CIFE CENTER FOR INTEGRATED FACILITY ENGINEERING

A Construction Planner

by
Lloyd McCara Waugh

TECHNICAL REPORT
Number 32

August, 1990

Stanford University

If you would like to contact the authors please write to:

c/o CIFE, Civil Engineering,
Stanford University,
Terman Engineering Center
Mail Code: 4020
Stanford, CA 95305-4020

A CONSTRUCTION PLANNER

By

Lloyd McCara Waugh

August 1990

This Center for Integrated Facilities Engineering Technical Report is also published as a Ph.D. dissertation in the Department of Civil Engineering of Stanford University. This report includes an extension to the Warehouse Project example which is not included in the dissertation.

ABSTRACT

The thesis of this research is that there are opportunities to extend network diagraming techniques which can be attained using a knowledge-based approach. These opportunities include: a) a one-stage treatment of resource and non-resource constraints; b) non-activity precedence: and c) recording and communicating construction scheduling knowledge. Previous research has been conducted on the use of knowledge-based approaches in construction planning. However, the areas emphasized in this research have not been addressed by previous knowledge-based construction planners.

The system developed in this research, called *A Construction Planner* (ACP), uses a knowledge-based approach to represent a construction project and its resources. It has a generic algorithm which can be applied to any construction project, thus incorporating an important procedural component. Finally, it uses rules which can easily be shared by similar projects and adapted or extended as desired.

ACP offers a more integrated approach to scheduling by considering all constraints concurrently. It offers a richer representation of the scheduling problem and, based on this representation, it allows the user to query the model about its reasoning.

ACKNOWLEDGEMENTS

I owe thanks to many people without whom the work which culminated in this thesis would not have been possible.

My primary advisor, Ray Levitt, has been my guide throughout this research and has offered relentless academic and personal support. I aspire to be as supportive to future students as he has been to me. Thanks Ray.

The remainder of my committee, John Fondahl, Boyd Paulson, and Paul Teicholz, have also been very helpful and supportive. I have relied on them to stabilize my ideas and tangents. Many thanks to John, for sharing his tremendous knowledge of scheduling and its practical application; to Boyd, particularly for his insight into my model in its early development and for his voracious appetite for thesis drafts; and to Paul, for his perspective of how ACP fits within a dynamic project model.

These professors continue to build on a rich history of scheduling research here in the Stanford Construction Engineering and Management Program; I have enjoyed attending this Program and extend appreciation for this ideal research environment.

I have received financial assistance from the Canadian Mortgage and Housing Corporation, the US National Science Foundation (grant MSM 87-16608-AI), the Stanford Construction Institute, the Stanford Center for Integrated Facilities Engineering, and the University of New Brunswick. I gratefully acknowledge this assistance.

Several of my colleagues have made this environment particularly educational—Alan Axworthy, Martin Fischer, Thomas Froese, and Nabil Kartam; without their presence, help, wisdom, and comradery my life wouldn't have been nearly as productive or as stimulating.

Finally, I want to acknowledge two especially important people. To my mother, who taught me that learning was interesting and that nothing is impossible, thanks. And to Maryhelen Stevenson whom I will marry this month, who kept me happy when everything looked bleak, and who stayed up long hours to help me, a special thanks.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES, CONTINUED

1. INTRODUCTION

Construction schedules are a prediction of the future. As such they are an essential tool for motivating personnel, an indispensable mechanism for coordinating the efforts of disparate project participants, a necessity for estimating time related costs, and a prerequisite for calculating cashflow requirements. Schedules inherently map predicted project activities and predicted project states to time. Furthermore, and as is clear from the items noted above, time often maps directly to money.

Yet we are surprisingly short of generally accepted scheduling tools. Gantt charts are a method of displaying a schedule which has been derived independently. PERT and CPM, which build on Gantt charts, have not received the wide acceptance which was originally suggested [Allam 88]. They do, however, represent precedence among activities and offer an executable algorithm which uses this precedence and the activity durations to map the activities to time. Resource leveling techniques allow the incorporation of resource constraints as a second stage subsequent to the development of the network diagram. These techniques, PERT, and CPM, are referred to in this thesis as Network Diagraming Techniques (NDTs).

One useful extension to NDTs is to automatically estimate the durations of activities rather than have them provided by the scheduler. This extension has been effectively addressed by Zozaya-Gorostiza and Hendrickson in their system called CONSTRUCTION PLANEX [Zozaya 88].

1.1. PURPOSE

The construction scheduling model developed through this research is called A Construction Planner or simply ACP. Its development has included the implementation of a rudimentary rule and frame system in a programing language called APL. The following objectives of ACP are further opportunities to extend NDTs:

- to consider resource constraints along with non-resource constraints in a single-stage approach
- to automatically generate precedence for activities rather than have it provided by the user
- to create a working model of the project which operates in conjunction with the user to develop a schedule

These objectives are not completely original, although the solutions are original and all efforts in this area are recent. References to related work such as GHOST [Navanchandra 88], CONSTRUCTION PLANEX [Zozaya 88], OARPLAN [Darwiche 88], and SIPEC [Kartam 89] are found in the remainder of this thesis.

1.2. SCOPE

Improving construction scheduling techniques is a very large goal, hence some large reductions in scope are necessary in order to make this a manageable project. The objectives noted above reduce this scope significantly and have implications which demand several scope assumptions. For all objectives, representation is important. Not only is it important to represent constraints and precedence, but it is also important to represent the project components, the component relationships, the project status, the available resources, the activities, and the scheduling process. Representation of these physical entities and concepts is necessary in order to reach the stated objectives. Although resources have been omitted from several recent construction planners, GHOST, OARPLAN, and SIPEC, they are an essential determinant of precedence and therefore are represented within ACP.

An aim to which I hope this research contributes is to have computers as assistants to construction schedulers. A competing goal is to have computers replace schedulers. I do not ascribe to this latter goal, since I believe that, for the foreseeable future, there will

remain considerations which are so specialized, or knowledge which is so unique that human review will always be required. To this end, the system which evolves on the ensuing pages maintains the options of having the scheduler query the basis for ACP decisions and to override decisions. I do, however, believe that by monitoring unsupervised performance one can obtain a good measure of an assistant's ability.

With respect to the various categories of construction, ACP will be tested primarily on building projects such as the warehouse described in Chapter 8. Modification of the knowledge bases will make ACP equally applicable to other categories of construction.

The model will not have the ability to look into the future to determine the repercussions of today's decisions. This approach is analogous to a knowledgeable manager who has many years of experience and makes decisions based on this experience rather than trying to predict project-specific implications. The model will flexibly incorporate knowledge which is derived from experience, such as, "assign high priority to all activities associated with the elevator installation."

1.3. READER'S GUIDE

Chapter 2 reviews the NDT and construction planner literature and summarizes the salient portions of this body of literature as a foundation for this research. Chapter 3 establishes several assumptions which guide the automation of construction scheduling, in particular some representation considerations, the additive nature of construction, and a chronological reasoning approach. Chapter 4 provides a detailed overview of the model. Chapter 5 demonstrates how input is entered and how it is represented—specifically, the project description, the resources, and the scheduling knowledge. Chapter 6 explains the attributes of the activities are how activities are created. Chapter 7 describes how a schedule is generated, the various categories of activities, the status, the application of knowledge bases, the scheduling algorithm, and the interface. Chapter 8 examines the NDT approach to the warehouse project and then applies ACP to the same project, first without, and then with, resource constraints. Chapter 9 summarizes with a comparison of ACP to previous efforts, the contributions of this research, and the opportunities for future work.

2. RELATED WORK

Chapter 2 describes work in three research areas related to this thesis: construction simulation, network diagraming techniques, and knowledge-based construction planners. An attempt is not made here to review all research in these areas, but rather to document those efforts which provide a basis for this research or which this research contradicts. The work described in this chapter provides a solid foundation for my research, and presents opportunities both to build upon, and to challenge the results or ideas of other researchers working in this field.

2.1. TRADITIONAL CONSTRUCTION SIMULATION

Computer simulation of construction activities is a valuable technique for analysis and synthesis of problems which are too complex to be solved using deterministic methods. Research in this area [Anderson 87; Halpin 76; Kalk 80; Paulson 78; Teicholz 63] uses stochastic arrival and processing times of individual operations to predict the durations of activities and groups of activities. Simulation techniques are most useful in situations which meet the following criteria:

- significant variations exist in activity durations
- adequate data exist to define probability distributions for the durations and variances of construction operations
- the time requirements due to interactions of these operations are not available

For example; soil is being moved using scrapers and pushers. It is observed that the scraper cycle time and pusher cycle time vary according to known probability distributions. But, the interaction of these distributions and the durations resulting from combining these machines is not available.

Although the model developed in this research has the flavor of a computer simulation in a general sense, it is not stochastic and therefore not similar to traditional construction simulation techniques.

> Models for project scheduling can be divided into two categories:
>
> - deterministic, when they assume that activity durations are fixed or may be expressed as a function of the cost incurred for doing the activities; and
> - probabilistic, when they consider activities' durations as stochastic variables with particular probability distributions.
>
> The distinction is made for several reasons. The information obtained from deterministic models is different from that provided by probabilistic models. Deterministic models are usually applied to find schedules that minimize total completion time or total budget. Probabilistic models, on the contrary, are used to estimate the total completion time of a project or to simulate the execution of the project.
>
> [Zozaya 88 p.31]

This research relies on deterministic knowledge to develop schedules.

