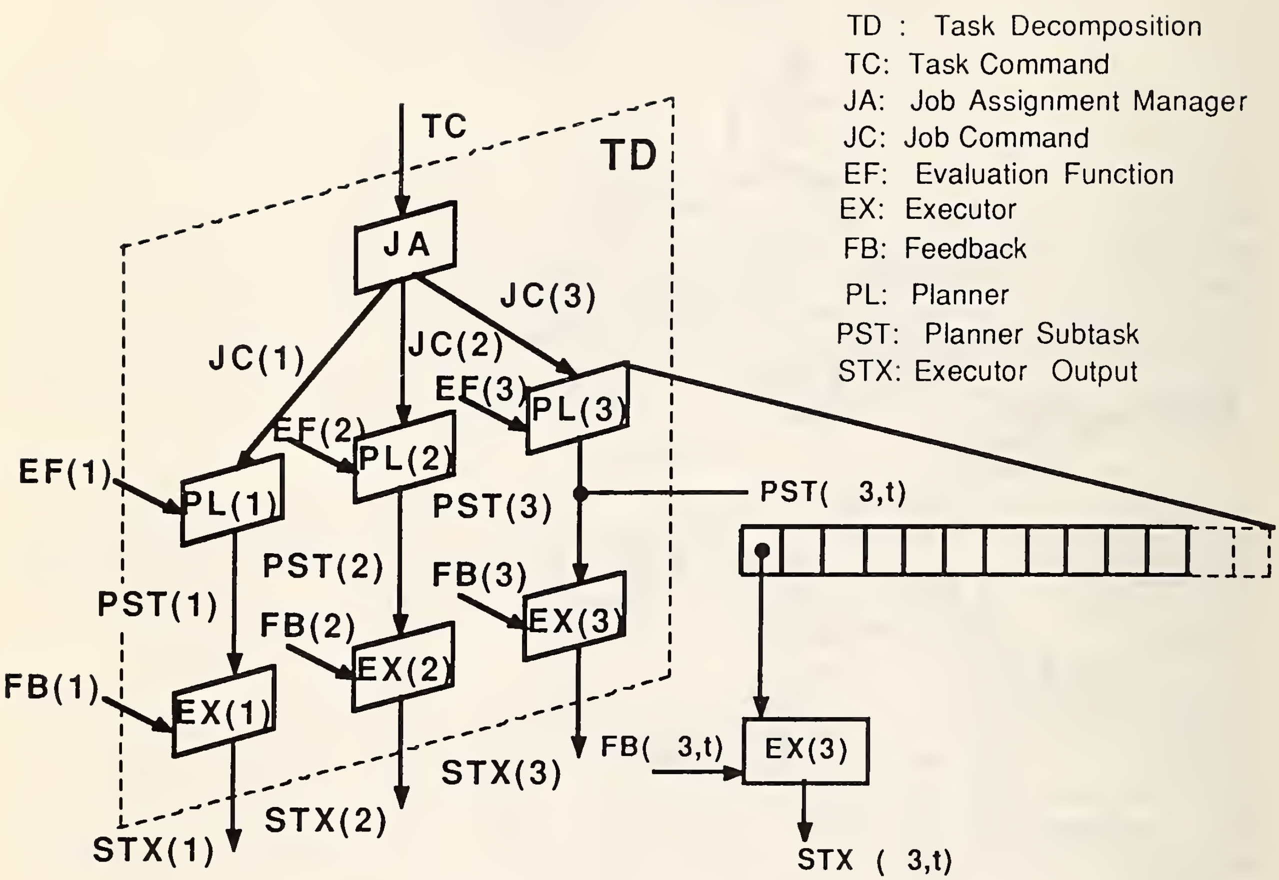


Figure 5.2: Each Planner PL(J) Produces a String of Planned Subtasks PST (J, T). At Time T The Executor EX (J) Reads the Planned Subtask PST (J, T), The Feedback FB (J, T), and Computes an Output STX (J, T).

where th is the value of the tt index at the planning horizon. The planning horizon is defined as the period into the future over which a plan is prepared. Each level of the hierarchy has a planning horizon of one or two expected input task time durations. The replanning interval should be one order of magnitude less than the planning horizon (or about equal to the expected output subtask time duration). Thus the planning horizon grows exponentially at each successively higher level of the hierarchy as illustrated in figure 5.3.

5.3 Executors

Each executor EX(s) is responsible for successfully executing the plan PST(s,tt) prepared by its respective planner PL(s). If all the subtasks in the plan PST(s,tt) are successfully executed, then the goal of the original task will be achieved. The executor operates by selecting a subtask from the current queue of planned subtasks and outputting a subcommand STX(s,t) to the appropriate subordinate TD module at time t. The EX(s) module monitors its feedback FB(s,t) input in order to servo its output STX(s,t) to the desired subtask activity. The executor output may then be expressed as the function EX(s) operating on the parameters PST(s,t) and FB(s,t), i.e.,

$$STX(s,t+n) = EX(s)\ [PST(s,t),FB(s,t)]$$

where n = the number of state clock periods required to compute the function EX(s). n typically equals 1.

The feedback FB(s,t) also carries timing and subgoal event information for coordination of output between executors at the same level. When the executor detects a subgoal event, it selects the next planned subtask from the queue.

Executor output STX(s,t) also contains requests for information from the world model WM module, and status reports to the next higher (i+1) level in the TD module hierarchy. The feedback FB(s,t) contains status reports from the TD module at the i-1 th level indicating progress on its current task. As a minimum, these reports provide a handshaking acknowledgment of receipt of the subtask command and an echo of the unique identification number of the command currently being executed. This enables the EX(s) process to know that the subtask output given has been received and is being executed. The EX(s) process generates error reports if time-outs or failures in handshaking with the TD module at the i-1 th level occur.

