MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES

3 4456 0263014 2

ORNL/TM-10397
(CESAR-87/08)

ornl

OAK RIDGE
NATIONAL
LABORATORY

MARTIN MARIETTA

Dynamic Task Allocation for a Man-Machine Symbiotic System

L. E. Parker
F. G. Pin

Center for Engineering Systems Advanced Research

OAK RIDGE NATIONAL LABORATORY
CENTRAL RESEARCH LIBRARY
CIRCULATION SECTION
4500N ROOM 175
LIBRARY LOAN COPY
DO NOT TRANSFER TO ANOTHER PERSON
If you wish someone else to see this report, send in name with report and the library will arrange a loan.

CESAR

OPERATED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
NTIS price codes—Printed Copy: A04; Microfiche A01

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ORNL/TM-10397
CESAR-87/08

Engineering Physics and Mathematics Division

DYNAMIC TASK ALLOCATION FOR A MAN-MACHINE SYMBIOTIC SYSTEM

L. E. Parker and F. G. Pin

Center for Engineering Systems Advanced Research

Date Published: June 1987

Prepared for the
Engineering Research Program
Office of Basic Energy Sciences
U.S. Department of Energy

Prepared by the
Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831
operated by
MARTIN MARIETTA ENERGY SYSTEMS, INC.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-84OR21400

MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES

3 4456 0263014 2

TABLE OF CONTENTS

Section		Page
1.	INTRODUCTION	1
2.	GENERAL TASK ALLOCATION STRATEGIES	3
3.	SELECTED STRATEGY FOR A MAN-ROBOT SYMBIOTIC SYSTEM	7
4.	DYNAMIC TASK ALLOCATION METHODOLOGY	11
	4.1 KNOWLEDGE AREAS	11
	4.1.1 Constraints/Criteria	11
	4.1.2 Resources	13
	4.1.3 Tasks	17
	4.1.4 Environment	22
	4.2 FLOW OF EXECUTION	23
	4.3 COMMUNICATION LINKS NECESSARY IN TASK ALLOCATION	25
	4.3.1 Maintenance of Current Knowledge Areas	27
	4.3.2 Communication Between Task Allocator and the Resources	27
	4.4 TASK AND CONSTRAINT CHANGES DUE TO ENVIRONMENTAL CHANGES	28
	4.5 REPLANNING THE TASK ALLOCATION	29
5.	EXAMPLE OF TASK ALLOCATION COMPUTATION	35
	5.1 CONSTRAINTS/CRITERIA	35
	5.2 RESOURCES	35
	5.3 TASKS	36
	5.4 ENVIRONMENT	38
	5.5 DETERMINING TASK ALLOCATION	39
	5.5.1 Part I - Minimize Time	39
	5.5.2 Part II - Maximize Level of Achievement	41
	5.6 EXECUTION OF SUBTASKS	43
	5.7 DYNAMIC REPLANNING OF TASK ALLOCATION	44
6.	CONCLUSION	47
	REFERENCES	49
	APPENDIX -- GLOSSARY	51

LIST OF FIGURES

Figure		Page
1	Man-Robot System Scenario	8
2a	Typical Task Breakdown Tree	18
2b	Typical Subtask Breakdown Trees	20
3	Task Knowledge with Corresponding Capabilities and Merit Factors	21
4	Primary Interactions in a Dynamic Task Allocation	24
5	Example Task Breakdown	36
6	Example of Simple Environmental Map	38
7	Example Task Allocation Recommendation	43

LIST OF TABLES

Table		Page
1	Connections Requiring Communication Channels	26
2	Connections Requiring "Data Lookup" Operations	26

ABSTRACT

This report presents a methodological approach to the dynamic allocation of tasks in a man-machine symbiotic system in the context of dexterous manipulation and teleoperation. This report addresses a symbiotic system containing two symbiotic partners which work toward controlling a single manipulator arm for the execution of a series of sequential manipulation tasks. It is proposed that an automated task allocator use knowledge about the constraints/criteria of the problem, the available resources, the tasks to be performed, and the environment to dynamically allocate task recommendations for the man and the machine. The presentation of the methodology includes discussions concerning the interaction of the knowledge areas, the flow of control, the necessary communication links, and the replanning of the task allocation. Examples of task allocation are presented to illustrate the results of this methodology.

1. INTRODUCTION

During the last few decades, there has been a growing awareness and belief that automation-related technologies and intelligent machines will play an increasing role in improving the development and operation of complex and advanced systems. In this context, research and development has taken place on a broad range of technologies aimed at achieving automated systems varying from fully remotely-controlled systems (e.g. ORNL-CFRP's efforts in advanced teleoperation and servomanipulation) to fully autonomous intelligent robots (e.g. ORNL-CESAR's work in artificial intelligence, super-computing, machine vision and advanced control). Within this large spectrum of technological research, work has recently been initiated on what is proposed to be a new class of automated systems which appear promising for improving the productivity, quality, and safety of operation of advanced systems. This new type of automated system is referred to as "Man-Machine Symbiosis" and would utilize the concepts of machine intelligence and remote-control technology to achieve full man-machine cooperative control and intelligence.[4]

The ultimate function of such symbiotic systems would be to dynamically optimize the division of work between the man and the machine and to facilitate their cooperation through shared knowledge, skills, and experiences. The optimization of the man-machine partnership in both the electromotive and intellectual domain would be realized by coupling a dynamic allocation of tasks between the human and the machine with an embedded system learning capability to allow the machine, an intelligent robotic system, to learn new tasks through assimilation of experience and observation of the human.

This report presents a methodological approach to the dynamic allocation of tasks for a man-machine symbiotic system in a simplified case of dexterous manipulation and teleoperation. In this formulation, two symbiotic partners are considered: a human teleoperator and an intelligent robotic system. Both partners work toward controlling a single manipulator arm for the execution of a series of sequential manipulation tasks. Section 2 of the report briefly presents the various strategies that are used in determining appropriate task allocation methodologies, while the characteristics of the specific man-robot symbiont considered here are outlined in Section 3. The knowledge bases and level of intelligence needed to dynamically allocate tasks, along with a generalized task allocation procedure, are presented in Section 4. Examples of task allocation in the symbiotic system are given in Section 5, and a glossary of terms used in this paper is given in the Appendix.

2. GENERAL TASK ALLOCATIION STRATEGIES

Before discussing the task allocator developed in this report, a brief presentation of some basic approaches toward the allocation of tasks between man and machine is given. The type of methodology used in determining the allocation of tasks in man-robot symbiosis depends on the approach to three basic issues:

1) Sequential task vs. Multitask Problems,

2) Static vs. Dynamic Allocation,

3) Explicit vs. Implicit Communication.

First, one must determine whether the problem to be solved consists of many tasks operating concurrently (multitask problem) or only one task operating at a time (sequential task problem).[8] Multitask problems require more considerations than sequential task problems, such as coordination of tasks, monitoring of multiple simultaneous tasks, and the interaction of tasks. Appropriate classification of the problem type is necessary for a successful task allocation solution.

Secondly, it must be determined whether the task allocation strategy should be static or dynamic.[1,10,11,12,13] A static allocation approach assigns a fixed subset of the tasks to each resource at the beginning of the job execution. The resources perform only their assigned tasks, never performing any other tasks. Although this type of allocation is relatively easy to implement, it is fault intolerant. If one resource failed in performing its task, another resource could not take over the operation of that task, since it was not initially assigned the task. In addition, static allocation has the disadvantage of fostering a low utilization of resources.

A more flexible allocation strategy is the dynamic approach. In dynamic task allocation, any resource which is currently free and able to perform a task could be assigned the next task to be performed. The determination of which resource is actually assigned the task is based on the effective constraints of the problem and the current environmental status. This type of allocation is event-driven and is sensitive to environmental and constraint changes. Dynamic task allocation reduces the impact of resource failure and leads to a more effective use of system resources.

The third issue to be considered in a task allocation strategy is the type of communication to be used between the human and the automated task allocator: explicit or implicit communication.[11] In implicit, or model-based communication, the computer uses a model of the human to predict what the human is likely to do next.[9] The computer then attends to tasks which are likely to be neglected by the human. This type of communication is typically used when the human performs the majority of the tasks, with the computer taking over some tasks when the human workload becomes too large. This method of communication would not require the human to take time away from task execution to communicate with the task allocator. However, this method does require the development of an appropriate predictive model of human task selection performance. Unfortunately, this model is usually difficult to build and often results in an imperfect model of the human. Due to this imperfection, conflicts may occur when the computer incorrectly guesses the human's next action.

Explicit, or dialogue-based communication,[2] requires the human to communicate with the task allocator using an input device such as a keyboard, mouse, or lightpen, or by using his voice, buttons, or switches.

This type of communication has the advantage of minimizing misunderstanding in intent between the human and the task allocator, and is relatively easy to implement. Unfortunately, explicit communication is costly in terms of taking up more of the human's time, since the human may have to stop performing tasks to communicate with the task allocator.

Thus, the approach to the issues of sequential vs. multitask problems, static vs. dynamic allocation, and explicit vs. implicit communication is critical in determining the basic nature of a task allocator. The necessary characteristics of the task allocator discussed in this report are described in the following section.

3. SELECTED STRATEGY FOR A MAN-ROBOT SYMBIOTIC SYSTEM

The man-machine system addressed in this paper consists of two symbiotic partners, a human teleoperator and an intelligent robot system, which cooperate to perform a series of sequential manipulation tasks involving a single manipulator arm. To facilitate the division of work between the man and the robot, several automated modules are proposed to be incorporated into the system to perform responsibilities such as task subdivision, analysis, and allocation. Such a scenario can be depicted as shown in Fig. 1.