GERT (Graphical Evaluation and Review Technique) models [Moder 83 pp.321-333; Pritsker 66; Wiest 77 pp.150-157] combine the stochastic nature of simulation with probabilistic techniques and network diagrams. Since I am emphasizing knowledge-based techniques in my research, GERT and related models will not be discussed further here (see Section 3.1. for further discussion of these alternatives).

2.2. NETWORK DIAGRAMMING TECHNIQUES

This review reveals several imperfections in the network diagraming approach. These imperfections are identified in order to point the way to a new approach. However, if used knowledgeably, network techniques are the best scheduling tool available today.

Network diagrams were developed in the late 1950s and early 1960s in several independent research programs, notably CPM supported by duPont Company and Remington Rand Univac [Kelly 61; Marshall 59; Walker 59] and PERT supported by the US Navy [Malcolm 59; Peck 62]. CPM originally emphasized the issue of time/cost trade-off and PERT originally emphasized the issue of uncertainty of activity durations. Concurrent with these efforts, John W. Fondahl at Stanford University was developing Precedence Node Diagrams [Fondahl 61; Moder 83 p.37; Wiest 77 p.13].

This research primarily investigates alternatives to network diagraming (including resource allocation and leveling) techniques. Example implementations of these techniques

include MacProject II [87] and Primavera [84]. Antill [82], Moder [83], Stevens [90], and Wiest [77] discuss applications of these techniques to construction. The graph theory which underlies networks is described in Antill [82 pp.380-391] and Fabrycky [84 pp.279-292]. The following authors have written critical reviews of network diagraming techniques: Alfam [88], Fondahl [75], Jaafari [84], Ritchie [85], and Wiest [85].

Network diagrams are an improvement over Gantt charts since they represent and graphically describe precedence and provide an executable algorithm for calculating activity times (earliest and latest start and finish) and float. The executable algorithm is of particular importance since it allows the model alone to predict the repercussions of activity duration changes (as long as these changes do not alter the network constraints) [Waugh 89a p.125]. For a discussion of the advantages of network diagraming techniques over Gantt charts, see Fondahl [75 pp.4-5].

2.2.1. Interface

In a critical review of network diagraming techniques, Ritchie [85 p.39] states

> This... does not explain why PERT, despite its much criticized shortcomings, can work well in practice. The reason is of course that planning is a dynamic process. In a well run project, the progress is reviewed at frequent intervals. ...control actions available to the project controllers are very varied. These may include transferring resources from one activity to another, increasing the availability of resources, or even changing the network logic and construction technology employed.

Following the view of Ritchie and other authors [Bergen 86 p.82], it is clear that scheduling systems must maintain an interface which makes the scheduler's actions as easy as possible.

> Much of the lack of success in applying network methods can be laid at the door of inflexible and slow to respond computer systems. [Ritchie 85 p.34]

The problem which precipitated Fondahl's [61 62] suggestion of a noncomputer approach to CPM and a similar suggestion by Lester [82] is the black box approach of network diagram algorithms.

In 1961 Fondahl stated,

> A step-by-step manual method allows the planner to retain more judgement control in making changes in the input data.... [p.11]

> While there are many disadvantages to employing a complex method without the aid of an electronic computer, there are also many justifications for a manual procedure. These justifications are possibly temporary ones and may disappear when computers become a more common tool in contractor's offices and better programs are devised that can more satisfactorily handle contractor's problems. [pp.9-10]

Levitt [85 p.57] notes that the ongoing requirement for high level management time to maintain the schedule is the Achilles' heel of network diagrams.

> Project managers and senior estimators are typically unwilling or unable to devote large blocks of time to maintain schedules for real-time planning and control purposes during a project.

This research regards the interaction between the scheduler and the scheduling model to be of ultimate importance.

This research will not specifically study interface issues, but will address the importance of this issue by using menus, charts (Gantt and sequence), and English-like rules.

2.2.2. Activities

Moder et al. [83 pp.14-17] list six steps as a summary of network-based project management methodology. These steps and a precis of the text are reproduced below:

1. Project planning
   - Define the activities.
   - Determine their technological precedence.
2. Time and resource estimation for each activity
   - The development of the network is, in a sense, the simulation of alternative ways of carrying out the project.
   - These estimates are based upon assumed manpower, equipment requirements and availability.
3. Basic scheduling
   - Calculate activity times and float.
   - Identify the critical path.
4. Time/cost trade-offs
   - Minimize the sum of direct and indirect costs.
5. Resource allocation
   - Establish an acceptable project plan for implementation may require the performance of a number of cycles of steps 3 and 4 and possibly steps 1 and 2 as well.
6. Project control
   - Check off progress against the schedule.

See Jaafari [84 p.225] for an alternate treatment of the steps involved in project planning. Moder et al. note that step 1 is the "most important step of the PERT/CPM procedure." However, only recently have tools become available to assist the scheduler with this step. These recent tools are discussed in Section 2.3.

This research identifies and creates activities using a straight-forward mapping to project components.

This research will also derive technological precedence from fundamental principles (this is discussed in detail below).

2.2.3. Constraints

In contemporary literature, the terms *technological constraints* [Adrian 85 p.221], [Fabrycky 84 p.282], [Moder 83 p.26], [Stevens 90 p.12], [Wiest 77 p.103], *natural constraints* [Moder 83 p.34], and *logic constraints* [Harris 83 p.14], [Moder 83 p.26], [Stevens 90 p.9] are typically used to refer to undefined or vaguely defined concepts. (The dictionary meaning for these words is inappropriate.) The usage of these terms falls in the same category as "gestalt" and "holistic" of which Minsky [88 p.27] states:

> True, sometimes giving names to things can help by leading us to focus on some mystery. It is harmful, though, when naming leads the mind to think that names alone bring meaning close.

This research does not use the terms technological constraints, natural constraints, or logic constraints. I use the unmodified term *constraints* when referring to all constraints collectively; I refer to non-resource constraints as *non-resource* constraints; and I refer by name to specific constraints and constraint groups (e.g. resource, crew continuity, gravity support, etc.).

It is notable that Antill and Woodhead [82 p.10 and 417] refer to additional constraints (crew, delay, equipment, hazard, management, safety, specified, and resource) as well as the typical technological and physical constraints noted by most other authors. Whiteman [88 p.192] also notes several other constraints (coordination, disturbance [of utilities], and space). A similar approach will be exploited by this research.

This research allows the user to represent arbitrary constraints.

2.2.4. Accounting for Resources

Using the definitions in the Glossary of this thesis, and the following definition, it is clear that a construction project without resources is not realistic.

> An activity is a portion of a project which consumes time or resources and has a definable beginning and ending. [Moder 83 p.23]

There are individual activities which could be viewed as not requiring resources, such as concrete curing, but a realistic construction project could not be made up of such activities.

Some authors state that they are intentionally not accounting for resources.

> Ignoring the restraints that will be placed upon the sequence of activities by resources, either labour or plant, the network... will show the logic relationships of all activities. [Harris 83 p.14]

Such authors say they are only accounting for non-resource constraints. If this were the case, then application of varying levels of resources would only further constrain the network. A review of a network diagram may indicate that changing resources can require a revision of the network. The reason for this lies in the close tie between resources and methods. Resources are inherently associated with methods. Therefore, changing resource types or quantities will often change the method used to complete the activity. This may, in turn, change the network since it is primarily the method which determines sequence constraints.

This leaves two options for planning and scheduling models:

- explicitly account for resources
- implicitly account for resources

Explicit

By explicitly stating the resources which are available, the scheduler defines the basis for the schedule and the parameters under which the schedule is expected to be valid.

However, an approach which generally is taken even in contemporary literature is to assume unlimited resources [Antill 82 p.47; Tenah 85 p.448; Wiest 77 p.103].

> The planner should be prepared for the task, and should have as open a mind as possible with regard to construction methods. Arbitrary restrictions on the activities must not be imposed, and unlimited resources should be assumed. [Harris 78 pp.27-28]

This is an acceptable approach as the first stage of a two-stage procedure (see Section 2.2.6.). However, many users never proceed to the second stage nor do they realize that the results produced by this assumption may be completely false. Earliest start and finish times and float times based on the unlimited resource assumption are seldom correct. They may be completely erroneous.

"Obviously, unlimited resources are not available." [Fondahl 75 p.9] The assumption of unlimited resources is at best unrealistic due to limitations such as the project space, the workers' safety, and the scheduler's imagination. Although this approach (assuming unlimited resources) acknowledges the need for resources to be associated with any realistic schedule, it is no better than ignoring resources from the point of view of documentation. This approach must assume rather, that resources will be explicitly stated and accounted for in the complete process of developing a final schedule.

Implicit

If resources are not explicitly considered, then they must have been implicitly assumed. This is a dangerous situation because the basis for the schedule and the parameters under which the schedule is expected to be valid are not available to the end user. Levy [87 p.102] states

> A job schedule is produced and monitored to serve as a coordination tool, to create a roadmap of the way from job start to project completion, to have a method for monitoring progress or lack of it, and to have a means of recognizing, anticipating, and compensating for the many detours that will occur along the way.

Without an explicit statement of resources, the "detours" will neither be recognized nor anticipated until it is too late. Documentation is necessary for any engineering model.

This research explicitly represents resources.

Scheduling models for specific research purposes and real-world applications have been developed which are only valid within a narrow range of resources and resource levels. In both cases it is common for these models to be discussed by those intimately

involved in the project without an explicit description of these resource assumptions. However, a full description of the model is deficient without an enumeration of its assumptions, including resources.

2.2.5. Resource versus Traditional Precedence Constraints

Barrie and Paulson [84 p.237] note that

> with basic CPM networks it is difficult to distinguish in logic between technological constraints and resource constraints

A conjecture which is implied and stated in contemporary literature [Antill 82 p.161] is that resource constraints can be represented as sequence constraints.