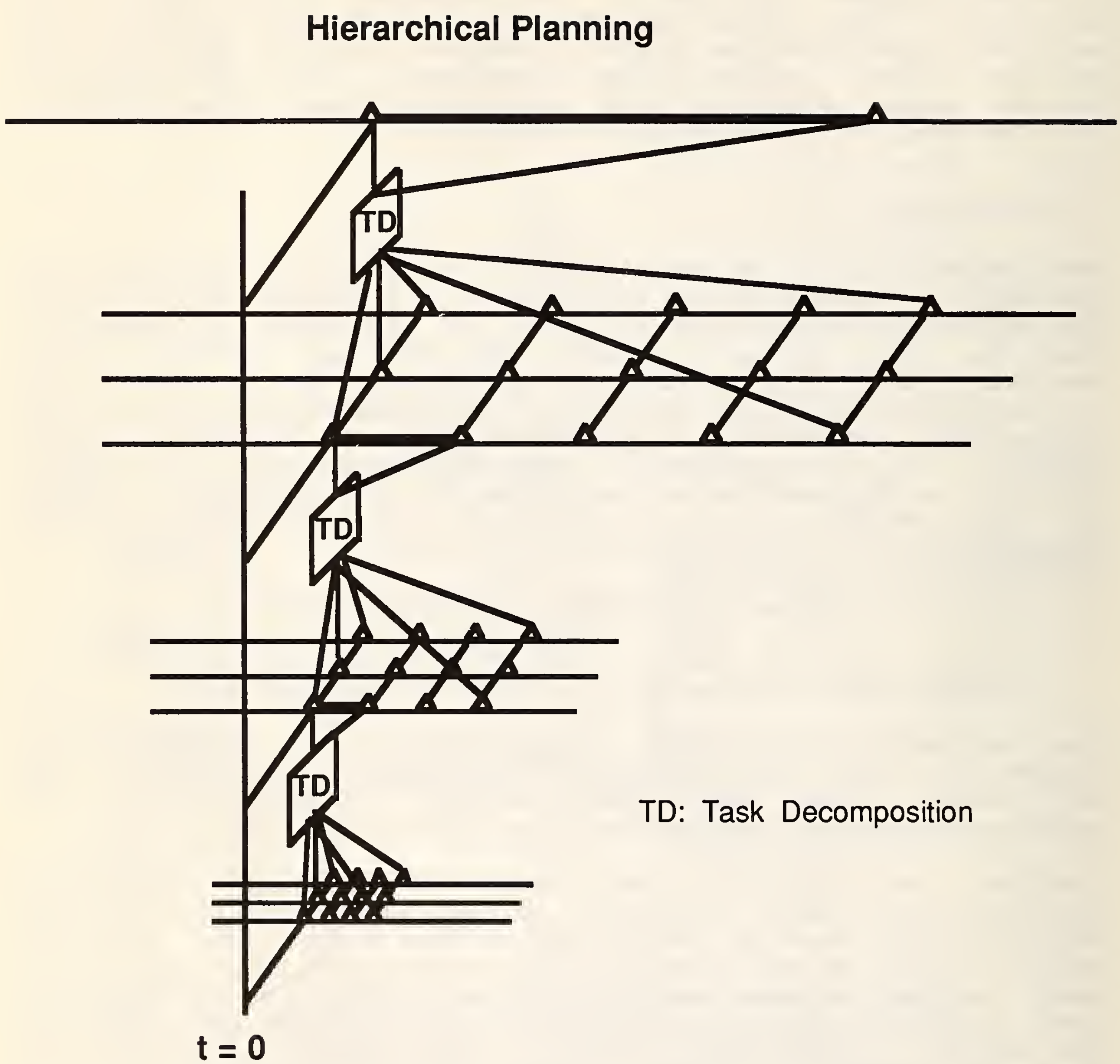


Figure 5.3: Three Levels of Real-Time Planning Illustrating the Shrinking Planning Horizon and Greater Detail at Successively Lower Levels of the Hierarchy.

6. TASKS AND PLANS

6.1 Task

Def 3:
As shown in figure 6.1, a task is an activity which begins with a start-event and is directed toward a goal. A goal is an event which terminates the task. A task command is an instruction to achieve a goal event of the form:

DO <Task> AFTER <Start Event> UNTIL <Goal Event>

6.2 Plan

Def 4:

A plan is a set of activity-event pairs which lead to the desired goal event. Each activity in the set leading to the goal is a subtask, and the event terminating each of the subtasks is a subgoal. The final event in the plan is the goal event. This is illustrated in figure 6.2.

A plan may involve the scheduling of several machines to simultaneously perform different activities on different objects as illustrated in figure 6.3. These subtasks may depend on each other for results, for example, when an operation on an object in one machine cannot begin before another machine finishes its operation on that same object.

Complex plans may involve conditional branching, or even probabilistic decision rules. Plans may also include provisions for branching to error correction activities and reporting failure in the case of lack of progress toward the goal.

In some cases, plans can be represented by mathematical functions of time and/or state variables such as distance from target, velocity, coordinate position, etc. For example, a path planner for the mining machine may compute a straight-line trajectory from the current point to a goal point, or as illustrated in figure 6.4, the planning function may compute acceleration and deceleration profiles as a function of time or position along the planned trajectory.

A plan can be represented in a number of different notations. The series of actions and events illustrated in figure 6.3 is in the form of a Gantt chart. Plans can also be represented as a graph of states and state-transitions in the form of a Pert or Critical Path Method (CPM) chart, as a Petri network, or any of several other methods for representing trajectories through state space, such as state-graphs or finite-state-automata (fsa) grammars.

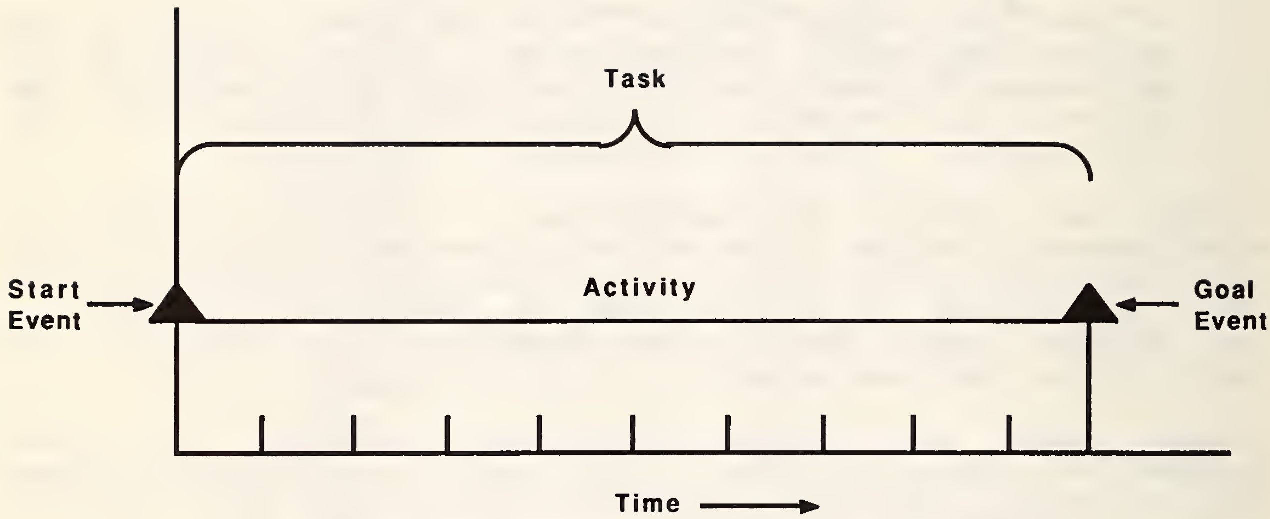


Figure 6.1: A Task Is an Activity Directed toward a Goal.