A job planner is responsible for decomposing the overall job to be performed (such as INSTALL ELECTRICAL EQUIPMENT) into its component lower-level subtasks (such as FIND WRENCH or GRASP WRENCH), indicating the order in which the subtasks must be performed. The resulting task decomposition tree (see Section 4.1.3), is passed to the task allocator, which assigns a subtask either to the human or to the intelligent controller of the manipulator. The human or the intelligent robot controller then sends controlling actions to the manipulator arm for execution of the subtask. To improve its performance and to increase its range of capabilities, the intelligent controller of the manipulator arm must ultimately use an embedded learning system to learn new tasks through assimilation of experience, observation of the human, and direct instruction.[5,6,7]

This report is concerned only with the task allocator and its relationship to the other entities in the man-machine symbiotic system. This report assumes that a complete description of the tasks to be performed is provided to the task allocator by either the human or an automated system. Research is currently being performed on automating the job planner. This report also does not discuss any details related to the

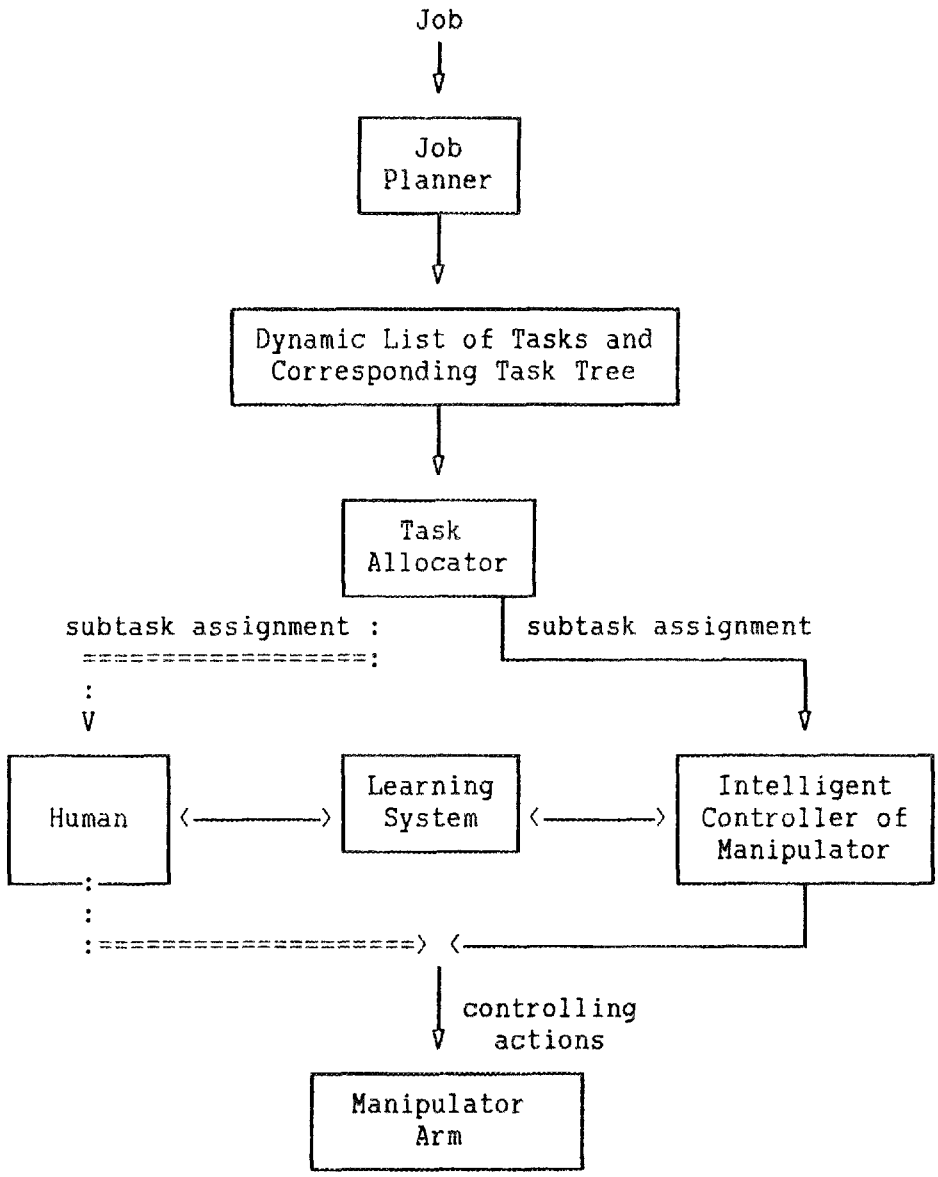


Fig. 1. Man-Robot System Scenario.

embedded learning system, which is currently being researched and will be addressed in future publications.

To determine the characteristics of the task allocator in this symbiotic system (i.e., sequential task vs. multitask problem, static vs. dynamic allocation, explicit vs. implicit communication), one can first observe that both intelligent resources (the human and the intelligent controller of the manipulator arm) are using the same medium (the

manipulator arm) to execute the subtasks. The manipulator arm actuator can receive and respond to commands from a single source at any instant in time. Consequently, the human and the intelligent robot controller cannot command the arm simultaneously or independently. In this respect, the problem to be solved is a sequential task problem. However, it is likely that while the human or the robot is performing a subtask with the manipulator arm, other actions are occurring in the background, such as monitoring of the task execution, world modeling, planning, and learning. This aspect is necessary in order for the symbiotic system to function effectively. Nevertheless, as a first step, this report will focus on the sequential task problem of allocating a series of sequential manipulation subtasks to the man and the machine. Research is currently underway to extend this methodology to allow the human and/or the robot to perform additional subtasks which compete for their time while the manipulation subtasks are being performed.

Secondly, this symbiont system must feature a dynamic (rather than a static) allocation of tasks, since both resources (the human and the intelligent controller of the manipulator arm) must be able to perform subtasks interchangeably as conditions warrant. For instance, new constraints in the problem or changing environmental characteristics may require a dynamic reallocation of tasks. In responding to such environmental and constraint changes, the task allocator demonstrates its ability to be event-driven and to cope with new situations. A reallocation is also required if the capabilities of the resources change (see Section 4.1.2), indicating an improvement or degradation in performance by a resource. Because of these requirements, the problem to be solved requires a dynamic allocation of tasks.

Finally, this symbiont system is not a system in which the human performs all of the subtasks and the robot takes over only when the human is overloaded. Instead, the system requires that the task allocator assign the "best" resource to perform each subtask for optimization of the entire problem. The task allocator must be able to communicate the subtask assignments to the resources. The human must retain control, however, and be able to approve or change the subtask allocation. These requirements mandate that explicit communication be used in the man-machine system.

In summary, the task allocator in this symbiotic system must be able to recommend a dynamic allocation of sequential manipulation subtasks to two resources, a human and an intelligent robot controller, responding to events during the subtask execution which will lead to a reallocation of subtasks, and using explicit communication to allow the human to approve or modify the allocation. The remainder of this paper will address the task allocation problem having these characteristics.

4. DYNAMIC TASK ALLOCATION METHODOLOGY

4.1. KNOWLEDGE AREAS

The purpose of the task allocator in man-robot symbiosis is to attempt to dynamically optimize the division of work between the man and the robot. Since the exact interpretation of "optimal division of work" must be allowed to vary according to the requirements of each individual problem scenario, the task allocator must know what constraints and criteria are placed on the task allocation, what the requirements of the subtasks are, and information concerning the characteristics of the environment in which the problem is to be solved. The task allocator must also have information about the capabilities of the human and the intelligent robot controller to determine the resource which is most appropriate for performing a subtask in a given scenario. The knowledge about these areas can be categorized into four main knowledge bases which are described in the following sections.

4.1.1. Constraints/Criteria

The constraints/criteria are determined by a source external to the task allocator and place performance measures, limitations, restrictions, and/or regulations on the task allocation problem solution. The intent of the constraints/criteria is to alter the task allocation strategy to adapt to differing problem contexts. The task allocator must adhere to these constraints/criteria in determining the optimal task allocation. These limitations may prevent the use of certain resources for some subtasks, or may mandate the use of certain resources for other subtasks.

Examples of possible constraints/criteria are as follows:

— minimize time of task completion,

— maximize quality of result,

— minimize human involvement (e.g. in a hazardous environment or to prevent boredom or fatigue).

The task allocator must know how to handle any constraint that is placed on the solution. For example, if the constraint is to minimize the time of task completion, the task allocator must compute the estimated time each resource will take to complete a subtask (refer to Sections 4.1.2 and 4.1.3 for further details) and then assign the subtask to the resource requiring the lesser time. For each application of the task allocator, certain constraints/criteria are initially in effect, while other constraints/criteria are ignored. Although the examples given in this report only deal with situations having one constraint in effect at a time, this methodology has the potential for being extended to handle combinations of several constraints/criteria for the optimization of the solution.

Once these constraints/criteria are determined for a particular application, they remain unchanged throughout the problem solution and execution until dynamic changes in the environment cause the constraints/criteria for the problem to be changed. If necessary, the human can also modify the constraints/criteria of the problem to cause a reallocation of the subtasks. For example, the human could experience fatigue after a long series of manipulation subtasks and could change the effective constraint from "minimize time of task completion" to "minimize human involvement". The task allocator would then allocate the subtasks by attempting to assign as few subtasks as possible to the human. In this manner, the task allocator demonstrates its ability to be dynamic, responding to changes in the constraints or criteria to reallocate the subtasks.