> We define the network logic to reflect our construction plan in a cost-effective manner, considering our available resources. [Stevens 90 p.13]

Resource constraints and traditional precedence constraints are types of scheduling constraints, as is evident from the following definitions:

- **sequence constraint**—A constraint between the start or finish of two activities. Typically the first activity must be complete prior to the start of the second activity.
- **traditional precedence constraint**—A constraint on the sequence of activities which is entered directly by the scheduler.
- **resource selection constraint**—A constraint on the sequence or scheduling of activities which stems from the method resulting from the resource(s) selected for an activity.
- **resource quantity constraint**—A constraint on the scheduling of activities which stems from the quantity of resource(s) which are available; this is affected by the sharing of resources among activities.

The difficulty which arises when the scheduler treats resource constraints and traditional precedence constraints on the same level is one of rigidity. It may be true that activity FOUNDATION cannot begin until activity EXCAVATION is complete (a plausible traditional precedence constraint). It is less likely, but possible, that the resources selected by the scheduler will be used in the field. But it is not very likely that the exact quantities of resources predicted by the scheduler will be used or even available in

the field. However, modifying a network diagram which is based on traditional precedence constraints due to incorrect resource assumptions is very difficult. By using a richer representation of all constraints, these difficulties can be avoided (see Section 3.1. for further discussion).

This research implements resource constraints.

2.2.6. Two-Stage Treatment of Constraints

The two previous sections have described some partial solutions to the difficulty of dealing with resources. Researchers have no doubt made these assumptions (ignoring resources, assuming unlimited resources, and representing resources constraints as sequence constraints) knowing that they were imperfect. I contend that the two-stage treatment of non-resource and resource constraints is the source of this difficulty.

The Moder et al. procedure listed in Section 2.2.2. implicitly assumes a treatment of non-resource constraints in step 3 and then a treatment of resource constraints in step 5. This approach is further supported by others [Antill 82 p.43; Tamimi 88 p.289].

> Resource allocation is a means of determining within the constraints of the float available an optimum combination of activity schedules which achieve contract completion. [Cormican 85 p.70]

Resource constraints are as important to an accurate project schedule as non-resource constraints. This point has been justified in the previous section. Since they are equally important, resource constraints should be considered at the same time as other constraints; they should not be treated as a second class at a later time. In addition, this approach is closer to the approach taken by human schedulers when they are unfettered by software limitations.

This research treats resource and non-resource constraints on an equal footing; i.e., all project constraints are considered concurrently.

Modern network analysis tools, such as MacProject II [87] and Primavera [84], currently allow users to represent resource and non-resource constraints and have them treated simultaneously. The major opportunity for improvement in these systems is a more rigorous look at non-resource constraints. These systems have a single representation for non-resource constraints: a sequence constraint (i.e., precedence link). The need for a

more fundamental representation and examples of this representation are described in Section 3.1.

Furthermore, resource constraints are not independent of non-resource constraints. Therefore separate treatment leaves all coordination and priority decisions to the user. This strands the user with no other option than to make these decisions through trial-and-error. In situations where there are many resource and non-resource constraints, it is unlikely that the user will even be aware of the default choices which are produced by the two-stage approach.

A partial, non-trial-and-error solution is to use simple fixed heuristics (e.g., minimum slack activity first, shortest duration activity first) to resolve conflicts in resource constrained scheduling. But the user is still not aware of when the heuristic was applied or what activities it was applied to; the black-box, batch-mode approach referred to by Fondahl [61 62] and noted in Section 2.2.1. remains.

This research offers a query facility through which the user can determine which constraints were applied in generating a schedule.

2.2.7. Representation Limitations

> vital ingredients [to the schedule] cannot be and are not expressed in CPM. [Birrell 87 p.345]

Limitations of early computer systems have influenced network diagraming theory in contemporary literature. Three examples follow.

Activities with Common End Points

In the activity-on-arrow representation, the *i-j* naming convention requires that dummies be inserted when two activities have the same start and finish nodes [Harris 78 p.35; Moder 83 p.29; Stevens 90 p.26; Tenah 85 p.427; Wiest 77 p.9]. This has nothing to do with either the real-world project or the basic scheduling concepts; it is necessary only because of the representation and conventions chosen. Although the *i-j* convention is a significant improvement over Gantt charts, it is an example of representation complicating understanding. (See Fondahl [68] for a further description of difficulties with the activity-on-arrow representation.)

Redundancy

Deletion of "redundant" sequence links (extra dummies in the activity-on-arrow representation) is typically suggested in contemporary literature.

> .. each activity is listed and a list of activities that must logically precede it is developed. This list is then reduced to an immediately preceding activities list which shows only the activities immediately preceding any given activity. [Stevens 90 p.9]

However, the only links that remain, after applying the techniques for removing redundancy described by Harris [78 p.40], Stevens [90 p.14] and Wiest [77 pp.22-25], are those which *the scheduler currently believes will be the most constraining*. This can result in sequence errors if unwittingly applied prior to finalizing the network. Even if prudently applied, there is a loss of sequence information which will haunt the scheduler when network revisions are required.

Consider the network fragment shown in Figure 2.1a. Assume that activities A and C are connected by a gravity support constraint—g and that activities B and C are connected by a safety constraint—s. Since link g is "redundant," the procedures noted above suggest its removal, resulting in Figure 2.1b being sent to the field.

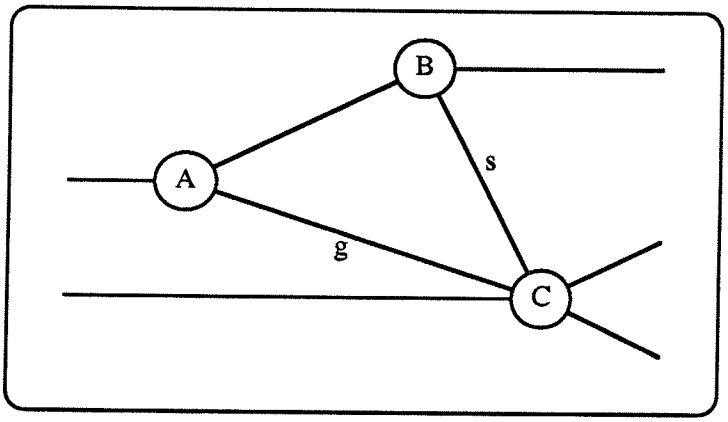

Figure 2.1a Network Fragment: With Redundancy

During project execution, unexpected circumstances arise and the safety constraint is removed (the analogy is equally valid for any constraint which is later removed by unforeseen circumstances). When the project manager removes link s (Figure 2.1c), it appears that activity C can start prior to activity A

One key to a robust, flexible model in a dynamic environment is an appropriate degree of redundancy. Redundancy is highly beneficial in the monitoring and control phases of construction projects, since schedule revisions are the norm. It would be possible to retain links from each activity to all of its predecessors and all of their predecessors. These predecessors of predecessors will not be retained since a truly redundant network would result, but I will retain at least one sequence link for each constraint type.

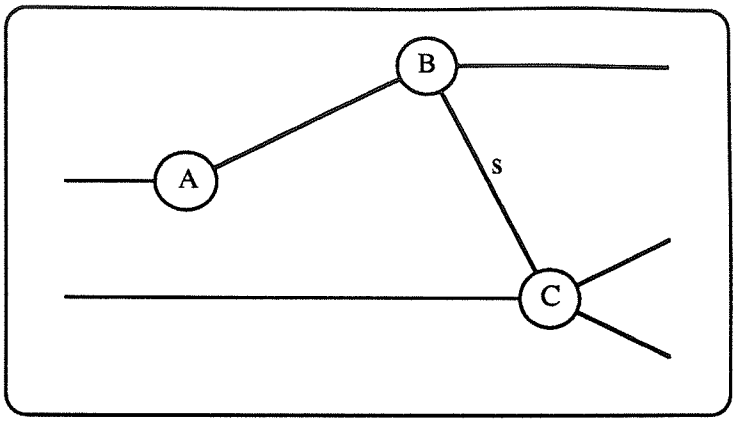

Figure 2.1b Network Fragment: Without Redundancy

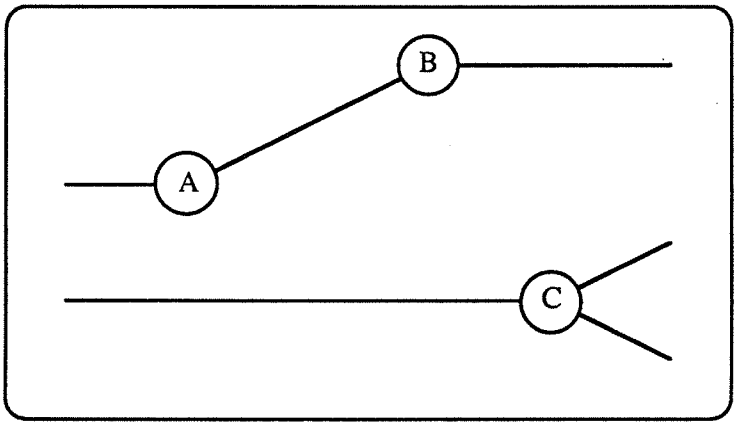

Figure 2.1c Network Fragment: Revised

This research will maintain redundancy among constraint types.

The model described in subsequent chapters does not represent constraints as lines between nodes as network diagraming techniques do, but the concept of constraints is common to both representations.