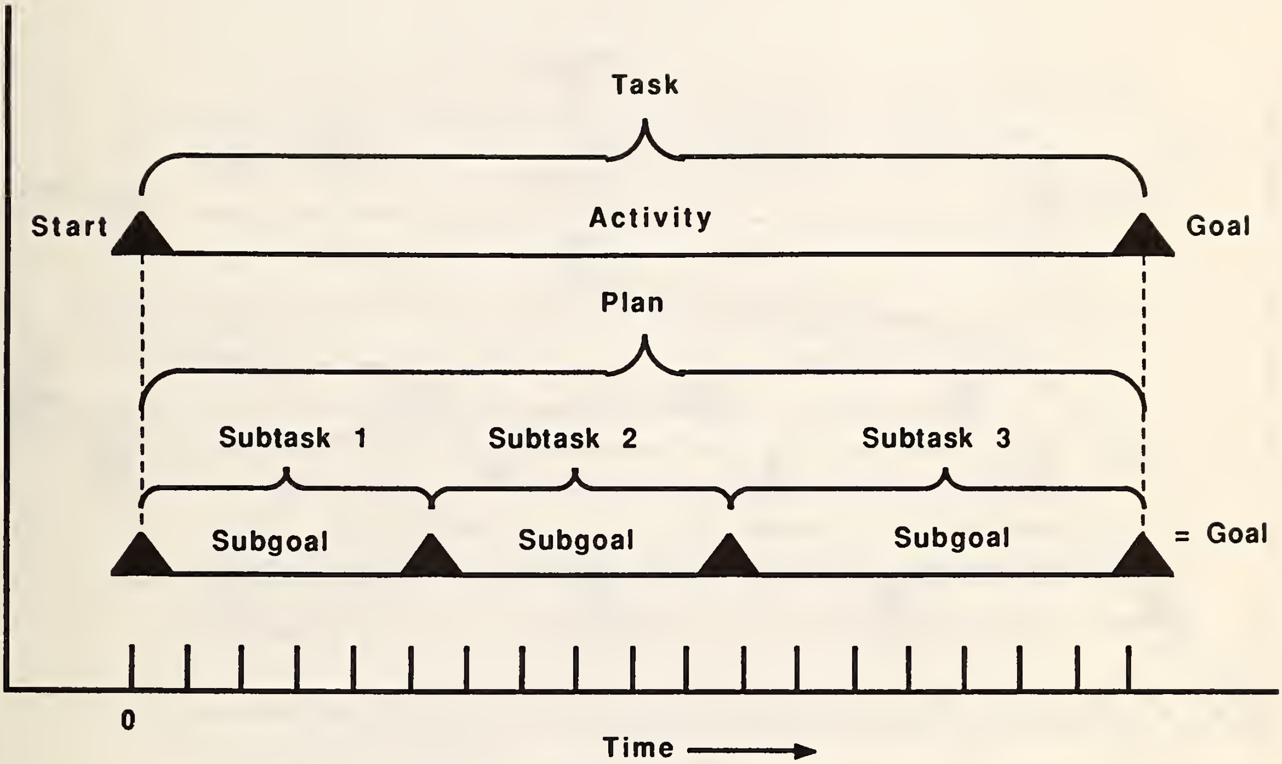


Figure 6.2: A Plan Is a Set of Activity-Event Pairs, or Subtasks Which Achieve the Goal Event.

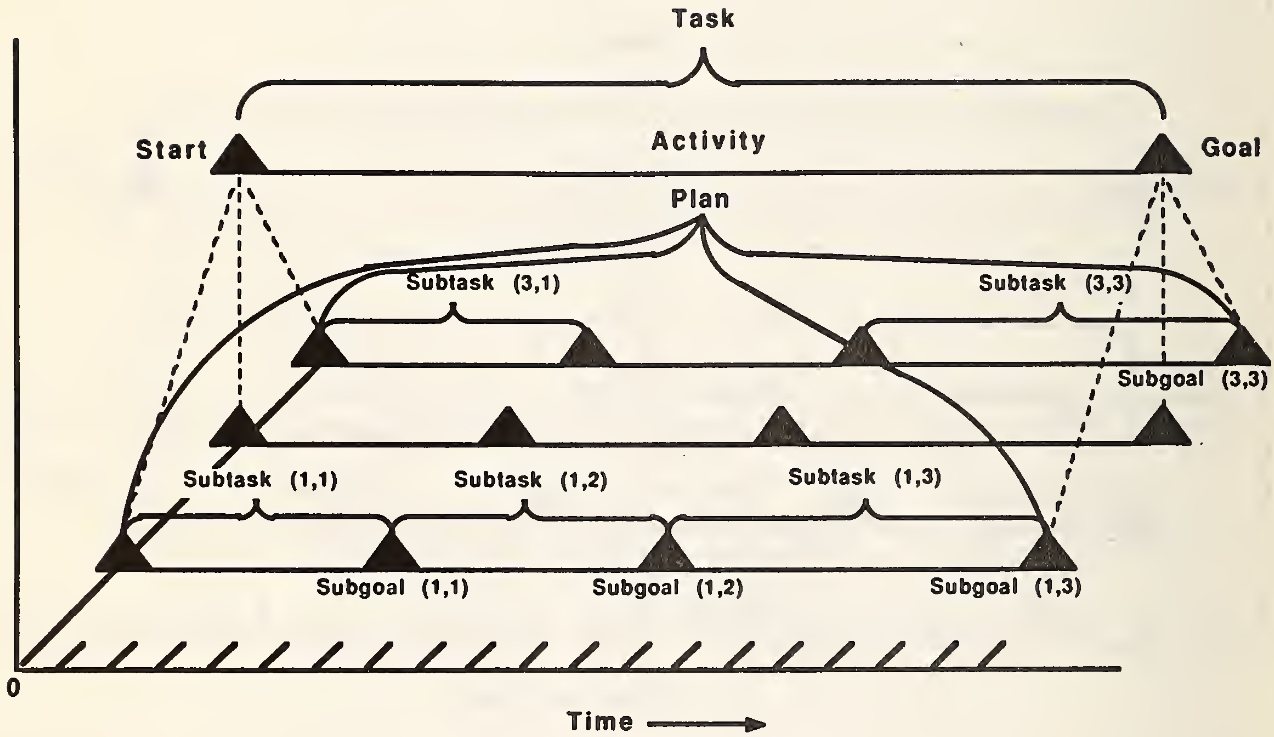
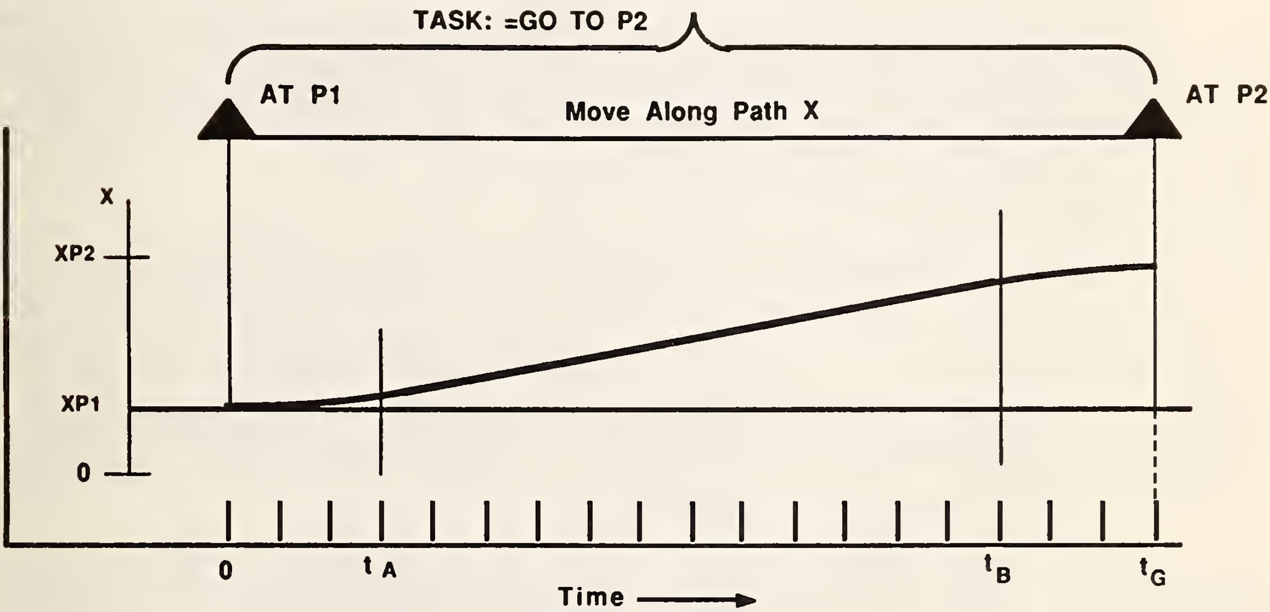