4.1.2. Resources

In this report, resources are defined to be intelligent entities (such as humans or computers) which are available for performing subtasks to solve a problem, or to achieve a goal. In this report, only two resources are considered: a human and an intelligent robot controller. Obviously, the task allocator must have a knowledge of the available resources before it can begin the job of task allocation. The task allocator must know what capabilities each of the resources possess, how well the resources use their capabilities in performing subtasks, how timely the resources use their capabilities to perform subtasks, and the current status of the resources (i.e., when each resource will be available to perform subtasks). The capabilities of the resources are defined in this paper to be either the abilities the resources have to perform certain physical actions, or the knowledge the resources have of certain objects. The capabilities can be defined as needed for particular applications, and could include physical abilities such as MANIPULATION or VISION, or knowledge of objects, such as WRENCH or BOLT.

Each resource can have many capabilities. However, a resource will probably not have the same level of achievement of each of its capabilities, and it certainly will not exercise each capability with identical speeds. For example, although a human has capabilities of both COMPUTATION and VISION, he probably can examine a photograph (using VISION) much easier and better than he can add a few numbers in his head (using COMPUTATION). On the other hand, a computer may also have capabilities of COMPUTATION and VISION, yet it is much more difficult for it to examine a photograph than it is for it to add a few numbers.

The knowledge about the capabilities of the resources is initially given to the task allocator as input. The actual information stored about the capabilities of the resources is directly related to the constraints which might at some time be present in the problem scenario. For example, the constraint "minimize time of task completion" requires that "timeliness of achievement" factors be provided, while the constraint "maximize quality of result" requires that "level of achievement" factors be provided. Additional constraints placed on the problem may require the storage of further information on the capabilities of the resources.

Although the knowledge about the capabilities is quantified differently depending upon whether the capability refers to a physical ability or to a knowledge about an object, one evaluation number is obtained for each factor (such as level of achievement and timeliness of achievement) of each capability. The evaluation numbers are then used to help determine the appropriate task allocation. If the capability refers to a physical ability, the evaluation number indicates the skill with which the ability is performed, perhaps on a scale from 0 to 10, or from "unacceptable" to "superior". If the capability refers to a knowledge about an object, the evaluation number indicates how complete the knowledge of that object is, perhaps on a scale from 0 to 10, or from "unknown" to "always known".

Depending on the constraints of the given problem and the subtasks to be performed, the task allocator can select the suitable resources to perform the subtasks based on the characteristics of the resources. This is done by determining what capabilities are required to complete each subtask, finding the available resources which possess the required capabilities, and applying the constraints/criteria of the problem to compute the optimal allocation.

The task allocator would thus have information as follows for the resources:

RESOURCE	CAPABILITY	LEVEL OF ACHIEVEMENT	TIMELINESS OF ACHIEVEMENT	AVAILABILITY
R_1	a_{11}	l_{11}	t_{11}	w units
	a_{12}	l_{12}	t_{12}	x units
	.	.	.	.
	.	.	.	.
	.	.	.	.
	a_{1n}	l_{1n}	t_{1n}	y units
R_2	a_{21}	l_{21}	t_{21}	w units
	a_{22}	l_{22}	t_{22}	x units
	.	.	.	.
	.	.	.	.
	.	.	.	.
	a_{2n}	l_{2n}	t_{2n}	y units
. . .				
R_m	a_{m1}	l_{m1}	t_{m1}	w units
	a_{m2}	l_{m2}	t_{m2}	x units
	.	.	.	.
	.	.	.	.
	.	.	.	.
	a_{mn}	l_{mn}	t_{mn}	y units

For example, information which could be obtained from a table such as this is as follows:

- The human has the capability of VISION, can perform VISION on a level of 10 (or "superior") with a "timeliness factor" of 2 (or "extremely fast"), and is currently available to perform VISION.
- The human has the capability of MANIPULATION, can perform MANIPULATION on a level of 7 (or "fairly good") with a timeliness factor of 4 (or "fairly fast"), but is not currently available to perform MANIPULATION. The human will be available to perform MANIPULATION in 3 time units.

- The computer has the capability to RECOGNIZE WRENCH, can RECOGNIZE WRENCH on a level of 4 ("sometimes known") with a timeliness factor of 7 ("fairly slow") , and is currently available to RECOGNIZE WRENCH.

Some important observations can be made in examining this table. First, a resource can have more than one capability available at a time, and it can also use more than one capability at a time. The use of more than one capability at a time should not be confused with the execution of more than one subtask at a time. The resource will only be performing one subtask at once, although it may use several capabilities to accomplish that subtask. For instance, a concurrent computer can use one processor for the capability VISION and another processor for the capability COMPUTATION. Likewise, humans can use the capability of VISION while using the capability of MANIPULATION to hammer a nail. Thus, the use of one capability of a resource does not necessarily mean that the other capabilities of that resource are inaccessible.

The second observation from examination of the table is that since only two resources are considered in this report (a human and a machine), the above table in an actual application would have only two entries: R1 and R2. However, the extension to m resources is possible and would allow many resources to be considered in the execution of the sequential manipulation subtasks.

As the resources execute the subtasks, the level of achievement factors and the timeliness-of-achievement factors for their capabilities may change, reflecting new knowledge about the resources. Such changes can take place in two ways: through a learning scheme and through monitoring of the resources. The learning scheme (discussed in a companion report)

allows the robot to learn and improve its capabilities by observing the human. For example, suppose the subtask to be allocated is FIND WRENCH. Initially, the robot will not know what a wrench looks like, indicated by a level of achievement factor of zero or "unknown" for the capability RECOGNIZE WRENCH. The task allocator will therefore assign the subtask to the human, who is then observed by the robot as he performs the task. In observing the human, the robot learns what a wrench looks like, and its level of achievement factor is upgraded accordingly. The allocation of the next subtask requiring the ability to recognize a wrench will take into account the new capability factors and will possibly result in a new allocation.

The second method in which the level of achievement factors and the timeliness of achievement factors can change is through monitoring of the resources. It is very important that the knowledge of the resources be consistent with the actual resources themselves. To accomplish this, some type of monitor must observe and quantify the resource's performance to determine if there is a proper correlation between the resource and the knowledge about the resource. If not, the resource knowledge base must be corrected. For example, if the human has a level-of-achievement factor of 7 (or "fairly good") for the capability MANIPULATION, but does not perform at that level after several hours of work (possibly due to fatigue or boredom), the factor should be appropriately updated in the knowledge base for use in future subtask allocations.

4.1.3. Tasks

A job planner must analyze and decompose the job to be performed into its component tasks, subtasks, and sub-subtasks. The role of the job

planner can be fulfilled by either the human or an automated job planning system. The current report does not address the operation of the job planner and assumes that the task breakdown is available as input to the task allocator. An automated job planner for the system will be addressed in a companion publication.

A typical task breakdown tree is shown in Fig. 2a.

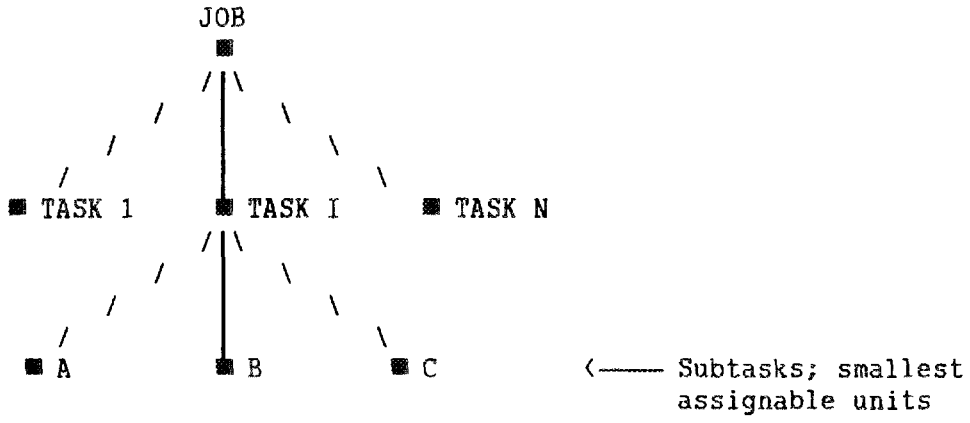


Fig. 2a. Typical Task Breakdown Tree.

The job is the highest-level description of a series of related tasks to be performed, such as ASSEMBLE MODULE. The job is decomposed into several tasks, such as INSERT ROD, which must be successfully completed by the resources in order to solve a problem, or to achieve a goal. Each task can be performed entirely by the human, entirely by the computer, or by the human and computer in cooperation. Each task is subdivided as much as needed until the smallest assignable units, or subtasks, are reached. These subtasks are the smallest units that can be feasibly assigned to a resource. For example, a task UNPLUG CABLE could consist of subtasks FIND CABLE, MOVE TO CABLE, GRASP CABLE, and PULL CABLE. It would be senseless to assign smaller components of these subtasks to more than one resource.

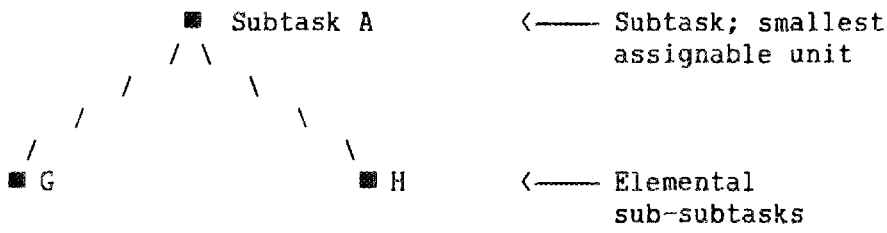
The concept of a "smallest assignable unit" is very important since it represents the smallest subdivision of the elements of a task which correlate with the physical mechanics of the actual operation of the symbiotic resources. The definitions of resources, capabilities, and smallest assignable units are, in general, system and task domain dependent.