Resources

The over-zealous use of sequence constraints to represent resource constraints discussed in Section 2.2.5. appears to lie in the limitations of hardware and software which were used in the early stages of PERT/CPM development. A user at that time who

recognized the need to represent resource constraints and did not have any other technique for constraint representation would understandably resort to this option.

The two-stage treatment of non-resource and resource constraints discussed in Section 2.2.6. can also be attributed to the limitations of computers in the 1960s.

In summary, representation often limits the reasoning capabilities of a model.

This research attempts to represent more knowledge in the construction planning domain, more robustly, and more flexibly.

If this and other similar research efforts are successful and accepted by industry, this enhanced view of constraints will be analogous to the improvement that network diagrams offered over Gantt charts due to their additional representation of sequence constraints.

2.2.8. Decision CPM

"Decision CPM" was designed by Crowston and Thompson [67] and is described by Wiest [77 pp.146-150]. This technique was an early attempt to bring together the planning and control phases of a project by adding decision nodes to the PERT/CPM representation. These decision nodes are documented by the scheduler prior to evaluation of the network.

The decisions referred to in the literature are based on the lowest cost of the total project. The cost of each activity is entered when the network is designed and optionally updated. Costs are added without significant modification of the network diagram representation. Although useful, this scheme serves a very narrow need. There are many other types of knowledge in addition to cost which are prevalent in construction (see Section 3.1.).

Decision CPM is an improvement over the limited representation which was discussed in the last section and is indicative of human planning. However, Decision CPM has the following drawbacks: (a) it requires the scheduler to enumerate all alternatives, and (b) it is limited by its lack of ability to represent and manipulate activity attributes other than cost.

Knowledge-based approaches offer a new generation of representation techniques which have the potential to overcome the representation difficulties of network diagraming techniques.

2.3. KNOWLEDGE-BASED CONSTRUCTION PLANNERS

For a theoretical review of AI planners see Tate [85] and Kartam [89 pp.8-20]. Levitt [87], Levitt [88], and Zozaya [88 pp.7-64] provide excellent reviews of construction planners. The following sections review the planning models which are relevant to this research and note the concepts from these models on which this research builds.

2.3.1. Domain-Independent Means-End Planners

Generating plans is not a new research topic: GPS [Newell 63], STRIPS [Fikes 71], ABSTRIPS [Sacerdoti 74], NOAH [Sacerdoti 75], INTERPLAN [Tate 75], NONLIN [Tate 76; Tate 77]. Extensions to these planners are underway: DEVISER [Vere 83] and O-PLAN [Currie 85]. Difficulties noted regarding these planners are summarized by Levitt [87 p.12].

> In projects where the preconditions for each action are confined to the effects generated by the completion of other unique actions, the only feasible sequence of activities is already implicit in the unlinked list provided as input to the planner....
>
> In developing a plan for this type of project, therefore, no new knowledge is generated by the means-end planner; the unique set of sequential relations which were implicit are simply made explicit.

2.3.2. GHOST

This construction planner [Navinchandra 88] begins with all activities in parallel and uses critics to force precedence.

GHOST has the following critic knowledge sources:

- critics that know about physics
- critics that know about construction
- refinement critics
- critics that check for redundancy

[Navinchandra 88 p.253]

The example given [Navinchandra 88 pp.250-252] applies the critics in the order listed above. The following critics are noted:

- Physics critics
  supported-by
- Construction critics
  curing time of concrete
  placing of rebars before pouring concrete
  placing of crane before concrete can be transported to elevated sites

[Navinchandra 88 pp.245-246]

Three concepts used by GHOST which are useful for this research are

- the initial state of having all activities in parallel,
- the use of critics (see discussion of constraints in Section 2.2.3.), and
- the removal of redundancy (see Section 2.2.7.).

This research will produce a schedule with all activities in parallel if no constraints are imposed on the system.

This is a necessary assumption in order to ensure that all constraints are explicit. Models which start with an assumed ordering of activities have implicit ordering constraints which are not revealed to the user.

2.3.3. SIPEC

Kartam represents construction projects using a detailed frame hierarchy [Kartam 89 p.73] in addition to STRIPS style operators. These operators, which are associated with project components, are used to generate activities and their precedence links. The following operators are used to create a structural concrete schedule for an office building:

- Build-CF to build a column and its associated footing at the ground level
- Build-Column to build a column at the first and subsequent floors
- Build-Beam to build a beam
- Build-Deck to build a deck
- Build-EW to build an exterior wall

[Kartam 89 p.69]

These operators are in a sense hierarchical, since one operator may call one or several other operators (e.g. BUILD-Beam calls Form, Rebar, Cast, and Cure [Kartam 89 p.71]).

The model is given a goal such as:

- construct a specific deck [slab]
- construct the first-floor structure
- construct the structure of the project
- construct the finishes of the first floor
- construct the whole project (structure and finishes)

[Kartam 89 p.77]

Using the goal, SIPEC backward chains using operators. These operators draw on constraints. For example, if the goal is to build a beam, the operator Build-Beam is called. This operator searches for the columns with upper end points which are the same as the end points of the beam. The model then selects one of these columns as a subgoal and checks:

Case 1 if the column has already been constructed (this will only be the case if this column was part of a previous subgoal), or

Case 2 if there are any further subgoals as a result of the column operator.

If, in case 1, the column has already been constructed, then the goal is established and the column at the other end of the beam is considered. If, in case 1, the column hasn't been constructed, then case 2 is applied. If, in case 2, the column operator has no further subgoals, then this subgoal has been established and the column at the other end of the beam will be considered using the same procedure. If, in case 2, the column operator has subgoals, then they will be followed using the same procedure.

This process is continued until all goals are met. A goal satisfaction trace is equivalent to the sequence of activities in the project network with the following alterations:

- All extremities of the trace are tied together with a node called "start."
- The trace is drawn in reverse order with the top-level goal as the final node.

The project hierarchy has components at its extremities and groupings or superclasses at each higher level. This type of hierarchy is of particular interest to this research. Although this is a very simple representation, it provides a potent source of class and relationship knowledge for reasoning purposes.

This research uses hierarchies as a method of adding class and relationship knowledge to elements such as project components and resources.

2.3.4. OARPLAN

Darwiche [88] employs (a) the representation aspects of PIPPA, (b) the critic aspects of GHOST, and (c) the elaboration aspects of SIPEC [Kartam 89].

The representational aspects of PIPPA:

> To summarize there are three rules which define the initial stages in the construction of project models.
>
> - An activity is considered to be an action on an object, the particular action and object has to be identified for each activity and drawn using the notation of activity node with "objects" and "action" links.
> - The objects form a hierarchy; the highest object is shown connected to its subcomponents via "sub" links, and then their subcomponents etc. Each object in the hierarchy can be physical or conceptual.
> - The actions form a hierarchy in a similar way to objects.
>
> [Marshall 87 p.290]

The critic aspects of GHOST:

> The more interesting case is when the general network does not meet duration constraints (deadlines) so that there is a need for overlapping activities to meet such constraints. For this case, GHOST picks a pair of general activities on the critical path and substitutes subnetworks. [Navinchandra 88 pp. 246-247]

The "object" hierarchy represented in OARPLAN is divided into two branches: simple objects (e.g. footing, beam, column) and compound objects (e.g. slab, wall, floor). Actions are assembled into a similar hierarchy.

> OARPLAN starts with a high-level activity such as <construct> <building-1> at the first level of the plan. [Darwiche 88 p.175]

This activity is then elaborated. Dependencies between activities are based on "logical" and relationship constraints. These relationship constraints are used in OARPLAN to modify precedence.

The use of relationships as a basis for constraints is very powerful as implemented by Kartam [89 pp.66-95] and Navinchandra [88 p.245]. The use of interacting hierarchies

[Waugh 88a p.II-7] adds an additional dimension and significantly reduces memory requirements and speeds retrieval time [Waugh 89a p.6] over many other possible storage arrangements.

> In summary, OARPLAN infers dependencies in two ways. The first is utilizing predefined dependencies that are inherited from sub-plans, such as placing concrete. These are of the type that are constant across projects and about which little reasoning is needed. The second is by inferring dependencies through applying dependency [knowledge sources] which reason about the constituents of activities. Since these vary across projects as a function of the objects, actions and resources of a given project, extensive project-specific data are needed for each of the plans that OARPLAN produces. [Darwiche 88 p.179]

Four pieces of knowledge (constraints) are noted [Darwiche 88 pp.178-179]:

- Supports Constraint: If activity-1 is <place> <member-1> and activity-2 is <place> <member-2> and <member-1> supports <member-2> then constrain activity-1 to be performed before activity-2.
- Safety Constraint: In steel-framed buildings, do not start work on the members of floor n until the slabs of floor n-1 or floor n-2 are constructed.
- Interior Wall - Slab Constraint: do not start constructing a wall until all (one or both) of the floor slabs adjacent to it are constructed.
- Interior Wall - Exterior Wall Constraint: within a given floor do not construct interior walls until all external walls have been constructed.

Ito [89] has implemented an interface for the AutoCAD [88] system called CIFECAD. Using building construction drawings, it extracts project assemblies (SLABS, COLUMNS, and WALLS) and relationships (SUPPORTED-BY, CONNECTED-WITH). The relationship output from this interface is of the following form:

```
(SUPPORTED_BY GIRDERS01 COLUMNS02 COLUMNS01)
(SUPPORTED_BY SLABS01 GIRDERS10 GIRDERS07)

(CONNECTED_WITH EWALLS01 COLUMNS02 COLUMNS01)
(CONNECTED_WITH EWALLS02 COLUMNS03 COLUMNS02)

etc.
```

CIFECAD has been used in conjunction with OARPLAN and SIPEC. The generation of precedence aspects of OARPLAN and SIPEC are then utilized since the output of CIFECAD essentially contains the precedence knowledge which they use; in addition, the elaboration aspects of these planners are employed.