Figure 6.3: A Plan May Consist of Several Concurrent Strings of Subtasks Which Collectively Achieve the Goal Event.



$$X = \begin{cases} \text{for } 0 < t \le t_A & X = XP1 + K_1 t^2 \\ \text{for } t_A < t \le t_B & X = K_2 t + K_3 \\ \text{for } t_B < t \le t_G & X = XP2 - K_1 (t - t_G)^2 \end{cases}$$

Figure 6.4: An Example of a Path Plan for Moving from Point P1 to P2
Only the X Component of the Plan Is Shown.

In fact, any program or procedure designed to accomplish a goal is a form of plan. Plans typically are prepared before action begins, and are used to sequence activities in pursuit of the goal.

6.3 Planning

Def 5:

Planning is the preparation of a plan. Planning can be done off-line (well before the action begins), or in real-time (immediately before the action begins or as the action is proceeding). Of course, planning may combine off-line and real-time elements. For example, off-line planning may be used to develop a library of prefabricated plans, and real-time planning can then select a particular plan, or modify a prefabricated plan in order to fit the conditions that exist at, or near, execution time.

6.4 Gantt Notation

The Gantt chart notation explicitly represents the time axis, and can conveniently represent parallel simultaneous activities along the time axis. This is convenient for graphically visualizing what is happening in a control system. For example, figure 2.1 illustrates how the planners in each TD module generate a temporal decomposition. Figures 6.3 and 6.5 show how the set of subgoals which terminate the plan combine to fulfill the goal event of the input task command.

Figure 5.3 shows three levels of planning activity. The activity represented by the Gantt chart at the highest level is input to the top level TD module as a task command. This task is decomposed by the job assignment manager and three planners of the top TD module into three simultaneous plans consisting of four activity-event pairs each. The first executor of the top level TD module outputs the current subtask command in its plan to a second level TD module. This second level task command is decomposed by the job assignment manager and three planners in the second level TD module into three plans, again consisting of four subtasks each. The first of the second level executors outputs the current activity in its plan to a third level TD module, which further decomposes it into three plans of four subtasks. At each level the final subgoal events in the plans correspond to the goal of the input task. At each successively lower level, the planning horizon becomes shorter, and the subtasks become more detailed and fine structured.

Planning is done top-down. The highest-level plan covers the entire backlog of work to be done. At each lower level, plans are formulated (or selected) in real-time to accomplish the next step in the plan of the level immediately above. Thus, a goal directed control system such as is described here always has a hierarchy of plans in place. If the work goes as planned, each level of the control system will always be able to anticipate the next subtask, and there is no need to pause to replan. However, if unexpected events cause a plan to become obsolete, the system may suddenly find itself without a plan. This condition can be described as a state of "confusion," in which one or more levels has no plan available for execution.

If the activity-event pairs at each level are displayed as illustrated in figure 6.6, the resulting Gantt chart has the form of a musical score. This form suggests a possible notation for programming multiple cooperative tasks. It may even be possible to develop a programming system using a computerized form of musical scoring, or ballet notation such as Labans.

Each activity on such a chart can be described by a frame. If the proper software tools are available, it is possible to bring up a window containing a frame describing the activity simply by pointing to the activity with a cursor. This is illustrated in figure 6.7. The slots in the frame can be edited by a human process planner. The process planning system developed by Brown and McLean [Mc 84, Mc 85] for the NIST AMRF contains most of the tools required for a mining machine task planner, but does not perform the actual planning task.

6.5 State-Graph Notation

The state-graph notation has the advantage that it can be directly translated into a finite state automata (fsa).

$$fsa = \{states, transition\ table, inputs, outputs\}.$$

The nodes of the state graph are states of the fsa, inputs are planned subtask commands plus feedback $PST(s,t) + FB(s,t)$, outputs are the executor outputs $STX(s,t)$. Edges are the lines in the transition table which define the IF/THEN rules for subtask selection [Al 82].

There is an important distinction to be made between states of the control system and states of the external world. Figure 6.8 illustrates this distinction. States of the world are transition conditions for the control system, and states of the control system produce actions that cause transitions to occur in the state of the world. Therefore, the state graph of the world is a dual of the state graph of the control system. The state graph of the world can be viewed as a Gantt chart, where states are nodes and activities are edges. The state graph of the control system can be viewed as a Pert chart, where nodes correspond to states, and edges correspond to events in the world that cause the control system to transition between states.