In order to allocate the subtasks, the task allocator must know what capabilities are required to perform the subtasks and any merit factors associated with each capability. Due to the considerable differences between the intelligent robot controller and the human, the capabilities required for one of these resources to perform a subtask may be very different from those required by the other resource. Because of this, the subtasks must be further subdivided for each resource down to the elemental sub-subtasks which can be characterized by one or more capabilities and merit factors which are independent of the environment or the context of the problem. An example of the subdivision is shown in Fig. 2b.

The list of capabilities required for each subtask is obtained by traversing the lowest-level nodes (leaves), elemental sub-subtasks, below the subtask in the task breakdown tree (as shown in Fig. 2b), noting all the capabilities required for the lowest-level nodes, or elemental sub-subtasks. This traversal must be performed for each resource, since the resources have different sub-subtask breakdowns, as shown in Fig. 2b. The merit factor associated with each capability indicates the importance of that capability in the successful performance of the elemental sub-subtask, relative to the other required capabilities. The merit factors are obtained for the capabilities in manner similar to how the list of required capabilities is obtained -- by traversing the leaves of the subtask in the task breakdown tree. If any capability is required by more than one of the

subtask's elemental sub-subtasks, the merit factors associated with that capability are combined to result in one merit factor for each capability required by the subtask. At the beginning of the problem execution, these merit factors have initial values. However, as the subtasks are performed, the job planner (not addressed in this report) can alter the merit factors as necessary after each subtask completion to reflect new knowledge about the tasks. The task allocator would then derive a new allocation based on the adjusted merit factors.

For R1:



For R2:

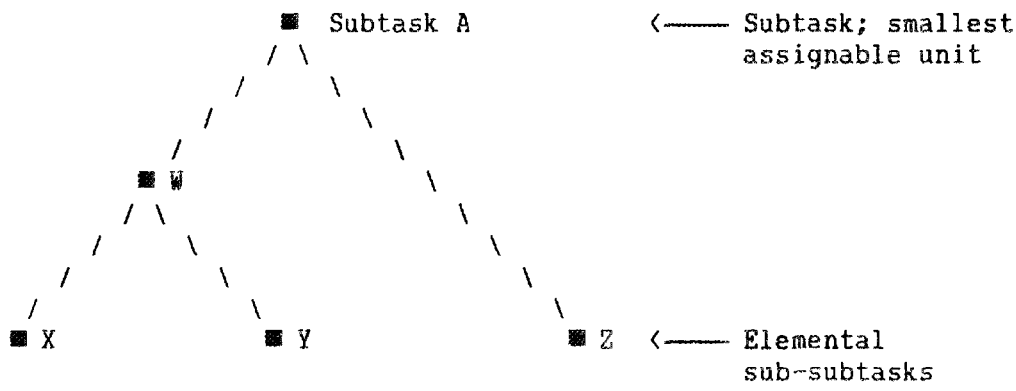


Fig. 2b. Typical Subtask Breakdown Trees.

Thus, the task allocator would have information such as that shown in Fig. 3 concerning the capabilities required to perform a task.

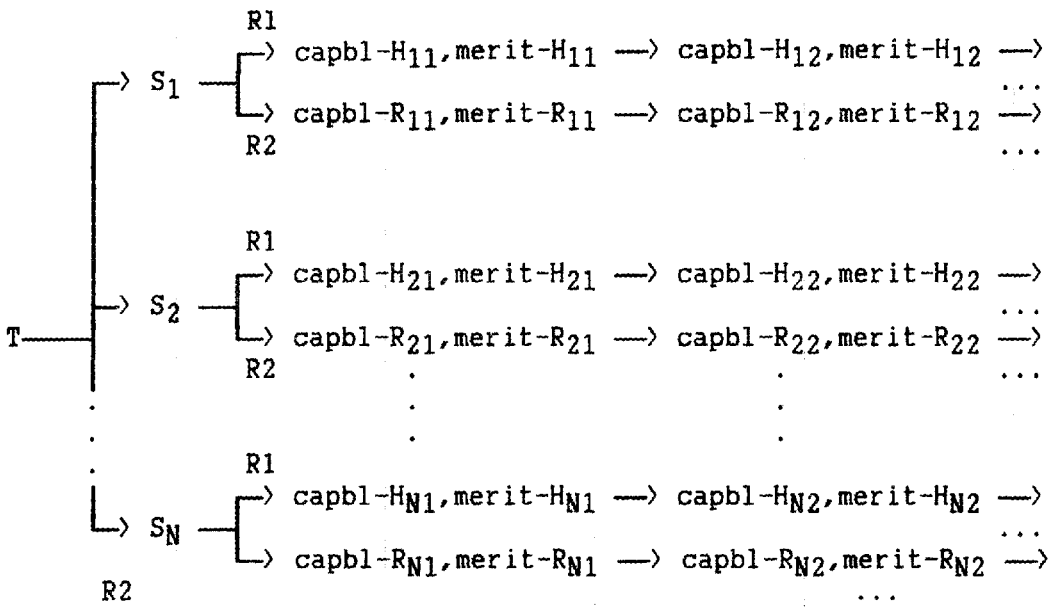


Fig. 3. Task Knowledge with Corresponding Capabilities and Merit Factors.

Figure 3 shows that task T consists of N subtasks S_1 through S_N. For each subtask, the task allocator knows the list of capabilities and merit factors required by each resource to perform the subtask. For example, to perform the subtask S_2, the human must possess capabilities "capbl-H_{21}", "capbl-H_{22}", and so on, which have merit factors of "merit-H_{21}", "merit-H_{22}", and so on. The task allocator can then compare the list of capabilities required for a resource to perform a subtask (the task information) with the actual capabilities possessed by the resource (the resource information) to determine whether the resource is capable of performing the subtask. After completing these comparisons for both resources, the task allocator can obtain the optimal subtask allocation by determining which resource most suitably meets the constraints/ criteria of the problem, and then assigning the subtask accordingly.

Although this report is addressing the allocation problem requiring only one subtask to be executed at a time (a sequential-task problem), the

extension to several machines and multitasking could be possible with this methodology by incorporating into the task allocator the ability to handle information such as precedence constraints among the subtasks.

During the execution of the subtasks, environmental changes may occur which require the job planner to update the list of subtasks to be performed. The task allocator should recognize these changes and be able to replan the task allocation appropriately. For example, if the event WRENCH DROPPED occurred, the subtask sequence would be reconfigured by the job planner to include the subtask PICK UP WRENCH. The task allocator should then respond to this event and reallocate the subtasks to reflect this change. Thus, the task allocation is dynamic, or event-driven -- it responds to changes in the work environment.

4.1.4. Environment

In order to satisfy the constraints and criteria of the problem, the task allocator may often need to have access to information about the environment. The details to be contained in the environmental knowledge base must include information on what is in the environment, what the environment looks like, and how the environment behaves. In addition, the presence of certain environmental conditions may activate certain new constraints/criteria which the task allocator must address.

The environmental information will also be accessed by the resources to help them function effectively in their environment. For example, there may be obstacles to avoid or tools available for use in performing a subtask. If the robot were told to GET WRENCH, it must know what a wrench looks like and possibly have an idea of where to find it.

Of course, the human could conclude many things about the environment by simply observing it. However, the computer must operate with an automated representation of its environment. The specific representation of the environment is highly dependent on the application and would thus vary accordingly. Possible representations include frames, rules, scripts, and nets.

4.2. FLOW OF EXECUTION

The current information about the constraints/criteria, resources, tasks, and environment will be stored in separate computerized knowledge bases, and will be shared among all the entities which need the information. These knowledge bases will be kept current by the use of sensors which monitor the resources, the environment, and the tasks, or they could be directly updated by the resources. In order for the man-machine symbiotic system to work effectively, it is important that the knowledge areas be able to interact. Figure 4 depicts the relationship between the knowledge areas.

In Fig. 4 the dotted oval indicates the actual environment. The three double-dotted lines connecting the resource and the resource knowledge, the environment and the environmental knowledge, and the task and the task knowledge indicate a close association between the physical entitites (resource, environment, task) and the knowledge of the entitites. The information which can be obtained from either the physical entities or from the knowledge of the entitites should be the same.