Since all of the above planners are based on the network representation (two-stage treatment of non-resource and resource constraints), they have many of the same limitations as those noted in Section 2.2.

2.3.5. ISIS

In his pioneering AI scheduling application, Fox [87] uses constraint-directed reasoning in the job-shop scheduling domain. The scheduling problem is viewed from a constraint-directed search perspective with five categories of constraints:

- Organizational goals: due to date requirements, work-in-process time requirements, cost restrictions, and machine utilization goals.
- Physical limitations: machine capabilities, product size and quality limitations.
- Causal restrictions: precedence of operations, and resource requirements to perform an operation.
- Availability: availability of resources (e.g., tools, fixtures, NC programs, and operators) to perform an operation.
- Preferences: qualitative preferences for operations, machines, and other resources [these constraints are subsequently referred to as an abstraction of other types of constraints].

[Fox 87 p.1]

Fox expands this view of constraints to include reasoning about operations to perform and when to perform them. He notes the need for the following types of constraint knowledge:

- The range of durations of the operation, including a probability density function.
- The operations which may precede or follow the current operation.
- The resources required: materials, tools, fixtures, software, etc.
- The period of time during which the above resources are required.
- The transformations applied to the resources. For example, is the cutting fluid on a milling machine totally consumed?
- Any constraints on the usage of the resources.
- The operator (i.e., who may perform the operation?).
- Substitutability of resources. If a machine is not available, can another be used?
- A description of how the operations are performed. What are the components (i.e., suboperations comprising the operation)?

[Fox 87 p.25]

As will be seen in following chapters, there are many similarities between the above constraints and the constraints used in construction scheduling. Fox's paradigm exploits constraints to the limit by using them as the primary method of reasoning and of representing knowledge.

This research uses a hybrid approach of combining knowledge and constraints in a procedural model.

2.3.6. CONSTRUCTION PLANEX

CONSTRUCTION PLANEX [Zozaya 88] is the first expert system to address the full spectrum of construction planning, from the derivation of project components to the generation of a time scaled activity sequence. The authors describe the scope and limitations of CONSTRUCTION PLANEX as follows:

> "The current version of CONSTRUCTION PLANEX has some limitations with respect to the manner in which the project network is generated:
>
> - CONSTRUCTION PLANEX generates plans using detailed information about the structural elements of a building (e.g., dimensions, type of materials, etc.). No knowledge has been incorporated to generate plans from more aggregate building information.
> - The system generates only those activities associated with the construction of design elements. For those activities that are not directly related to building components (e.g., site clearing), artificial design elements have been created (e.g., an object describing the geometric characteristics of the site). Future versions of the systems may include additional domain operators that generate project activities from other project activities.
> - CONSTRUCTION PLANEX first identifies the activities to be performed and then establishes precedences among them. The system could have other type of [knowledge sources] or operators that perform these tasks simultaneously.
> - In CONSTRUCTION PLANEX, activities were divided into two levels of aggregation: (1) element activities representing activities on individual design elements, and (2) project activities representing groups of element activities on a particular floor of the building. Future versions of the system should include more sophisticated schemes for spatial aggregation of the construction activities.
>
> Future work on activity generation and aggregation models may overcome these limitations."
>
> [Zozaya p.169]

"Establishing precedence" in the above quotation refers to the model accessing predefined precedence relationships among activities. PLANEX includes knowledge sources which have been implemented for other reasoning tasks, but knowledge sources have not been employed to generate the ordering of activities.

This research emphasizes generating the ordering of project activities.

2.3.7. Echeverry's Proposal

Echeverry [89] proposes an interesting construction planning model for use in mid-rise building construction. He notes [p.225] the following constraints which he obtained from field research and references [Gray 86; Levitt 88; Navinchandra 88; Zozaya 88]:

- Functional relationships
  supported-by
  covered-by
  weather-protected-by
  embedded-in
- Trade interaction
  occupies-same-space-as
  provides-services-to
  may-damage
  affects-environment-of
- Resource limitations
  forces non-parallelism
- Code regulations
  metal deck must be within two floors of steel erection

Four knowledge modules applied in a blackboard architecture are also proposed [Echeverry 89 p.226].

To convert building systems into building components
To convert "construction of the building" into activities
To apply activity logic
To allocate crews

As noted previously, this research implements arbitrary constraints. This research will not convert building systems into building components as done by CONSTRUCTION PLANEX and proposed by Echeverry.

2.4. SUMMARY

This review is intended neither to belittle the major contributions to scheduling which network diagraming techniques have offered nor to underestimate the advances which knowledge-based construction planners have made in such a short period; the purpose is to point out several opportunities for further improvement. The following is a summary of the assumptions, representation, reasoning, and emphasis of this research:

Assumptions:

- *This research relies on deterministic knowledge to develop schedules.*
- *This research will maintain redundancy among constraint types.*
- *This research will produce a schedule with all activities in parallel if no constraints are imposed on the system.*

Representation:

- *This research allows the user to represent arbitrary constraints.*
- *This research explicitly represents resources.*
- *This research uses hierarchies as a method of adding class and relationship knowledge to elements such as project components and resources.*
- *This research attempts to represent more knowledge in the construction planning domain, more robustly, and more flexibly.*

Reasoning:

- *This research identifies and creates activities using a straight-forward mapping to project components.*
- *This research uses a hybrid approach of combining knowledge and constraints in a procedural model.*
- *This research implements resource constraints.*

- *This research treats resource and non-resource constraints on an equal footing, i.e., all project constraints are considered concurrently.*

This research offers a query facility through which the user can determine which constraints were applied in generating a schedule.

Emphasis:

- *This research emphasizes generating the ordering of project activities.*
- *This research regards the interaction between the scheduler and the scheduling model, to be of ultimate importance.*

The above items summarize the results of the literature review and provide a foundation for the model. As this chapter has indicated, there are a variety of approaches to scheduling; the choices among these approaches are best made by considering the characteristics of the domain--construction scheduling. The following chapter elucidates several of these choices which begin to define the research approach and thereby extends the foundation for the model. Of particular importance in the next chapter are the choice to represent knowledge at a fundamental level and the choice to use iterative, chronological search.

3. RESEARCH ASSUMPTIONS AND APPROACH

This chapter documents aspects of the research approach which address three characteristics of the construction industry—problem representation, the additive nature of construction activities, and subgoaling versus chronological reasoning. These characteristics are not specifically documented in the literature.

As noted in Chapter 2, a concise but robust representation is very important. The additive nature of construction influences the choice between an opportunistic or a constraint-based approach. An iterative, chronological reasoning structure was selected due to the representational deficiencies of subgoaling techniques.

3.1. PROBLEM REPRESENTATION

3.1.1. Assemblies, Methods, and Crews

PIPPA [Marshall 87] implements object and action hierarchies to define activities; the use of a resource hierarchy to extend this definition of an activity is a natural extension [Darwiche 88 p.174; Waugh 88a p.II-7]. A further extension to this is the grouping of hierarchy branches. It is useful to refer to specific portions of a hierarchy by name and to have a formal representation for these in the model. In a previous publication [Waugh 89c p.4], I suggest the following element and hierarchy names:

Element	Hierarchy
components	assembly
action	method
resource	crew

Each of these terms is defined in the Glossary.

This research suggests the definition of an action as a combination of an assembly, a method, and a crew.

3.1.2. Reasoning

A richer representation of non-resource constraints ensures a more robust model. A single class of precedence constraints alone is not enough; I described in Chapter 2 why constraints must be explicitly based upon facts and knowledge about the project. By providing the model with more knowledge about the project components, the resources, the constraining rules, the activities, and the priority of activities, a richer representation can be attained. The following list provides some examples of construction scheduling data and knowledge:

Project component data
- the size of footing X
- footings adjacent to footing X
- components which provide gravity support for column X
- components which enclose stud X
- completed components

Resource data
- carpenters assigned to the project
- carpenters now available
- resources which belong to crew X

Constraint knowledge
- activities completed by ABC (a subcontractor) should be done together
- activities that install components for which no gravity support is available should not be scheduled

Activity data
- activities remaining to be scheduled
- activities not constrained by non-resource constraints
- activities not constrained by resource constraints
- activities currently being executed
- activities already completed
- activities just completed
- the time period(s) when activity X was scheduled
- activities which immediately constrain activity X

Priority knowledge
- activities which construct structural components should be given high priority

- activities which construct landscaping components should be given low priority
- when one activity for which XYZ (a subcontractor) is responsible is scheduled, the priority of other activities performed by XYZ should be increased

The most obvious characteristic of a system which uses these kinds of knowledge is its need for facts; this is typical of knowledge-based expert systems. The key to *this* model is not just arbitrary knowledge, but knowledge which constrains the ordering of activities in *fundamental* ways.

Often emperical sequencing knowledge is used to force an ordering of activities. Suppose, for example, that a one-story window wall is designed to span vertically between two concrete floor slabs. The mullions of this window wall are laterally and vertically supported by the slabs above and below.

The usual network diagraming approach is simply to state that "the slabs must precede the window wall."

The following constraint captures a more fundamental basis for ordering activities: "support (lateral and vertical) for the window wall is required prior to its installation." This is an improvement over the network diagraming approach from a representation point of view because it relies on, and explicitly states a more fundamental concept—support. This constraint could be met by the slabs or through temporary means.

Another example of a constraint which is necessary to represent reality more rigorously is, "to avoid breakage, the window wall should follow structural activities above or be protected from falling objects."