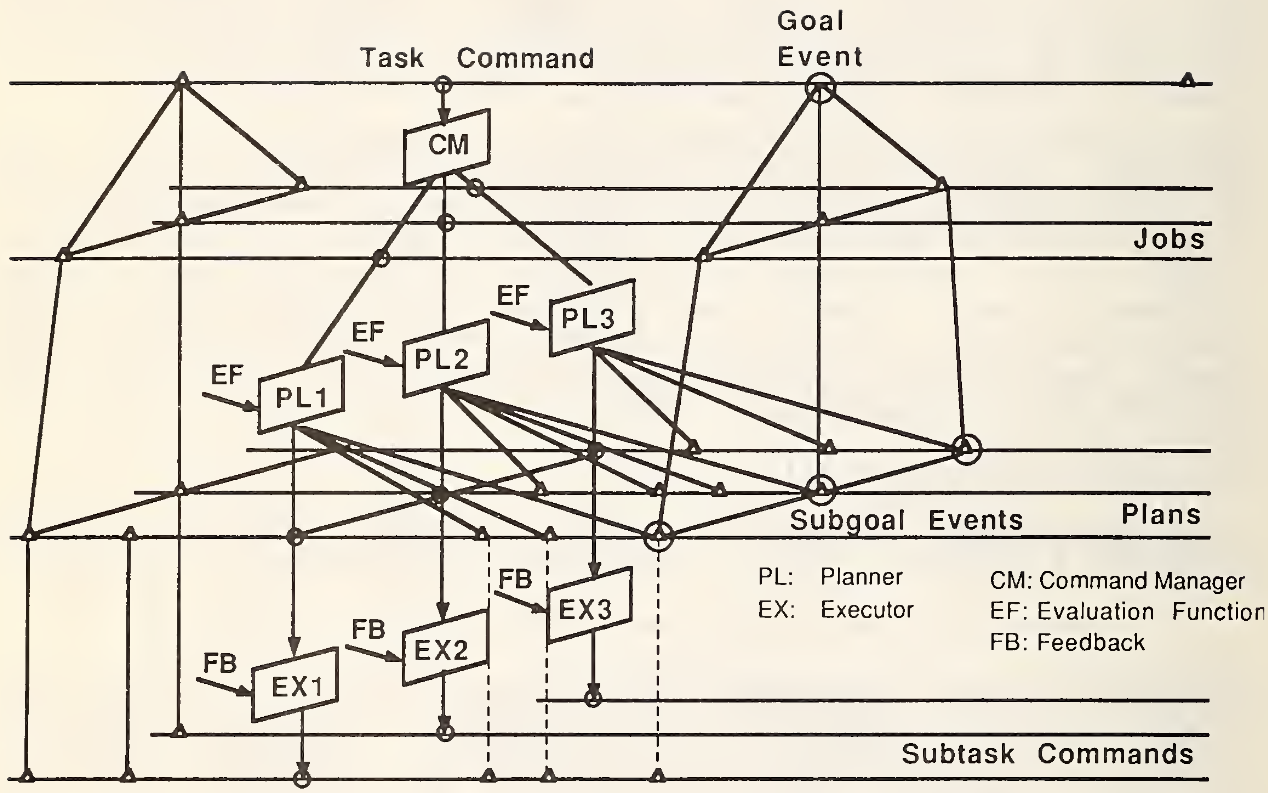


Figure 6.5: The Job Assignment Manager Decomposes a Task Command into a Set of Jobs. The Planners PL (j) Decompose Jobs into Plans, and the Executors EX (j) Translate Plans into Subtask Commands.

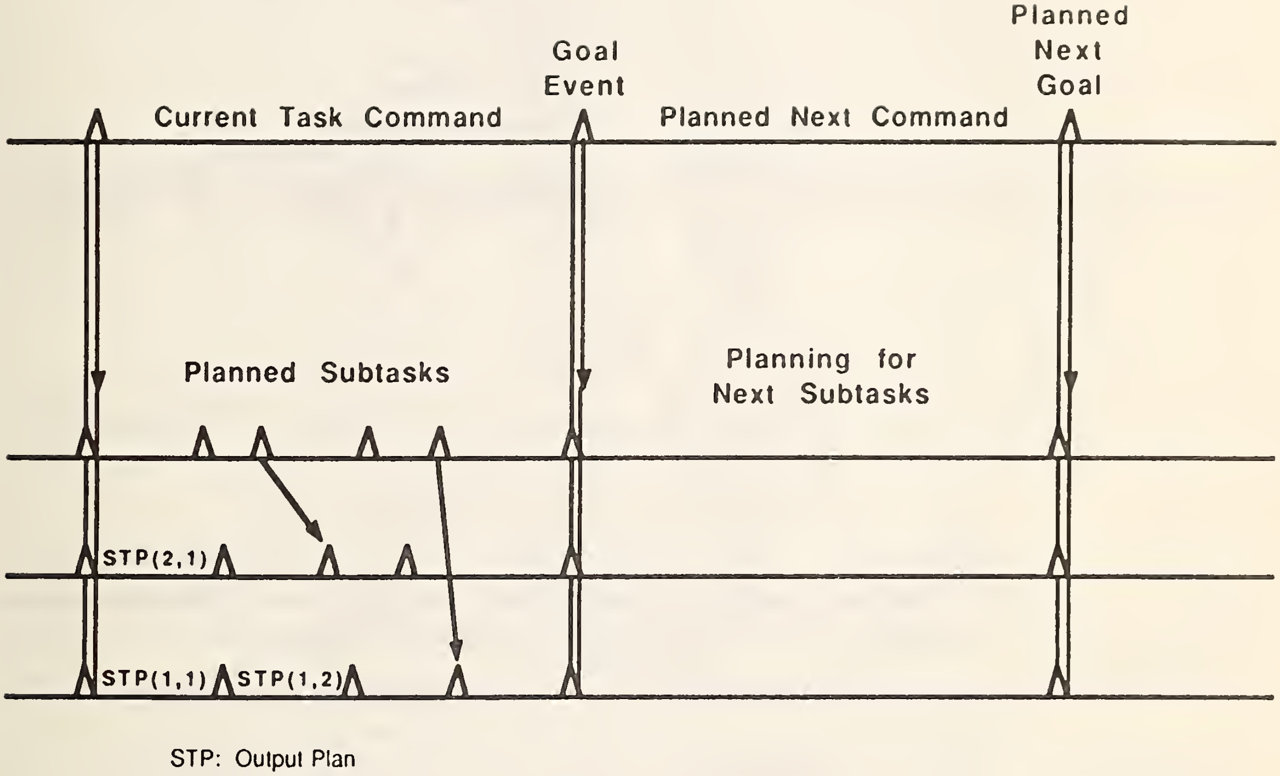


Figure 6.6: The Gantt Chart Notation for Planning Is a Potential Programming Interface