Figure 4 shows that the task allocation uses knowledge about the resources, environment, tasks, and constraints/criteria (links a, b, c, d) to make a task allocation recommendation. If necessary, the human task

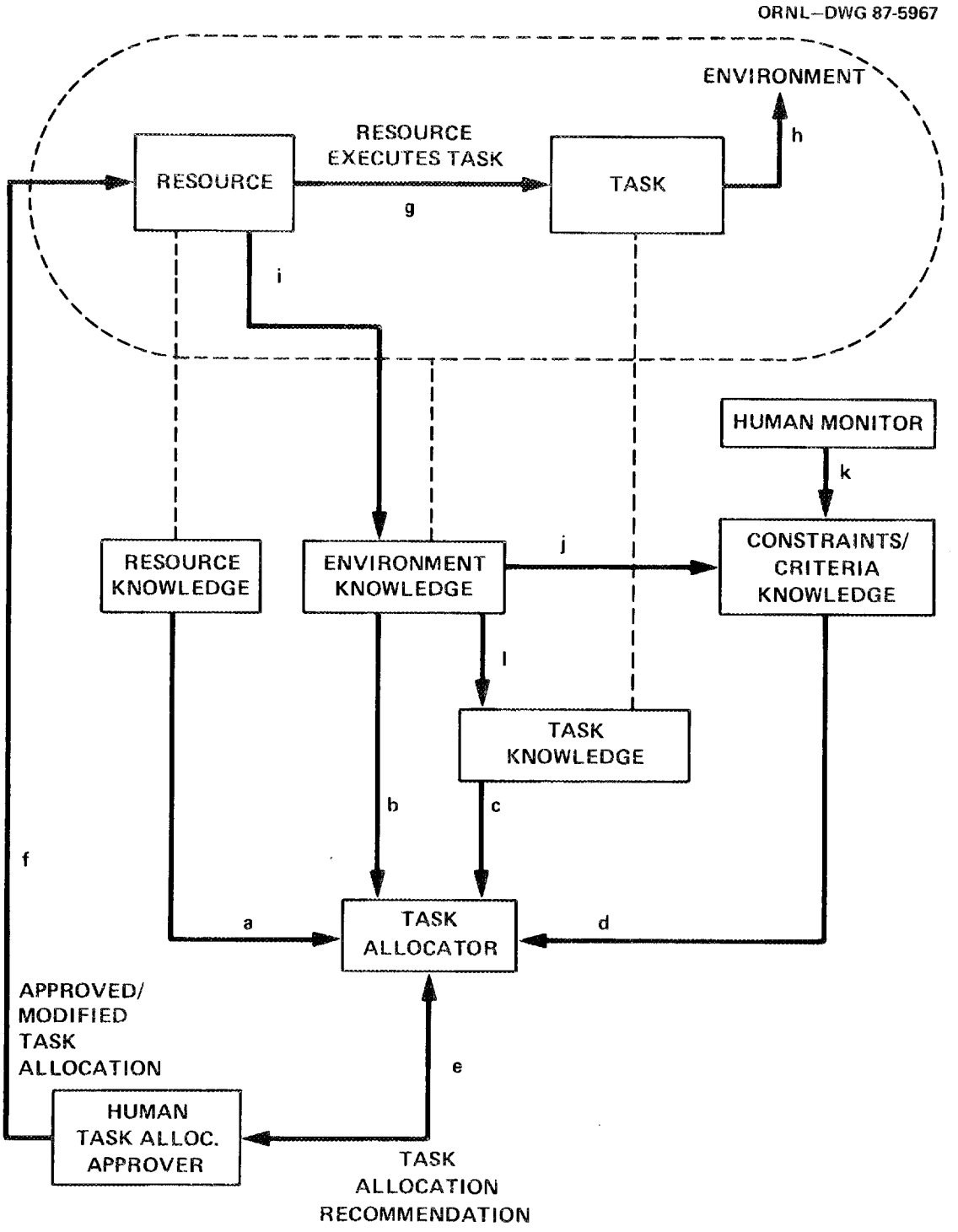


Fig. 4. Primary Interactions in a Dynamic Task Allocation.

allocation approver may change this task allocation (link e). (Note that this human need not necessarily be the same human who will perform the subtasks.) The resource is then assigned a subtask according to the approved/modified allocation (link f). As the resource executes the subtask (link g), the changing subtask status in itself modifies the environment (link h). Possibly, the resource will notice additional events or changes in the environment and will update the environmental knowledge directly (link i). As the environment changes, the constraints/criteria may need to be changed automatically to reflect the new conditions (link j), or manually by a human who monitors the problem execution (link k). (Again, this human need not necessarily be the same human who performs the subtasks or approves the task allocation.) Additionally, the list of subtasks to be performed might need to be altered because of environmental modifications (link l). Using the updated knowledge about the resources, the environment, the subtasks, and the constraints/criteria, the task allocator can replan the task allocation as necessary to repeat the cycle.

4.3 COMMUNICATION LINKS NECESSARY IN TASK ALLOCATION

In Fig. 4, it is important to know which of the interconnections require communication channels and which require only "data lookup" operations. This information is necessary to design appropriate interfaces between the various entities in the task allocation scenario. Typically, the links requiring communication channels are links between heterogeneous entities (e.g. man vs. computer) rather than homogeneous entities (e.g. processor A on a concurrent computer vs. processor B on a concurrent computer). What is referred to here as a "data lookup" operation is analogous to a computer program reading a data file; both the

program and the data file usually reside on the same computer or on similar computers. The data lookup operations are usually less complicated and The following Tables 1 and 2 categorize these connections:

Table 1. Connections Requiring Communication Channels.

1. Resources have to be able to update the environmental knowledge base with changes in the environment that they notice. This will allow the the environmental knowledge base to reflect the current environment (link i).

2. Other sensors monitoring the resources, the environment, and the tasks must be able to update the knowledge about these entities to reflect the current conditions (double-dotted lines).

3. The task allocator has to be able to communicate with the human to confirm or change the task allocation recommendation. (link e).

4. The task allocator must be able to communicate with the resources to inform them of their subtask assignments (link f).

5. The human must have access to the constraints/criteria to update this information as required (link k).

Table 2. Connections Requiring "Data Lookup" Operations.

1. Task allocator retrieval of computerized resource information (link a.)

2. Task allocator retrieval of computerized environmental information (link b.)

3. Task allocator retrieval of computerized task information (link c.)

4. Task allocator retrieval of computerized constraint/criteria information (link d.)

5. Retrieval of computerized environmental information to update applicable constraints/criteria (link j.)

6. Retrieval of computerized environmental information to update list of tasks to be performed (link l.)

The connections requiring communications channels, as listed in Table 1, can be condensed to two main areas: maintenance of current knowledge

bases and communication between the task allocator and the resources. The following sections address these two areas.

4.3.1. Maintenance of Current Knowledge Bases

The current information about the resources, environment, tasks, and constraints/criteria will be stored in separate computerized knowledge bases, and will be shared among all the entities which need the information. For instance, the task allocator will use the information to generate subtask allocations, the resources will access the environmental information for help in performing the tasks, the sensors will update the environmental information to reflect environmental changes, and so on. In order for these interactions to operate properly, it is imperative that the information in the knowledge bases be current.

These knowledge bases will be kept current by the use of sensors which monitor the resources, the environment, and the tasks, or they could be directly updated by the resources. Obviously, the sensors and the resources must have access to the knowledge bases for these updates to occur. The exact method used to maintain the databases is application dependent, requiring different methods for different circumstances.

4.3.2. Communication Between Task Allocator and the Resources

When the task allocator has a task allocation plan ready for approval, it needs to be able to communicate with the human to have the allocation approved. The task allocator must be able to accept modification of the allocation from the human and be told why the changes are needed. This communication should be as quick as possible to minimize the overhead delays in deriving the task allocation. Once the approved allocation is

derived, the task allocator must clearly and quickly communicate the definition and scope of the subtask to the resource. This is particularly important when the task allocator communicates with the human. For example, if the human knew that the next task to be performed was UNSCREWBOLT, and the task allocator gave the human the job GRASP WRENCH, the human might act ahead and begin to place the wrench on the bolt and use the wrench to unscrew the bolt. It is possible that the task allocator intended to have the human give the wrench to the robot to continue the task. Thus, it must be clear to the resource (in particular the human) exactly what the definition and scope of the subtask is.

4.4. TASK AND CONSTRAINT CHANGES DUE TO ENVIRONMENTAL CHANGES

As mentioned previously, changes in the knowledge about the environment might require an automatic change in the list of subtasks to be performed or in the constraints/criteria of the problem. Although these changes will be made by a job planner or a monitor which are not addressed in this paper, it is important to have a concept of how the changes might occur, in order to understand the dynamic nature of the task allocator. How might the task and constraint knowledge bases be updated due to environmental changes? One good way is to create a set of rules which detect certain "environmental events" when they occur. These environmental events can indicate, for instance, that a subtask is complete, that a subtask has failed, or that an unexpected occurrence has arisen. The left-hand side of the rule lists the environmental conditions which must be met for the rule to fire. The right-hand side of the rule indicates the environmental event which has occurred. For each event, certain changes in the subtasks or in the constraints may be required.

For example,

```
IF   condition-1, condition-2, ..., condition-n
THEN event-x.

IF   condition-a, condition-b, ..., condition-z
THEN event-y.

.
.
.

IF   event-x
THEN constraint-update-A,
        task-update-A.

IF   event-y
THEN constraint-update-B,
        task-update-B.

.
.
.
```

(where constraint-update-x and/or task-update-x may be empty)

In this manner, the constraints/criteria and subtasks can be updated according to the current environmental situation.

4.5. REPLANNING THE TASK ALLOCATION

When should the task allocator replan the task allocation? In order to ascertain the answer, one must first recall the task allocation method. To generate the allocation, the task allocator first examines the list of subtasks to be performed, determining the capabilities needed for each of the subtasks. The allocator then matches the capabilities required for the subtask to the actual capabilities of the resources, selecting the resources possessing the mandatory capabilities according to the constraints and criteria of the problem. The task allocation will not change unless one of the following changes:

a. list of subtasks to be performed,
b. constraints/criteria,
c. environment,
d. capabilities of the resources.

First of all, when the list of subtasks to be performed changes, the subtasks must be reallocated, since any new or changed subtasks on the list have not yet been assigned. Secondly, when the constraints/criteria of the problem change, the subtasks must be reassigned, since the entire basis for the task assignment was rooted in the constraints/criteria of the problem. Thirdly, when unexpected environmental changes occur, the subtasks must be reallocated, since the constraints/criteria upon which the subtasks were allocated may be violated by the unexpected environmental change. Fourthly, when the capabilities of the resources change, the subtasks must be reallocated, since the resources currently assigned to the subtasks may not continue to be the best resources to perform the subtasks.