The latter two rules, taken together, are more *generic rules* which draw on *specific facts* about the project and capture the fundamental knowledge which controls the sequencing of the activities.

Fundamental information and knowledge about attributes, relationships, status, and sequencing knowledge are prevalent in construction. This research emphasizes knowledge and constraints that influence the ordering of activities at a fundamental level.

3.2. ADDITIVE NATURE OF CONSTRUCTION

Historically, on-site construction included a great number of shaping and moulding activities [Waugh 88a p.IV-5]. Prefabrication of components has decreased this work considerably since the 1960s. Today's construction activities are primarily additive as indicated by the definition of the word *construct* in the World Book Dictionary:

> to put together; fit together; build; frame [77 p.445]

Surely there are construction activities which involve a removal stage. Scaffolding and concrete formwork are examples of temporary components which must be removed prior to project completion, but temporary components are the exception rather than the rule in construction. Masonry and wood are often cut to size on site, but most schedules do not address construction projects at this level of detail since activities are defined at a more abstract level than individual saw cuts. With more and more prefabrication being conducted off site, examples of on-site removal activities are less and less common.

This additive nature of construction has three simplifying effects for a construction scheduling model.

- There is a straight-forward mapping between project assemblies and methods. Contrast this with the types of problems on which AI planning models are routinely tested: the blocks world, the nine puzzle, chess, etc. Here considerable effort is required to match an operator (action or method) to an object (component or assembly).

- Minimal manipulation of each assembly is required in construction. Again, compare this with typical AI problems where each object (assembly) is manipulated many times. Task level planning for repetitive construction is a special case where different assemblies can be treated in the same way. In these situations, means-end AI techniques may prove useful [Levitt 87].

- Finding an action which will help reach the goal is far simpler (but more knowledge intensive) in construction. This consideration is related to "legal" versus "illegal" actions. In AI terminology, a legal action is one which the model can execute without violating its assumptions; e.g., it is usually considered illegal in the blocks world for a robot arm to pick up more than one

block at a time. Correspondingly in construction, it would be considered illegal to start work on the second floor of a building prior to providing gravity support for that floor (e.g. the first floor). In typical AI domains, an arbitrary choice from within legal activities has a low probability of leading toward the goal. In contrast, the additive nature of construction implies that a choice from within legal activities (of which there may be fewer) has a high probability of leading in the direction of completing the project. There are some situations where a legal construction activity may not lead toward the goal (e.g., closing in a building prior to moving in large equipment, such as a boiler), therefore, this concern is not negligible. However, from the arguments presented above, I assert that the probability of moving toward the goal with an arbitrary legal activity is far higher in construction than it is in typical AI planning problems.

3.3. SUBGOALING VERSUS CHRONOLOGICAL REASONING

Although the models described in Section 2.3. are not solely backward chaining, several use a kind of backward chaining called subgoaling to develop plans. Purely forward or backward chaining models operate on a basic program structure (often called rules) of the form:

IF premise A, THEN conclusion B

IF premise C, THEN conclusion D

etc.

In order to ask the model a question, the user poses a goal which the model attempts to establish by checking the validity of all of the goal's premises. Each premise can then be set up as a subgoal and the systems look for rules with the subgoals as a conclusion. That rule's premises are then established as new subgoals, and so on. See any of the following texts for a detailed description of forward and backward chaining: [Dym 91; Harmon 85 pp.53-57; Rich 83 pp.56-57; Walters 88; Waterman 86 pp.66-69].

Three factors influence the question of whether it is better to reason forward or backward:

- Are there more possible start states or goal states? We would like to move from the smaller set of states to the larger (and thus easier to find) set of states.

- In which direction is the branching factor (i.e., the average number of nodes that can be directly reached from a single node) greater? We would like to proceed in the direction with the lower branching factor.
- Will the program be asked to justify its reasoning process to a user? If so, it is important to proceed in the direction that corresponds more closely with the way the user will think.

Rich [83 p.58]

Backward chaining has proven to be very useful for diagnosis problems. In construction the goal is to complete the project. The permanent project components are well described and the initial situation (state) is evident. Due to the additive nature of construction, there is a close correlation between the activities and the project components. This results in there being an alternative and easily defined goal, the completion of all activities. Generating a project network is essentially a problem of ordering activities sequentially in time.

Compare Rich's three factors with the following characteristics of construction scheduling.

- With respect to time there is only one start state—the present, all future states are estimates. The end (or goal) states include an infinite number of possible conditions, when combinations of resource consumption, management, weather, etc. are considered.

- Branching possibilities occur as new knowledge is gained; this happens as time passes. By starting at the beginning of the project, all time variables can be updated, e.g., seasons, financial state of the contractor, etc.

- When taught scheduling or estimating, humans are told: "build the project in your mind." It is difficult for a person to think of a building being constructed while moving backwards in time.

An additional benefit of moving forward in time is the ability to back up without wasting the effort already expended; this may be particularly useful for monitoring purposes. If the model imposes an erroneous constraint in the twelfth week of a project, the user can stop the model, back up, and start again from the eleventh week. In a system which moves backward in time the model must be reinitialized and started again.

This research follows a procedural approach which generates the sequence and timing of activities by searching forward in time.

The previous chapter described the foundations for the model. This chapter justified three basic choices for the research approach. The following chapters describe the model itself.

4. ACP OVERVIEW

I have reviewed the literature in Chapter 2 and in Chapter 3 I have indicated some basic assumptions upon which the model is based. This chapter provides an overview of the model.

Network diagraming techniques (NDTs) are the most widely used methods for planning and scheduling in the architecture, engineering, and construction industry. However, many researchers and practitioners find them to be inadequate tools for the task.

> After more than two decades of applying CPM as a planning and scheduling technique there is growing doubt about some of the advantages initially attributed to it. It is being discarded by both large and small construction companies, and its use limited to cases where it is required by the clients. [Allam 88 p.93].
>
> Vital ingredients [to the schedule] cannot be and are not expressed in CPM. [Birrell 87 p.345].

I contend in Chapter 2 that the capabilities of NDTs are restricted by limitations inherent in their representation of schedule constraints (typically expressed through precedence relationships between activities). This assertion suggests that a richer representation will extend the utility of planning and scheduling techniques. I suggest that such a representation is provided by a system which employs a general model of a project's status and which allows schedule constraints to be expressed as rules which refer to this status. This system offers the advantages of allowing the precedence of activities to be based on more fundamental states or events than just the completion of other activities. It also utilizes an efficient knowledge-based approach to scheduling that can express the reasoning underlying scheduling actions.

The following section discusses how precedence and resources are represented in NDTs, the next section discusses how constraints are represented and what types of constraints can be represented in the proposed system—called A Construction Planner (ACP). This is followed by a comparison of the schedule generation algorithms used in NDTs and in ACP.

4.2. NDT Precedence and Resource Representation

All NDTs employ a common representation of schedule constraints using precedence relationships of the form "activity A can start as soon as activity B is finished." In activity-on-node representations, an activity can begin when all activities which have precedence links leading into that activity are complete. In activity-on-arrow representations, an activity can begin after the completion of all other activities which have that activity's start node as their end node. Variations to this precedence representation include the use of zero duration activities such as dummies and milestones; the use of different classes of relationships such as start-to-start, finish-to-finish, and start-to-finish; the use of lag factors related to any of the relationships just mentioned; and the use of probabilistic duration representations. However, a fundamental assumption of NDTs remains:

Precedence for an activity is based on the status of completion (from start to finish) of other activities.

The consequences of this assumption is that there is no opportunity to represent many schedule constraints which frequently arise in construction, but which are not directly translatable into precedence relationships between activities. The following examples typify such constraints:

- Gypsum wallboard should only be installed if there is protection from precipitation and if the temperature is above 10 degrees Celsius.
- Concrete should be placed on Fridays (to allow the weekend for curing).
- If there is an ample supply of labor and labor conditions are stable, then fabricate finish carpentry on-site, otherwise prefabricate.

A second problem that arises from NDTs representation scheme is that precedence links convey no information about the reasons and assumptions underlying their inclusion into the network. This has significant implications, both in terms of failing to convey critical details of the construction plan to those who must carry it out, and when subsequently updating and modifying the plan to reflect changed project conditions. For example, one cannot tell by examining a precedence link between two activities whether it exists because the first activity provides structural support for the second activity, or whether it was included because work on the first activity is potentially damaging to the results of the

second activity (such as painting before carpeting). The latter constraint could be removed by providing additional protection if time were of the essence (see Section 2.2.7.).

As I have shown, these problems stem from the representation of schedule constraints as precedence links between activities. However, many NDTs do allow the representation of a second type of schedule constraint—resources.

NDTs adopt a two-stage approach to resource constraints [Antill 82 p.43; Cormican 85 p.70; Moder 83 pp.14-17; and Tamimi 88 p.289]. In the first stage, the precedence relationships are defined; while in the second stage, resources are explicitly incorporated and their impact on the schedule is calculated using resource leveling or allocation algorithms. This results in the second fundamental assumption of NDTs:

Resource constraints are introduced into the schedule using a two-stage approach.

See Section 2.2.6. for further explanation. Assuming then, that a two-stage approach will be used, two alternatives remain:

1. Ignore resources in the first stage.

2. Introduce resource constraints in both stages.

Both alternatives pose difficulties. [Stevens 90 p.13] and others [Antill 82 p.161] clearly state that resources should be ignored during the first stage. However, I contend that realistic construction activity networks cannot be developed without considering resources (see Section 2.2.4.). In scheduling practice, varying the available resource assumptions (i.e., changes to the types of available resources or significant variations in the resource quantities) usually requires revisions to the network's precedence relationships.