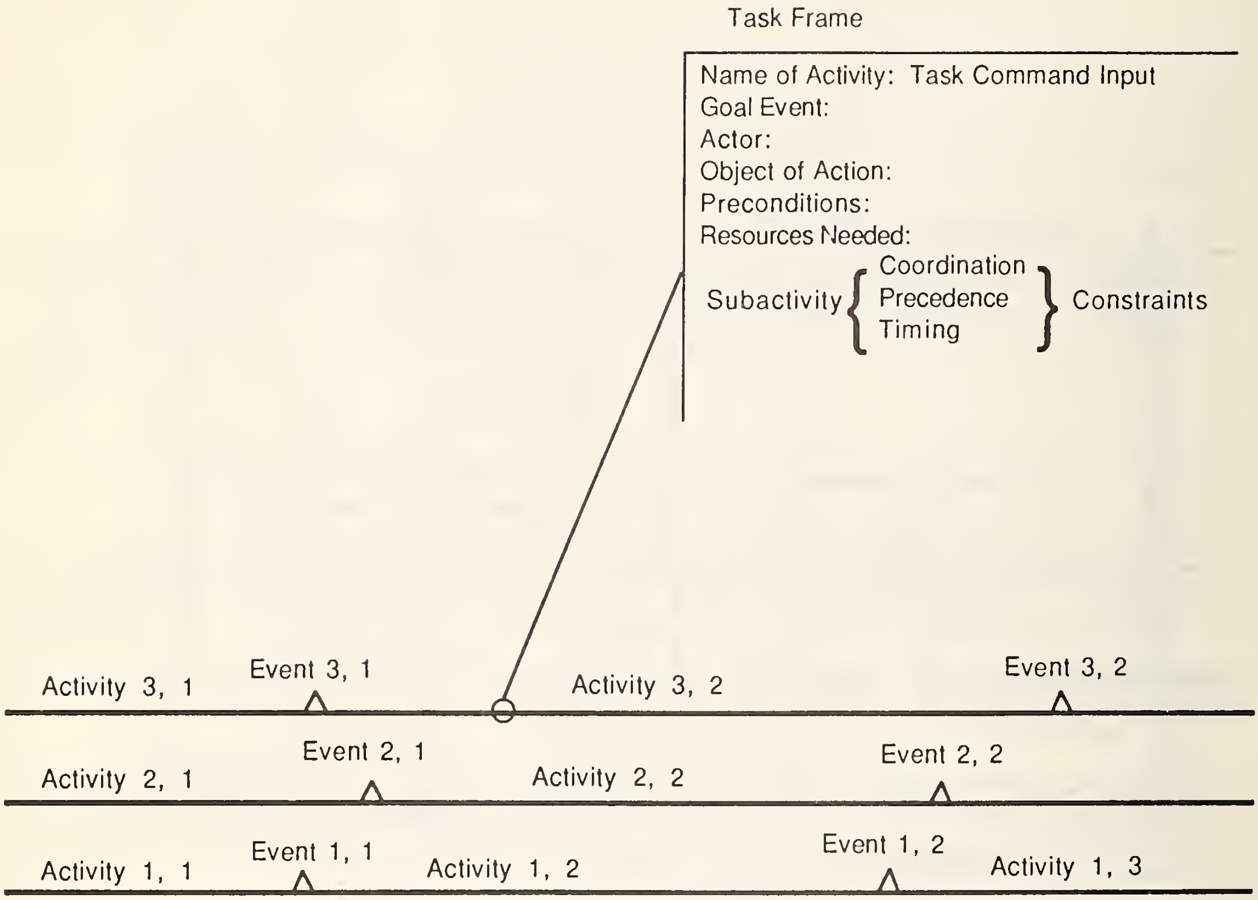


Figure 6.7: Task Frame Format

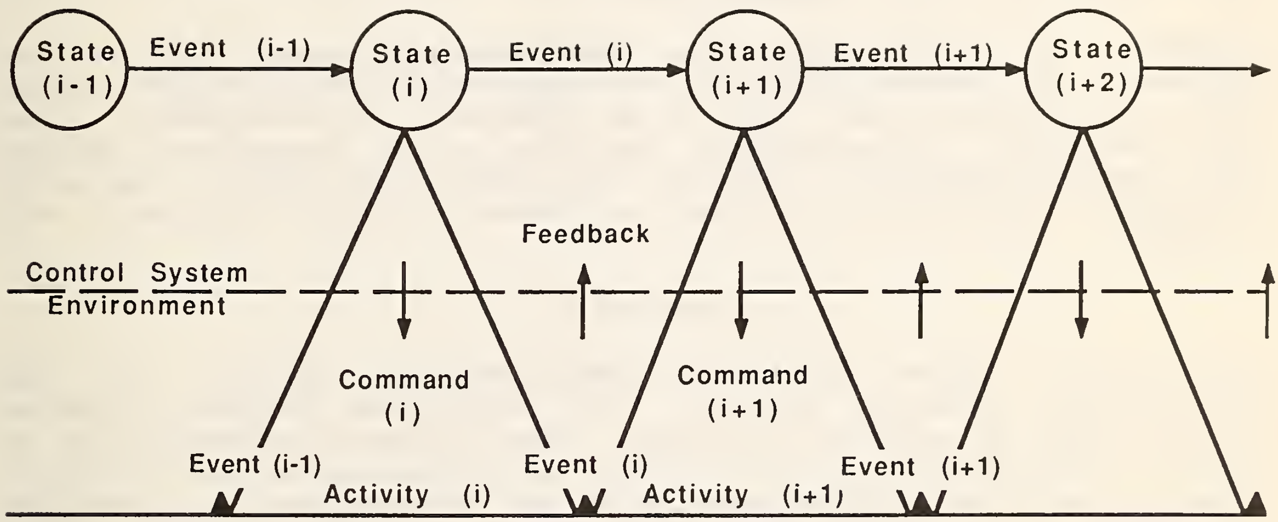


Figure 6.8: State Graph for the Control System and the External World

If plans are expressed in state-graph form, EX(s) is the fsa defined by the state graph. The state of EX(s) corresponds to the currently active node in the state graph. The output of EX(s) at time t is STX(s,t). EX(s) monitors its input PST(s,t) + FB(s,t), and discovers which line (or lines) in the fsa state transition table matches the current situation. EX(s) then executes the appropriate line in the state table; i.e., it goes to the next state called for by that line, computes the functions called, and outputs the STX(s,t) subtask output commands selected.

In the ideal case where the task decomposition works according to the plan, a planner PL(s) merely needs to add one new activity-event pair to the end of the current plan on average as often as the Executor EX(s) achieves a sub-goal event and steps to the next activity in the current plan.

However, in cases where the task execution does not go as planned, the current plan may need extensive modification, or a completely new plan may need to be generated (or selected). The time required to generate a new plan is an important system requirement parameter, and what the system does while a new plan is being computed is an important issue in error recovery and restart.

7. CONCLUSION

This first volume provides a comprehensive discussion of the theoretical aspects of the NIST real-time control system architecture as applied to mining. What is being suggested is a logical hierarchical structure for coal mine automation in which commands and status flow vertically and information is exchanged horizontally. A detailed treatment of the subject on a broad scale was presented, in order to provide a reference model for the automation of a wide variety of mining applications. This first volume can be used as a guide for the development of a mining control architecture which could theoretically be extended to automate every aspect of a coal mining operation.

The second volume will outline a portion of a simple mining operation example and discuss the application of the reference model architecture described in this volume in terms of this example. The second volume could be used as a guide to implementing the first volume in the development of a mining control architecture.

ACKNOWLEDGMENTS

The development of this document was sponsored by the U.S. Bureau of Mines, Pittsburgh Research Center, Pittsburgh, PA. We, the authors, acknowledge the assistance of Dr. George H. Schnakenberg, Jr. and his colleagues at the U.S. Bureau of Mines, and Mr. George W. Graham of the Marrowbone Development Company.

Reference to specific brands, equipment, or trade names in this document is made to facilitate understanding and does not imply endorsement by the National Institute of Standards and Technology or the U.S. Bureau of Mines.

GLOSSARY

[AC - 0] Actuators -- Within any machine designed to do physical work there are actuators which move, exert forces, and position arms, legs, hands, and eyes. Actuators generate forces to point sensors, excite transducers, move manipulators, handle tools, and steer locomotion. An intelligent machine system may have tens, hundreds, or even thousands of actuators, all of which must be coordinated in order to perform tasks and accomplish goals. Natural actuators are muscles and glands. Machine actuators are motors, pistons, valves, solenoids, and transducers.

[Ev - 0] Event -- An event is the occurrence of a condition, or situation, at a point in time. An event can be represented by a state-time vector, or by a point in state-time space.

[Fi - 0] Finite State Automata (fsa) -- A finite state automata is a functional computing module whose output depends both on the input and the internal state of the module.

A fsa is defined by a set of state vectors X, a set of input vectors S, a set of output vectors P, a state-transition table E, and a function H which maps the input and state vectors into the output vectors.

[Go - 0] Goal -- A goal is an event which successfully terminates a task. A goal is the objective toward which task activity is directed.

[Go - 1] Goal Selection -- Goals are selected by a looping interaction between task decomposition, world model, and value systems. Goals are selected by hypothesizing actions (selecting a possible plan of action), the world model predicts probable results, and the value system evaluates the predicted results. Goals are then selected by choosing the hypothesized action with the highest value as a goal to be pursued. By this process, goals are chosen, plans are generated, priorities are computed and resources are assigned to maximize benefit, and minimize cost and risk.