It is preferable to minimize the number of times the task allocation has to be replanned, in order to reduce the computational time of the task allocator and to avoid redundant task planning. To do this, the task allocator must be able to detect when any of the above conditions "a" through "d" occurs. Changes in the list of tasks to be performed, in the constraints/criteria, or in the environment are controlled by events external to the task allocator; the task allocator cannot be expected to predict these changes. The task allocator must detect or be informed of changes in any of these three areas, and then replan the task allocation.

However, the task allocator can be more intelligent in responding to changes in the capabilities of the resources. Rather than replanning the entire task allocation whenever any level of achievement or timeliness of achievement factors change, the task allocator can predict when these factors might change and how the changes might affect the task allocation.

An intelligent task allocator can recognize when certain changes in the capability factors require replanning of the task allocation, and when they do not. For a given list of tasks to be performed with fixed constraints in a stable environment, this intelligence can greatly reduce the instances of task allocation planning, thus saving much valuable time.

How does the task allocator predict when the capabilities of the resources might change? How does the task allocator know how these changes might affect the task allocation? The answers are grounded in the basic assumption that the capability factors of a resource cannot change unless the capability is exercised either by the resource itself or by the resource learning from observing other resources using the capability. [For example, a person (a resource) cannot learn to play the piano (a capability) without practicing on the piano or watching someone else play.] Based on this assumption, the task allocator can predict when capability factors of the resources might change, thus determining when replanning is required. In this manner, the task allocator avoids replanning the allocation after the completion of each subtask.

The following example is given to illustrate how the task allocator can use this intelligence to its benefit, reducing the number of times the task allocation must be replanned.

EXAMPLE: Assume that the task allocator must allocate ten subtasks T1 through T10 to some resources. The subtasks require a total of nine capabilities S1 through S9. The subtasks have the following capability requirements, and must be performed in the order listed:

SUBTASK	REQUIRED CAPABILITIES
T1	S1, S2
T2	S3, S4
T3	S1, S4
T4	S5, S2
T5	S6, S7
T6	S3
T7	S8, S9
T8	S9
T9	S1, S2
T10	S4

The task allocator wants to minimize the number of task replanning steps which must occur due to changes in the capability factors of the resources. To do this, the task allocator first observes that the execution of subtask T1 may result in updated capability factors (i.e., level of achievement and timeliness of achievement factors) of capabilities S1 and S2 for any of the resources. A change in the capability factors of S1 indicates that the allocation of subtask T3 may change, since it also requires capability S1. Likewise, the allocation of subtask T4 may change due to the execution of subtask T1, since both subtasks T1 and T4 require capability S2. Continuing in this manner, the task allocator can determine the earliest step at which no changes in the subtask allocation will occur due to variations in the resource capability factors. The task allocator therefore avoids allocating a task which might later have to be reallocated due to changes in resource capability levels. The following table shows the earliest steps at which the subtasks can be allocated to the resources without future changes in the allocation:

Step	Subtask	Earliest step at which subtask can be allocated without further change
0		
1	T1	0
2	T2	0
3	T3	2
4	T4	1
5	T5	0
6	T6	2
7	T7	0
8	T8	7
9	T9	4
10	T10	3

Thus, at each step, the following subtasks can be assigned:

Step	Subtasks Assigned
0	T1, T2, T5, T7
1	T4
2	T3, T6
3	T10
4	T9
5	
6	
7	T8
8	
9	
10	

From this table, we see that each subtask is assigned exactly one time. The task allocator does not have to replan the entire list of subtasks after the completion of each subtask. In fact, after the completion of some of the subtasks (T5, T6, T8, T9, and T10), no additional task allocation is required at all.

If necessary, this task allocation strategy can be modified if it is preferred that the tentative allocation of all the upcoming subtasks be generated before any subtasks are executed. With this modification, the task allocator can allocate all the subtasks before the task execution

begins (i.e., at step 0), indicating which subtasks are tentative assignments, and which are firm assignments. The task allocator can then proceed as described above, indicating when the tentative subtask assignments become firm subtask assignments.

An additional time-saving measure may be taken in some applications if it is known that certain subtasks will always be executed better by particular resources. By giving these subtasks a fixed allocation, the task allocator can ignore these subtasks when computing a task allocation. This not only speeds the derivation of the task allocation, but also eliminates the need to update the capability factors of the resources after they perform these subtasks.

5. EXAMPLE OF TASK ALLOCATION COMPUTATION

To obtain a better idea of how information in the knowledge bases is actually used to dynamically allocate subtasks between man and machine, consider the task MOVE TO WRENCH AND GRASP. In this example, there are two resources: a man and a robot. The robot can either control itself, or it can be remotely controlled by the human. The following sections discuss the allocation of the subtasks to the man and robot.

5.1. CONSTRAINTS/CRITERIA

In this simple example, we assume that, on occasion, we would like to minimize the amount of time required to complete the task, and at other times we would like to maximize the quality of the result. Our constraints would therefore be represented as:

constraints/criteria	In effect?
1. minimize TIME	Y or N
2. maximize LEVEL OF ACHIEVEMENT	N or Y

5.2. RESOURCES

In this example, the resources available to perform the subtasks are a human and a robot. The capabilities of these resources may vary depending upon the particular human or robot, but in this example the capabilities, level of achievement factors, and timeliness of achievement factors are assigned as follows:

RESOURCE	CAPABILITY	LEVEL OF ACHIEVEMENT	TIMELINESS OF ACHIEVEMENT	AVAILABILITY
human	Vision	10	5	0 (available)
	Search	8	5	0
	Manipulation	7	9	0
	Knowledge of Wrench	9	2	0
robot	Vision	5	9	0
	Search	6	8	0
	Manipulation	9	6	0
	Knowledge of Wrench	4	4	0

In this table, higher LEVEL OF ACHIEVEMENT numbers indicate higher performance in that capability. Lower TIMELINESS OF ACHIEVEMENT numbers indicate quicker execution of the capability.

5.3. TASKS

In this example, the task to be performed is MOVE TO WRENCH AND GRASP. This task can be divided into 3 subtasks, one of which has 2 sub-subtasks. The relationship between the task and subtasks can be represented symbolically as in Fig. 5, with the subtasks listed from left to right in the order they are to be performed.

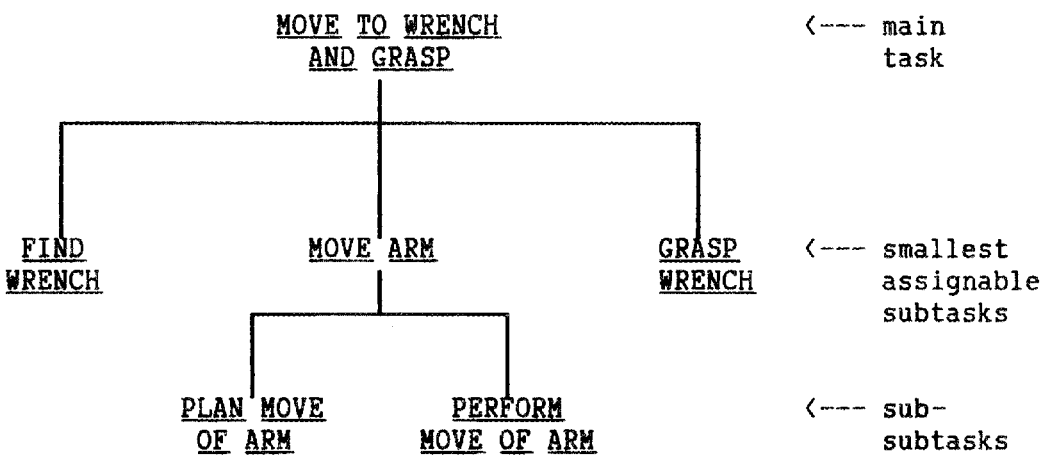


Fig. 5. Example Task Breakdown.

For the sake of clarity, this example does not break the subtasks into the elemental sub-subtasks for both the man and the machine. Recall that the elemental sub-subtasks are those which can be characterized by capabilities and merit factors which are independent of the environment or the context of the problem. For this example, assume that the capabilities required by the subtasks for both resources are the same. The subtask FIND WRENCH involves examining the environment (requires VISION) for the wrench,

and having a knowledge of what a wrench is. The subtask MOVE ARM involves planning a route to the wrench (SEARCH and VISION) and performing the move to the wrench (MANIPULATION and VISION). The subtask GRASP WRENCH involves having a knowledge of what a wrench is and looking at the wrench (VISION) while grasping it (MANIPULATION).

Thus, the required capabilities and merit factors for the subtasks can be assigned in this example as follows:

SUBTASKS	REQUIRED CAPABILITIES	MERIT
FIND WRENCH	Vision	0.4
	Knowledge of Wrench	0.6
MOVE ARM	Search	0.4
	Manipulation	0.4
	Vision	0.2
GRASP WRENCH	Vision	0.2
	Manipulation	0.6
	Knowledge of Wrench	0.2

In this example, the merit factors have been assigned to each capability to reflect the percentage of the subtask requiring use of the capability. For example, the subtask GRASP WRENCH requires use of the capability VISION 20% of the time, use of the capability MANIPULATION 60% of the time, and use of the capability KNOWLEDGE OF WRENCH 20% of the time. Although not shown in this example, the capabilities required for subtask MOVE ARM were obtained by merging the capabilities required to PLAN MOVE OF ARM and PERFORM MOVE OF ARM. Similarly, the merit factors for the capabilities required by MOVE ARM were obtained by appropriately combining the merit factors of PLAN MOVE OF ARM and PERFORM MOVE OF ARM.