> With basic CPM networks it is difficult to distinguish in logic between technological constraints and resource constraints. [Barrie 84 p.237]

An unrealistic but illustrative example of assuming unlimited resources is also given in Section 2.2.4. Alternative 1, then, seems unsuitable for realistic planning and scheduling. A more likely option is to implicitly assume resources and to incorporate these assumptions in the precedence constraints. However these implicit assumptions are not documented in the network and, as I have discussed above, this leads to problems in conveying the plan to others and in subsequently updating the plan.

This brings us to the second alternative—explicitly introducing resource assumptions in both stages. Here an iterative trial-and-error process must be followed. The scheduler assumes resource types in the first stage (in order to determine methods upon which precedence relationships are based), assumes resources quantities in the second stage (in order to run resource leveling or allocation techniques), then considers the resulting schedule, and repeats with modification until all constraints are met.

The task of manually recording and coordinating these assumptions becomes onerous on a project of practical proportions. For large projects which are scheduled using conventional heuristic algorithms on a computer, the scheduler does not receive the necessary feedback, resulting in a black-box approach [Fondahl 62; Lester 82]. Here the user tends to focus on modifying the resource quantities and ignores the resource selection decisions. This leads to an unnatural and overly restrictive planning tool.

This second approach is more correct in theory but more difficult in practice—thus it is rarely employed in industry. The result is that NDTs normally lead to two separate sets of assumptions about resources, with no record of the first set and no automated coordination mechanism between the sets.

ACP uses a single stage approach which explicitly accounts for all constraints simultaneously, including resources.

4.3. ACP Constraint Representation

The constraints represented in NDTs—activity precedence and resources quantities—are a subset of all real project constraints. Simple and elegant methods of representing this subset have been found and used successfully within NDTs, but advances in computer science (particularly, those related to symbolic processing and AI) have provided the opportunity to extend this approach to include a richer representation. I suggest that the documentation of a project's current state (labeled STATUS in the ACP system) solves many of the representation limitations which have plagued NDTs since their inception. First I will describe how activity precedence and resource constraints can be represented as constraints which draw on the project STATUS and then I will describe how additional constraints can also be represented in this extended paradigm.

Suppose that a project P entails the erection of the structural steel for a warehouse. The reason why a specific steel member can or cannot be installed is embodied in the following rule:

If

the members which provide gravity support for member X do not exist in the STATUS

then

member X cannot be started.

This rule draws not only on a documentation of what the status of the project is at all times, but it also requires information about the project components and their relationships. Fortunately, it is possible to obtain the project components and their relationships directly from the computer-aided design. A successful prototype for this has already been developed by [Ito 89]. The result is that one rule which draws on previously entered information (the structural load path) and documentation of STATUS will replace all gravity support precedence lines for project P.

Similarly, one rule can replace many resource constraints:

If

activity X consumes Q units of resource R

and

less than Q units of resource R are available

then

activity X cannot be started

Furthermore, many other types of constraints can be represented. The examples of constraints listed above which can not possibly be represented as activity precedence (the weather, the day of the week, the current labor conditions) can be incorporated directly once the appropriate information is documented in the project STATUS.

Information added to STATUS includes the effects of ENDED activities, such as the statement that project component X is now complete. Cumulative effects (milestones) are also recorded in STATUS using rules such as:

If

the roof is complete
and
the walls are complete
and
the glazing is complete

then

a controlled environment exists.

The result is a concise representation for a more general perspective of project planning constraints.

4.4. SCHEDULE GENERATION

After the first activity is complete, the algorithm used by NDTs for the forward pass of a project network is:

1. Consider all activities whose predecessors are complete.

2. Assign to the start time of each activity the maximum finish time of its preceding activities.

3. Add the duration of these activities to their start times and assign this to their finish times.

Loop until no activities remain.

Just as the representation for NDTs is simple and elegant, so too are the timing calculations. The algorithm for ACPs timing calculations is illustrated in Figure 4.1. Initially, it is assumed that all activities remain to be done, represented by their inclusion in the TODO list. As the model cycles toward a solution and activities are slated for completion, they are incrementally transferred through the five activity categories to the DONE list. Meanwhile, the STATUS is updated each cycle and is used by constraint rules, such as those noted above, to determine which activities can be cycled toward DONE. Three knowledge bases (rules) are utilized: CANDO KNOWLEDGE which determines if it is possible to begin an activity; ASSIGNMENT KNOWLEDGE which selects from within

the CANDO activities; and STATUS KNOWLEDGE which updates the STATUS due to the effects of completed activities and other states or events such as weather, labor availability, etc.

The algorithm used by ACP to generate a schedule is described by the following six steps:

1. The CANDO KNOWLEDGE is applied to each of the activities in the TODO list. If the CANDO KNOWLEDGE rules are satisfied (i.e., the activity can be started), then the activity is moved to the CANDO list. This determination is based on the contents of STATUS. A gravity constraint is an example of CANDO KNOWLEDGE.

2. The ASSIGNMENT KNOWLEDGE is applied to each of the activities in the CANDO list. This knowledge prioritizes the CANDO list and then moves to the DOING list the activities which satisfy all ASSIGNMENT KNOWLEDGE rules. Resource constraints are an example of ASSIGNMENT KNOWLEDGE.

3. The project clock is incremented by one time period. The remaining execution time of all activities is decreased by one. All activities with a remaining execution time of zero are assigned to ENDED.

4. The effects of ENDED activities and project milestones are added to the STATUS by the STATUS KNOWLEDGE.

5. ENDED activities are moved to DONE.

6. If TODO, CANDO, and DOING are empty, then stop; otherwise repeat from step one.

The algorithm is summarized in the following schematic.

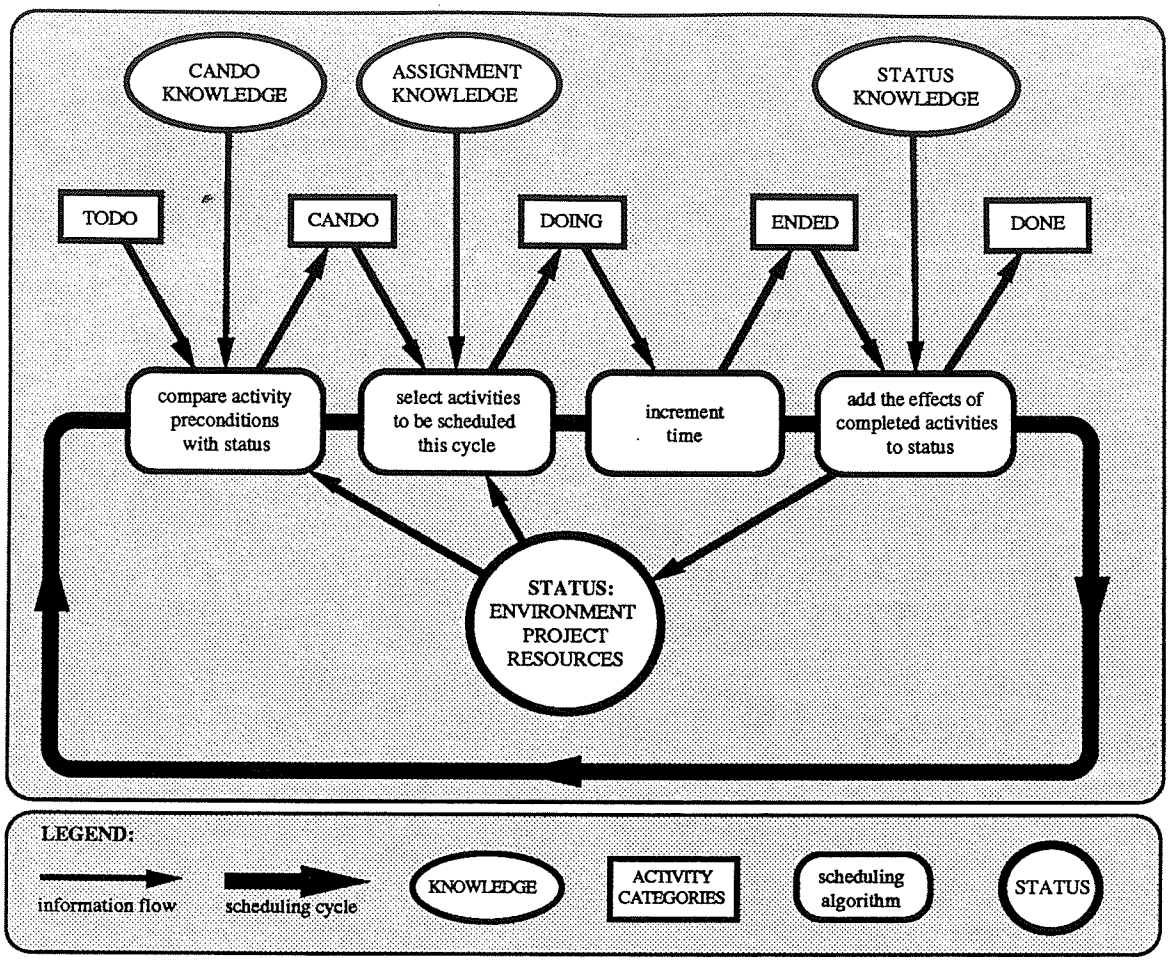

Figure 4.1 ACP's Algorithm and Representation

This scheduling algorithm is more complicated than the one used in NDTs, but gains from the ability to represent far more constraints and to query the system about the rules which were applied.