[Gl - 0] Global Memory -- Global memory is the database wherein knowledge is stored about the state of the world including the internal state of the control system.

[In - 0] Intelligence -- Intelligence is an amalgamation of perception, reason, emotion, and intuition which allows an intelligent entity to function and react in an uncertain environment. Intelligence is thus a product of the whole mind. It involves perception, cognition, knowledge representation, information storage and retrieval, spatial and temporal reasoning, logical deduction and inference, uncertainty and guessing, self consciousness, emotional feelings, value judgments, communication and language, goal selection, planning, task decomposition, and control of action.

[Jo - 0] Job -- A job is the portion of a task performed by a single subsystem.

[Pe - 0] Perception -- Perception is the analysis of sensory input that allows an intelligent system to build and maintain an internal model of the external world. Perception involves both the temporal and spatial processing of data from sensors. Machine perception implies a system that hypothesizes models, compares predictions from models with observations from sensors, and applies various algorithms to measure similarities and differences.

[Pl - 0] Plan -- A plan is a set of activity-event pairs which is designed to accomplish a task and produce a goal event. A plan can be represented using a Gantt chart, a plan state-graph, or a PERT chart.

[Pl - 1] Planning -- Planning is the selection, or generation, of a plan, schema, or script. There are two types of planning: search based, and rule based. Rule based planning implies the generation of a plan by traversing an "IF/THEN" decision tree. Search based planning involves the selection of a plan from a game tree generated from pre-stored library of subplans based on some search value.

[Pl - 2] Planning Horizon -- The planning horizon is the time period into the future for which a plan is prepared.

[Pl - 3] Plan Schema -- A plan schema is a plan state-graph with zero or more loops. Plans are the subset of plan schema without loops. Any plan schema can be expanded into a plan by instantiating it in time.

[Ru - 0] Rule -- A rule is a function written in IF/THEN form.

For example, the function P = H(S) may also be written as a rule of the form

IF (S), THEN (P).

[Sc - 0] Script -- A script is a generic plan for a class of tasks. A script can be turned into a plan or a plan schema by instantiating it in time with specific actors, objects, and situations.

[Se - 0] Sensors -- Sensors may include vision, tactile, force, torque, position, distance, vibration, acoustic, smell, taste, pressure, and temperature measuring devices. Sensors may be used to monitor both the state of the external world and the internal state of the intelligent system itself. Sensors provide input to a sensory processing system.

[Se - 1] Sensory Processing -- An intelligent sensory processing system compares observations with expectations generated by an internal world model. Sensory processing algorithms perform both temporal and spatial integration, so as to detect events and recognize features, objects, and relationships in the world. Sensory input data from a wide variety of sensors over extended periods of time are fused into a consistent unified perception of the state of the world. Sensory processing algorithms may compute distance, shape, orientation, surface characteristics, and material properties of objects and regions of space. Sensory processing may include recognition of speech and interpretation of language and music.

[Sp - 0] Spatial Decomposition -- Spatial decomposition consists of splitting a task into job assignments that can be accomplished by spatially (i.e., physically) distinct subsystems. Spatial task decomposition results in a tree structure, where each node corresponds to a Task Decomposition module, and each branch of the tree corresponds to a communication link in the chain of command.

[St - 0] State -- A state is a condition, or situation, which can be described by a set of state variables.

[St - 1] State Vector -- A state vector is an ordered set of state variables. A state vector defines a point in state space.

[St - 2] State Space -- State space is the space defined by all possible state vectors.

[St - 3] State-time Vector -- A state-time vector is a state vector that contains time as one of the state variables.

[St -4] State Trajectory -- A state trajectory is a sequence of state vectors, or a path traced out by a state vector as it moves through state space.

[Sy - 0] System Architecture -- Many of the elements of intelligent systems are reasonably well understood. However, Machine Intelligence is more than a set of disconnected elements. Intelligence requires an interconnecting system architecture that enables the various system components to interact and communicate with each other in an intimate and sophisticated way. A system architecture is what enables the task decomposition system to direct sensors, to focus sensory processing algorithms on objects and events worthy of attention, and to ignore things that are not important to current goals and task priorities. It is what enables the world model to answer queries from task decomposition modules, and make predictions and receive updates from sensory processing modules. It is what conveys value judgments from the value estimating system to the goal selection system as to the success of behavior and the desirability of states of the world.

[Ta - 0] Task -- A task is a piece of work to be done, or an activity to be performed. Typically a task is performed by one or more actors on one or more objects.

The performance of a task can be described as an activity which begins with a start-event and is directed toward a goal-event.

[Ta - 1] Task Command -- A task command is an instruction to perform a task.

A task command may have the form:

DO <Task> AFTER <Start Event> UNTIL <Goal Event>

[Ta - 2] Task Decomposition -- An intelligent system has processes which decompose high level goals into low level actions. Task decomposition both plans and executes actions. Intelligent task decomposition requires the ability to reason about geometry and dynamics, and to formulate or select plans based on values such as cost, risk, utility, and goal priorities. Task planning and execution must often be done in the presence of uncertain, incomplete, and sometimes incorrect information. Intelligent task decomposition must monitor the execution of tasks, and modify existing plans whenever the situation requires. Task decomposition is a hierarchical process requiring a multiplicity of planners that simultaneously generate and coordinate plans for many different subsystems with different planning horizons and different degrees of detail at each hierarchical level.

[Ta - 3] Task Frame -- A task frame is a data structure in which task knowledge is stored.

For any intelligent system, there exists a set of tasks which the system knows how to do. Each task in this set can be assigned a name.

[Ta - 4] Task Knowledge --Task knowledge is knowledge of how to perform a task; plus information such as what tools, materials, time, resources, and conditions are required; a knowledge base containing tactics and strategies which may be selected; and what are the expected costs, benefits, and risks.

[Ta - 5] Task Vocabulary -- is the set of names assigned to the set of tasks the system is capable of performing.

For each task in the task vocabulary, there exists a task frame, or data structure, of the form:

TASKNAME	-- name of the task
a) actor	-- agent performing the task
b) action	-- activity to be performed
c) object	-- thing to be acted upon
d) goal	-- event that successfully terminates the task

e) requirements -- tools, time, resources, and materials needed
-- conditions that must be satisfied to begin
-- information that may be required

f) procedures -- a state graph defining the plan (or schema)
-- functions that may be called
-- algorithms that may be needed

g) effects -- expected results of task execution
-- expected costs, risks, benefits
-- estimated time to complete

Task frames are essential to task planning. They are used by the task planners for generating hypothesized actions. They are used by the world model in predicting the results of hypothesized actions. All the slots in a generic task frame need not be filled before a task can be planned. In some cases, some slots may remain unfilled even after the task is finished.

[Te - 0] Temporal Decomposition -- Temporal decomposition consists of dividing job assignments into intermediate subtasks that are to be performed sequentially. In a plan involving concurrent activities, there may be mutual constraints between different job assignments, requiring that the activities of the subsystems be coordinated. For example, a start-event for a subtask activity in one subsystem may depend on the goal-event for a subtask activity in another subsystem.