5.4. ENVIRONMENT

Since the task allocator in this example will seek to either minimize the time of task completion or to maximize the level of achievement, it does not have the need to directly access the environmental information as it would if the constraint were to "assign subtasks to human when manipulator is within 6 inches of area x". However, the environmental information can indirectly affect the task allocation by causing a change in the list of tasks to be performed or in the constraints/criteria of the problem.

The intelligent robot controller, however, does need access to the environmental information to assist it in performing subtasks. A simple environment for this example would be to have a computerized map of the room available for the robot's use. This map would correspond to the important features of what the human would visually see in the room. In this situation, the robot would need to know its location in the room, where the tool shelf is (to locate the wrench), and the location of any obstacles which may be encountered. Such a map could be as shown in Fig. 6.

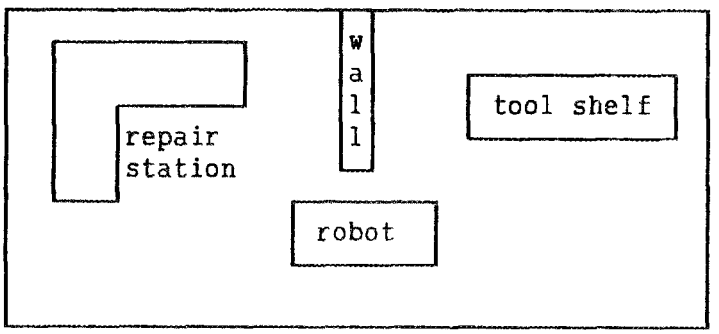
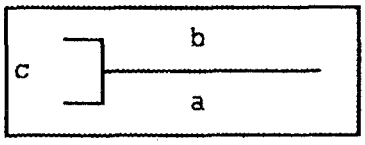


Fig. 6. Example of Simple Environmental Map.

The robot would also need to know where to find a wrench, what it looks like, and how to use it. An elementary representation of knowledge about the wrench is as follows:

NAME: WRENCH
STORAGE LOCATION: Tool Shelf
CURRENT LOCATION: Tool Shelf
APPEARANCE:



USE: Loosen bolt --
1. Grasp at points a and b, orienting c outward.
2. Fit end at c around bolt.
3. Keeping end c around bolt, rotate wrench counter-clockwise.

Of course, any additional information needed by the intelligent robot controller to perform subtasks could be included in the environmental knowledge base.

5.5. DETERMINING TASK ALLOCATION

Using the above information, it is possible to determine the optimal task allocation, based on the criteria/constraints in effect. Suppose the task allocator is instructed to first determine the task allocation using the criteria "minimize time", and then to determine the task allocation using the criteria "maximize level of achievement". The following sections describe how the task allocator would allocate the subtasks.

5.5.1. Part I -- Minimize Time

Assume the constraint in effect is "minimize TIME". The capability factor to minimize, therefore, is "TIMELINESS OF ACHIEVEMENT". To solve this problem, the task allocator examines the list of tasks to be performed. In this case, the list has one member, MOVE TO WRENCH AND GRASP. From the task knowledge base, the task allocator determines that

there are 3 subtasks to be assigned: FIND WRENCH, MOVE ARM, and GRASP WRENCH. The subtask FIND WRENCH requires two capabilities, VISION and KNOWLEDGE OF WRENCH, with merit factors of 0.4 and 0.6, respectively. From the resource knowledge base, the task allocator determines that both the human (R1) and the robot (R2) possess the capabilities of VISION and KNOWLEDGE OF WRENCH. The task allocator must now determine which resource should be allocated to this subtask, based on the constraint of minimizing time.

To do this, the task allocator computes relative time factors for both the human and the robot and then selects the resource with the lowest time factor. A resource's time factor for a given subtask is computed as:

```
time factor =
   (subtask-merit-for-capbl-1 * timeliness-of-achvmt-capbl-1)
 + (subtask-merit-for-capbl-2 * timeliness-of-achvmt-capbl-2)
 + ...
```

where "subtask-merit-for-capbl-x" is the merit factor associated with capability x which is required by the subtask, and "timeliness-of-achvmt-capbl-x" is the timeliness of achievement factor for capability x of a resource.

The following tables show the operations the task allocator would perform to compute the time factors for the human and the robot, using the resource and task knowledge given in Sections 5.2 and 5.3.

SUBTASK: Find Wrench

RESOURCE: Human
TIME FACTOR: (40% * 5) + (60% * 2) = 3.2 <--- lowest

RESOURCE: Robot
TIME FACTOR: (40% * 9) + (60% * 4) = 6.0

SUBTASK: Move Arm

RESOURCE: Human
TIME FACTOR: (40% * 5) + (40% * 9) + (20% * 5) = 6.6 <--- lowest

RESOURCE: Robot
TIME FACTOR: (40% * 8) + (40% * 6) + (20% * 9) = 7.4

SUBTASK: Grasp Wrench

RESOURCE: Human
TIME FACTOR: (20% * 5) + (60% * 9) + (20% * 2) = 6.8

RESOURCE: Robot
TIME FACTOR: (20% * 9) + (60% * 6) + (20% * 4) = 6.2 <--- lowest

---

Thus, in this example, the human would be assigned the subtask of remotely controlling the robot to FIND WRENCH and MOVE ARM to the wrench, while the robot would have the assignment of GRASPing the wrench.

5.5.2. Part II - Maximize Level of Achievement

Secondly, assume the constraint in effect is "maximize LEVEL OF ACHIEVEMENT". The variable to maximize, therefore, is "LEVEL OF ACHIEVEMENT". The steps to follow in this example are analogous to those in PART I. Here, the task allocator computes the relative quality factors for both the human and the robot and then selects the resource with the highest quality factor. A resource's quality factor for a given subtask is computed as:

```
quality factor =
   (subtask-merit-for-capbl-1 * level-of-achvmt-capbl-1)
 + (subtask-merit-for-capbl-2 * level-of-achvmt-capbl-2)
 + ...
```

where "subtask-merit-for-capbl-x" is the same as above and "level-of-achvmt-capbl-x" is the level of achievement factor for capability x of a resource.

The following tables show the operations the task allocator woulderform to compute the quality factors for the human and the robot, using the resource and task knowledge given in Sections 5.2 and 5.3.

SUBTASK: Find Wrench

RESOURCE: Human
QUALITY FACTOR: (40% * 10) + (60% * 9) = 9.4 <--- highest

RESOURCE: Robot
QUALITY FACTOR: (40% * 5) + (60% * 4) = 4.4

SUBTASK: Move Arm

RESOURCE: Human
QUALITY FACTOR: (40% * 8) + (40% * 7) + (20% * 10) = 8 <--- highest

RESOURCE: Robot
QUALITY FACTOR: (40% * 6) + (40% * 9) + (20% * 5) = 7

SUBTASK: Grasp Wrench

RESOURCE: Human
QUALITY FACTOR: (20% * 10) + (60% * 7) + (20% * 9) = 8 <--- highest

RESOURCE: Robot
QUALITY FACTOR: (20% * 5) + (60% * 9) + (20% * 4) = 7.2

Thus, in this example, the human would be assigned the entire task, remotely controlling the robot to FIND THE WRENCH, MOVE ARM to the wrench, and GRASP THE WRENCH, resulting in the highest quality result as possible. Obviously, this robot is not yet a good substitute for the human in performing this task.

5.6. EXECUTION OF SUBTASKS

After the task allocator has determined the optimal allocation, the human must optionally approve the task allocation, causing the need for link e in Fig. 4. For example, the human could be shown a display such as Fig. 7.

```
TASK: Move to Wrench and Grasp

Resource        Subtask
 Human          Find Wrench

 Human          Move Arm to Wrench
                o   Plan Move of Arm
                o   Perform Move of Arm

 Robot          Grasp Wrench

    ACCEPTABLE? _
```

Fig. 7. Example Task Allocation Recommendation.

Alternatively, the task allocator could audibly communicate the task allocation plan to the human for verbal approval. If the human disapproves of the allocation, the allocation problem can be resolved via a user-friendly interface. Otherwise, the task allocator will communicate the assignments to the resources. The human can be informed of his assignments as above via a video display or audible communication. The robot can be informed of its tasks via an electronic command.

For the first situation, in which the constraint is to minimize time (see Section 5.5.1), the human will first perform the subtask FIND WRENCH. As the human performs the subtask, there must be a method for maintaining the environmental knowledge base to reflect the current situation. Perhaps there are external sensors which "watch" the environment for task status

and update the knowledge base appropriately. Alternatively, the resources could explicitly update the environmental knowledge by informing it when certain events occur, such as WRENCH DROPPED or WRENCH GRASPED. This method of updating the environmental knowledge requires link i in Fig. 4. After the human completes the first subtask, he begins the second subtask which he is assigned, MOVE TO WRENCH, which consists of sub-subtasks PLAN MOVE OF ARM and PERFORM MOVE OF ARM. When the human completes this subtask, the robot will be informed of its task, GRASP WRENCH, and will proceed to execute that subtask.

5.7. DYNAMIC REPLANNING OF TASK ALLOCATION

As the task execution proceeds, events may occur in the environment which require the constraints to be modified and/or the list of tasks to be modified. (This requires that links i, j, and k in Fig. 4 be present.) Assume that the task allocator has assigned the subtasks according to the constraint "maximize quality of result" as described in Section 5.5.2. Assume that as the human was performing the last subtask (GRASP WRENCH), he dropped the wrench. This causes an environmental event to occur, as described in Section 5.4, resulting in a modification of the task list by the job planner. Suppose the rule concerning this environmental event is as follows:

```
IF WRENCH-DROPPED
  THEN  task-update: NEXT-TASK = PICK_UP_WRENCH
```

This causes the next task for allocation to be PICK_UP_WRENCH. The task knowledge would contain the relationship:

```
PICK_UP_WRENCH
      |
      |
MOVE_TO_WRENCH_AND_GRASP
```

which indicates that the task PICK UP WRENCH is a synonym for the task MOVE TO WRENCH AND GRASP. The task allocator could then refer to the tree shown in Fig. 5 to determine the smallest assignable subtasks to be allocated, and then allocate the subtasks as described in Section 5.5 However, assume in this example that as the human was performing the subtasks as described in Section 5.6, the robot observed the human's actions. While observing the human's actions, the robot's learning system was able to greatly improve its knowledge of a wrench, thus causing a change in the robot's level of achievement factor for the capability KNOWLEDGE OF WRENCH. The new information about the resources would then be as follows:

RESOURCE	CAPABILITY	LEVEL OF ACHIEVEMENT	TIMELINESS OF ACHIEVEMENT	AVAILABILITY
human	Vision	10	5	0 (available)
	Search	8	5	0
	Manipulation	7	9	0
	Knowledge of Wrench	9	2	0
robot	Vision	5	9	0
	Search	6	8	0
	Manipulation	9	6	0
	Knowledge of Wrench	9	4	0

To allocate the next task, PICK UP WRENCH, the task allocator would use this new resource information. Assume that the constraint in effect is still "maximize quality of result". The task allocator would compute the quality factor for the human and the robot as before:

SUBTASK: Find Wrench

RESOURCE: Human
QUALITY FACTOR: (40% * 10) + (60% * 9) = 9.4 <--- highest

RESOURCE: Robot
QUALITY FACTOR: (40% * 5) + (60% * 9) = 7.4

SUBTASK: Move Arm

RESOURCE: Human
QUALITY FACTOR: (40% * 8) + (40% * 7) + (20% * 10) = 8 <---highest

RESOURCE: Robot
QUALITY FACTOR: (40% * 6) + (40% * 9) + (20% * 5) = 7

SUBTASK: Grasp Wrench

RESOURCE: Human
QUALITY FACTOR: (20% * 10) + (60% * 7) + (20% * 9) = 8

RESOURCE: Robot
QUALITY FACTOR: (20% * 5) + (60% * 9) + (20% * 9) = 8.2 <---highest

---

In this computation, the allocation of the subtask GRASP WRENCH has changed. Instead of allocating the subtask to the human, the task allocator assigns it to the robot, reflecting the robot's improved knowledge of the wrench. Thus, to PICK_UP_WRENCH, the human would FIND WRENCH and MOVE ARM TO WRENCH, followed by the robot taking control of the manipulator arm to GRASP WRENCH.

In this manner, the task allocator demonstrates its ability to dynamically allocate the tasks. It can respond to changes in the environment, the capabilities of the resources, the list of tasks to be performed, or the constraints/criteria to be a fully event-driven system.

6. CONCLUSION

A methodological approach for dynamically allocating tasks to humans and intelligent machines involved in man-machine symbiotic systems has been presented. The necessary flow of control, knowledge areas, com-munication links, and man-machine interfaces have been outlined, and the proposed architecture has been shown to allow dynamic response and task reallocation due to changes in the work constraints, physical environment, and capabilities of the human and the machine, as well as to unanticipated events and human requests or controls. Major man-machine task allocation issues such as event-driven dynamics, knowledge updating through observation and learning, and performance-based work distribution have been discussed. Examples of task allocation have been presented to illustrate the results of the conceptual architecture in the context of remote manipulation, focusing on a system involving only two symbiotic partners, a man and an intelligent controller, sharing control of a single manipulator arm to accomplish a series of sequential tasks. The methodology, however, has been shown to be extendable to systems including more than two partners, multitasking operations, or multi-constraint situations. The architecture has been designed to be fully compatible with learning schemes and job-planning methodologies and future work will include the addition of automated monitoring, automated learning, and job planning modules to the current system.

REFERENCES

1. Chu, Y., W. B. Rouse, "Adaptive Allocation of Decisionmaking Responsibility Between Human and Computer in Multitask Situations," IEEE Trans. Syst., Man, Cybern. **SME-9**(12), 769-778 (1979).

2. Greenstein, J. S. and S. Lam, "An Experimental Study of Dialogue-Based Communication for Dynamic Human-Computer Task Allocation," Int. J. Man-Machine Studies 23, 605-621 (1985).

3. Greenstein, J. S. and M. E. Revesman, "A Monte-Carlo Simulation Investigating Means of Human-Computer Communication for Dynamic Task Allocation," Proceedings of the IEEE 1981 International Conference on Cybernetics and Society, 968-970.

4. Hamel, W. R., C. C. Jorgensen, C. R. Weisbin, "Man-Robot Symbiosis: Schemes for the Evolution of Autonomous Systems," ORNL/TM-10396 (in process).

5. Jorgensen, C. C., "Neural Network Recognition of Robot Sensor Graphs Using Hypercube Computers," Second Conference on Hypercube Multiprocessors, Sept. 29 - Oct. 1, 1986, Knoxville, TN.

6. Jorgensen, C. C., "Neural Network Representation of Sensor Graphs for Autonomous Robot Navigation," to be presented IEEE International Conference on Neural Networks, June 21 - 24, 1987, San Diego, CA.

7. Jorgensen, C. C. and C. Matheus, "Catching Knowledge in Neural Nets," AI Expert 1, December 1986.

8. Licklider, J. C. R., "Man-Computer Symbiosis," IRE Trans. on Human Factors in Electronics HFE-1, 4-11 (1960).

9. Revesman, M. E. and J. S. Greenstein, "Application of a Mathematical Model of Human Decisionmaking for Human-Computer Communication," IEEE Trans. Syst., Man, Cybern SMC-16(1), 142-147 (1986).

10. Rieger, C. A. and J. S. Greenstein, "The Allocation of Tasks Between the Human and Computer in Automated Systems," Proceedings of IEEE 1982 International Conference on Cybernetics and Society, pp. 204-208.

11. Rouse, W. B., "Human-Computer Interaction in the Control of Dynamic Systems," Computing Surveys 13(1), 71-99 (1979).

12. Rouse, W. B., "Human-Computer Interaction in Multitask Situations," IEEE Trans. Syst., Man, Cybern. SMC-7(5), 384-392 (1977).

13. Walden, R. S., W. B. Rouse, "A Queuing Model of Pilot Decisionmaking in a Multitask Flight Management Situation," IEEE Trans. Syst., Man, Cybern. SMC-7(5), 384-392 (1977).

APPENDIX -- GLOSSARY

For the purposes of this paper, the terms listed below are given the following definitions:

Capability - either the ability of a resource to perform a certain physical action, or the knowledge a resource has about a certain object.

Elemental Sub-Subtask - a sub-subtask which can be characterized for a particular resource by one or more capabilities and merit factors which are independent of the environment or the context of the problem.

Job - the highest-level description of a series of related tasks to be performed.

Merit Factor - indicates the importance of a capability in the successful performance of a subtask (or elemental sub-subtask), relative to the other capabilities required by that subtask (or elemental sub-subtask).

Resource - an intelligent entity such as a human or a machine which is available for performing subtasks to solve a problem, or to achieve a goal.

Smallest Assignable Unit - (see Subtask).

Sub-Subtask - a component of a subtask; all of the sub-subtasks of a subtask must be performed by the same resource.

Subtask (also Smallest Assignable Unit) - the smallest unit of a task which can be feasibly assigned to a single resource.

Task - actions which must be successfully completed by the resources in order to solve a problem, or to achieve a goal.

ORNL/TM-10397
CESAR-87/08

INTERNAL DISTRIBUTION

1. S. M. Babcock
2. J. Barhen
3. D. Barnett
4. M. Beckerman
5. T. Burgess
6. B. Burks
7. G. de Saussure
8. J. R. Einstein
9. C. Glover
10. M. Goldstein
11. E. Halbert
12. W. R. Hamel
13. J. N. Herndon
14. J. P. Jones
15. C. C. Jorgensen
16. S. M. Killough
17. J. L. Lucius
18-22. F. C. Maienschein
23. R. C. Mann
24. J. R. Merriman
25. E. M. Oblow
26-30. L. E. Parker
31-35. F. G. Pin
36. D. B. Reister
37. L. D. Trowbridge
38-42. C. R. Weisbin
43. B. A. Worley
44. A. Zucker
45. P. W. Dickson, Jr. (Consultant)
46. G. H. Golub (Consultant)
47. R. Haralick (Consultant)
48. D. Steiner (Consultant)
49. EPMD Reports Office
50. Central Research Library
51. ORNL Technical Library
Document Reference Section
52-53. Laboratory Records
54. ORNL Patent Office
55. Laboratory Records - RC

EXTERNAL DISTRIBUTION

56. Office of the Assistant Manager, Energy Research and Development, DOE-ORO, Oak Ridge, TN 37831

57. Dr. Wayne Book, Department of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA 30332

58. Dr. J. S. Coleman, Division of Engineering, Mathematical, and Geosciences, Office of Basic Energy Sciences, ER-15, U.S. Department of Energy - Germantown, Washington, DC 20545

59. Dr. R. J. Goldstein, Professor and Head, Department of Mechanical Engineering, University of Minnesota, 111 Church Street, SE, Minneapolis, MN 55455

60. Dr. Ewald Heer, Heer Associates, Inc., 5329 Crown Avenue, LaCanada, CA 91011

61-90. Technical Information Center, P.O. Box 62, Oak Ridge, TN 37831