[Va - 0] Values -- Any intelligent system must have a value system in order to make value judgments as to what is good and bad. The value system must evaluate both the observed state of the world and the predicted results of hypothesized plans. It must compute costs, risks, and benefits of observed situations and of planned activities. Without a means of making value judgments, an intelligent task decomposition system has no basis for choosing one action as opposed to another, or for pursuing one object and fleeing from another. Without a value system, any biological creature would soon be eaten by others, or destroyed by its own inappropriate actions.

[Wo - 0] World Model -- The world model contains the intelligent system's best estimate of the state of the world. The world model includes a database located in global memory in which is stored knowledge about world. The world model provides information about the state of the world to the task decomposition system so that it can make intelligent plans and behavioral choices. The world model also provides expectations and predictions to the sensory processing system in order to enhance its ability to analyze sensory data. The world model is kept up-to-date by the sensory processing system. Thus the world model is a set of software modules and databases which act as memory and servers to the sensory processing and task decomposition modules of a real-time intelligent control system.

REFERENCES

[Al 81] Albus, J.S., "Brains, Behavior and Robotics," BYTE/McGraw-Hill, Petersborough, NH, 1981.

[Al 82] Albus, J., McLean, C., Barbera, A., and Fitzgerald, M.," An Architecture for Real-Time Sensory-Interactive Control of Robots in a Manufacturing Environment," _4th IFAC/IFIP Symposium on Information Control Problems in Manufacturing Technology_, Gaithersburg, MD, Oct. 1982.

[Al 88] Albus, J.A.,"System Description and Design Architecture for Multiple Autonomous Undersea Vehicles," NIST Technical Note 1251, Sept. 1988, 126 pages.

[Ba 84] Barbera, A.J., Albus, J.S., Fitzgerald, M.L., and Haynes, L.S., "RCS: The NBS Real-time Control System," Robots 8 Conference and Exposition, Detroit, MI, June 1984.

[Ba 81] Barr, A., Feigenbaum, E., _The Handbook of Artificial Intelligence_, (Los Altos, William Kaufman, 1981).

[Bo 83] Bolles, R.C., Horaud, P., and Hannah, M.J.,"3DPO: Three Dimensional Parts Orientation System," _Proc. of The Int. Joint Conf. on Artificial Intelligence_, August 1983, p. 1116.

[Gl 79] Gleason, G.L., Agin, G.J.,"A Modular Vision System for Sensor-controlled Manipulation and Inspection," _Proc. 9th Int. Symposium on Industrial Robots_, 1979, p. 57.

[Ha 85] Hayes-Roth, B., "A Blackboard Architecture for Control, Artificial Intelligence," 1985, pp. 152-321.

[Kn 86] Knoll, T.F., Jain, R.C. "Recognizing Partially Visible Objects Using Feature Indexed Hypotheses," _Proc. IEEE Conf. on Robotics and Automation_, San Francisco, CA, 1986, p. 925.

[Mc 82] McLean, C., Bloom, H., and Hopp, T., "The Virtual Manufacturing Cell," _IFAC/IFIP Conf. on Information Control Problems in Manufacturing Technology_, Gaithersburg, MD, Oct. 1982.

[Mc 83] McLean, C., Mitchell, M., and Barkmeyer, E., "A Computer Architecture for Small Batch Manufacturing," _IEEE Spectrum_, May 1983, p. 59.

[Mc 84] McLean, C., "Process Planning Research at the AMRF," _Navy Manufacturing Technology Report_, Nov. 1984, p.5.

[Mc 85] McLean, C., "An Architecture for Intelligent

Manufacturing Control," 1985 ASME Computers in Engineering Conf., Boston, MA, Aug. 1985.

[Mi 84] Mitchell, M., Barkmeyer, E., "Data Distribution in the NBS Automated Manufacturing Research Facility," Proc. of the IPAD2 Conf., Denver, CO, April 1984.

[Pe 78] Perkins, W.A., "A Model Based Vision System for Industrial Parts," IEEE Trans. on Computers, Vol. C-27, 1978, p. 126.

[Sh 84] Shneier, M.O., Kent, E.W., Mansbach, P., "Representing Workspace and Model Knowledge for a Robot With Mobile Sensors," Proc. 7th Int. Conf. on Pattern Recognition, 1984, p. 199.

[Si 83] Simpson, J.A., Hocken, R.J., Albus, J.S., "The Automated Manufacturing Research Facility of the National Bureau of Standards," Journal of Manufacturing Systems, Vol. 1, No. 1, p. 17, 1983.

☆U.S.GOVERNMENT PRINTING OFFICE:1989-242-200/00016

NBS-114A (REV. 2-80)

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET (See instructions)	1. PUBLICATION OR REPORT NO.	2. Performing Organ. Report No.	3. Publication Date
	NIST/TN-1261 Volume 1		May 1989

4. TITLE AND SUBTITLE

Mining Automation Real-Time Control System Architecture Standard Reference Mode. (MASREM)

5. AUTHOR(S)

James Albus, Richard Quintero, Hui-Min Huang, Martin Roche

6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions)

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
(formerly NATIONAL BUREAU OF STANDARDS)
U.S. DEPARTMENT OF COMMERCE
GAITHERSBURG, MD 20899

7. Contract/Grant No.

8. Type of Report & Period Covered

Final

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP)

U.S. Bureau of Mines
Cochrans Mill Road
P.O. Box 18070
Pittsburgh, PA 15236

10. SUPPLEMENTARY NOTES

☐ Document describes a computer program; SF-185, FIPS Software Summary, is attached.

11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here)

The Mining Automation Real-Time Control System Architecture Standards Reference Model (MASREM) defines a logical hierarchical architecture for mining automation derived from a number of concepts developed in previous and on-going research programs such as the hierarchical control system developed for the Automated Manufacturing Research Facility at the National Institute of Standards and Technology.

The MASREM architecture defines a set of standard modules and interfaces which facilitates software design, development, validation, and test, and makes possible the integration of software from a wide variety of sources. Standard interfaces also provide the software hooks necessary to incrementally upgrade future mining automation systems as new capabilities develop in computer science, robotics, and autonomous system control.

12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons)

automation; global memory; hierarchical control; levels; mining; real-time control; sensory processing; task decomposition; world modeling

13. AVAILABILITY

☒ Unlimited
☐ For Official Distribution. Do Not Release to NTIS
☐ Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402.
☐ Order From National Technical Information Service (NTIS), Springfield, VA. 22161

14. NO. OF PRINTED PAGES

60

15. Price

NIST Technical Publications

Periodical

Journal of Research of the National Institute of Standards and Technology—Reports NIST research and development in those disciplines of the physical and engineering sciences in which the Institute is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Institute's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Institute's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NIST, NIST annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NIST under the authority of the National Standard Data Act (Public Law 90-396). NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published quarterly for NIST by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW., Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Institute on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NIST under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NIST administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NIST research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order the **above** *NIST publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.*

Order the **following** *NIST publications—FIPS and NISTIRs—from the National Technical Information Service, Springfield, VA 22161.*

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NIST pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NIST Interagency Reports (NISTIR)—A special series of interim or final reports on work performed by NIST for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Institute of Standards and Technology
(formerly National Bureau of Standards)
Gaithersburg, MD 20899