ACP does not represent the backward pass calculations which are performed in NDTs. The reason for this is that the earliest completion date is unknown at the time activities are being assigned. Further research is required to determine whether concepts such as lag and float have a practical application when constraints other than activity precedence are considered. One possibility is to use historical data on typical float times for classes of activities; e.g., if elevator installation activities are historically critical, then they should be assigned a high priority.

A major benefit of ACP, which will be discussed in Chapter 7 and demonstrated in Chapter 8, is the ability to query the system about why an activity was not scheduled at a

specific time. This ability allows the scheduler to monitor the application of the planning process clearly and identify the schedule rationale. The lack of this information is one of the major drawbacks of current NDTs when employed with resource constraints—the user is not aware of when and where individual constraints (heuristics) influence the schedule.

As will be seen in the following chapters, the goal of representing more of the vital ingredients to planning has been attained. These extensions have resulted in some complications over traditional NDTs, but methods of dealing with these extensions have already been found or are currently being developed. The result is a concise representation, a more robust model of planning, and an opportunity to take advantage of advancing computer technologies. Furthermore, ACP more closely and explicitly reflects the methods followed by human planners.

This chapter has given an overview of ACP. Chapter 5 discusses the input required for ACP and Chapter 6 discusses activities. Chapter 7 provides a more detailed explanation of how ACP generates schedules. Chapter 8 applies ACP to schedule a construction project.

5. INPUT

This chapter describes the three modules of the system which are used to input the project description, resources, and scheduling knowledge. Refer to Chapter 4 for an overview of the system and to Appendix 2 for a schematic of the system menus.

5.1. PROJECT DESCRIPTION

A project is described by specifying a) the physical components which constitute the completed facility and b) the relationships between these components.

The system is designed with the intent that, at a later date, the component list and the relationships between components will be obtained from a computer-aided-design database as implemented by Ito [89] for AUTOCAD.

In ACP each component name is typed in by the user and is stored in a list. In order to enter relationship information, the user must specify the relationship type (e.g., gravity-support or simply *supports*) and a project component (e.g., Footing_X). The user then selects from a menu any component(s) which *supports* Footing_X and any component(s) which *is-supported-by* Footing_X.

Each relationship is stored in a square boolean matrix with dimensions equal to the length of the component list. Ones in this matrix indicate a relationship between the components. Figure 5.1 shows a *supports* relationship matrix with its corresponding component list. Entering a relationship matrix from the left, indicates the primary relationship; entering the matrix from the top indicates the inverse relationship.

In Figure 5.1, the primary and inverse relationships are *supports* and *is-supported-by* respectively, e.g., FOOTING_A1 *supports* COLUMN_A1 and SLAB_A1B2_2 *is-supported-by* the four beams. I use the term *child* to refer to the primary relationship and *parent* to refer to the inverse relationship. To find the *child* relationship, you enter the relationship matrix from the left; to find the *parent* relationship, you enter from the top; e.g., COLUMN_A1 is a *child* of FOOTING_A1 and FOOTING_A1 is a *parent* of COLUMN_A1. If the children are the same as the parents, as is the case with any two-

directional relationship such as *adjacent-to*, then the matrix will be symmetric (around the leading diagonal).

When defining a new project there are two approaches: a) start with an empty project description module and enter the new project component list and relationships, or b) enter an existing project description and delete or revise unwanted project-specific information and then input additions. In either case the workspace should be saved with a new name.

5.2. RESOURCES

Elemental resources include workers, equipment and materials. Although the representation which is used allows for materials, emphasis is placed on workers and

	EARTH	FOOTING_A1	FOOTING_B1	FOOTING_A2	FOOTING_B2	COLUMN_A1_1	COLUMN_B1_1	COLUMN_A2_1	COLUMN_B2_1	BEAM_1AB_2	BEAM_A12_2	BEAM_2AB_2	BEAM_B12_2	WALL_1AB_12	WALL_A12_12	WALL_2AB_12	WALL_B12_12	SLAB_A1B2_1	SLAB_A1B2_2
EARTH	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0
FOOTING_A1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0
FOOTING_B1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0
FOOTING_A2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0
FOOTING_B2	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
COLUMN_A1_1	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0	0	0
COLUMN_B1_1	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	1	0	0
COLUMN_A2_1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	0
COLUMN_B2_1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	0	0
BEAM_1AB_2	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
BEAM_A12_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
BEAM_2AB_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
BEAM_B12_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
WALL_1AB_12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
WALL_A12_12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
WALL_2AB_12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
WALL_B12_12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SLAB_A1B2_1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
SLAB_A1B2_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 5.1 Relationship Matrix

equipment. This emphasis was chosen because workers and equipment dynamically affect the schedule. Permanent materials can be accounted for by adding a procurement activity prior to scheduling. Temporary materials have dynamic effects similar to workers and equipment.

For each elemental resource, the user enters its

- name,
- resource class (workers or equipment),
- quantity available, and
- unit of measurement.

Figure 5.2 shows two examples. Note that the unit of measurement, EACH, is taken from construction estimating terminology.

LABORERS	PILE_DRIVER
WORKER	EQUIPMENT
10	1
EACH	EACH

Figure 5.2 Typical Elemental Resources

A crew is a selection of one or more elemental resources or other crews. A crew also has production and consumption rates. Figure 5.3 shows the representation used for a pile-driving crew. The lower-case words in Figure 5.3 are common to all crew representations; the upper-case words and numbers are specific to the pile-driving crew.

When a crew is initialized, the user is asked the crew name, production information,

name:	CREW_PILES		
production:	5		
unit of measure:	PILES		
unit of time:	DAY		
resources:	quan	unit	name
workers:	1	EACH	PILE_DRVR_OPER
	2	EACH	LABORERS
equipment:	1	EACH	PILE_DRIVER

Figure 5.3 Typical Crew

and the associated resources. The production information in Figure 5.3 is read as "5 PILES/DAY." Each elemental resource is selected from a menu of available resources. The crew is inserted on the appropriate line automatically by the program using the resource class. The unit of measurement is also available from the resource, but the quantity of the resource to be consumed by this crew needs to be entered by the user.

Relationships between elemental resources and between crews are each stored in a relationship matrix. Facilities for creating elemental resources, assembling crews, and editing the results are implemented.

The same procedure should be followed when creating new resource modules as described above for new project description modules.

5.3. SCHEDULING KNOWLEDGE

The scheduling knowledge module holds a number of user-defined knowledge bases. Each knowledge base contains STATUS knowledge and CANDO knowledge; the purpose and function of these two categories are briefly described in Chapter 4.

Both STATUS and CANDO knowledge are composed of English-like rules. These rules use a restricted vocabulary of words. The vocabulary includes a) system function words and phrases, b) system variable names, c) activity names, d) project component names, e) user-defined variable names, and f) user-defined words and phrases. Figure 5.4 lists the vocabulary.

All system function words and phrases are converted into APL functions. The syntax and effect of these system functions are evident from their use in Chapter 8. All other words in the vocabulary are used as arguments to these functions.

The following is an example of a situation where a user-defined variable and a user-defined phrase are useful. The user wants to a) create a rule that constrains the interior electrical activity from beginning until the building envelope is complete and b) create a rule that adds to the STATUS the phrase "Controlled Environment" when the building envelope is complete. But rather than list all the building envelope components in both of the rules, the user chooses to define the variable BUILDING_ENVELOPE. This is done by creating a variable with this name and entering into this variable the appropriate list of project

components, e.g., wall, roof, and window components. This will allow the following rules to obtain the desired result:

If

{the} STATUS does not contain all BUILDING_ENVELOPE {components}

then

delete from NEWCANDO 'Const_Interior_Electrical'

System function Words and Phrases:

If	then
which do not meet	which meet
precedes	delete from MILESTONES
delete from NEWCANDO	delete from CANDO
add to MILESTONES	add to STATUS
support for	cando test
priority increase by	is a member of
is assigned to	is assigned
does not contain only	does not contain all
does not contain	contains only
contains all	contains
there exists	also
less than	not greater than
equals	not less than
greater than	not equal
not	or
nand	and
Comment	End

System Variable Names:

TODO	NEWCANDO
CANDO	DOING
ENDED	DONE
STATUS	

Activity Names

Project Component Names

Crew Names

User-Defined Variable Names

User-Defined Words and Phrases

Figure 5.4 Vocabulary

If

{the} STATUS contains all BUILDING_ENVELOPE {components}

then

add to STATUS 'Controlled Environment'

The above example uses the *if-then* rule format and indicates the opportunity to include within rules, nonexecutable text for clarity by delimiting this text by parentheses. The *if-then* format is the most general rule type.

A second rule type uses the phrase *which meet* or the phrase *which do not meet.* The format for these functions is

Activity_List_Variable_Name *which do not meet* 'Condition'

NEWCANDO is typically used as the Activity_List_Variable_Name, but the functions work equally well with any other activity list variable name. The Condition must be enclosed in quotes, must contain the variable "X" to be instantiated with each of the activities, and can be any statement which evaluates to true or false (1 or 0). The function *which meet* evaluates the condition for each of the activities in the list and returns as a result a list which only includes those activities which meet the condition. The function *which do not meet* similarly evaluates the condition, but returns as a result only those activities which do not meet the condition. The following *supported_by* rule provides further explanation:

delete from NEWCANDO

(NEWCANDO which do not meet 'STATUS contains all support for X')

The function "support for" finds the project components which are parents of the component which activity X installs (these are found in the Supported-By slot of activity X). Then the function "contains all" checks if STATUS contains all these supporting components. If the condition is not met, then the activity name is recorded. After all activities in NEWCANDO have been considered, the part of the rule within the parentheses is complete. (In this case, the parentheses are used for explanation purposes, but they also can be used to control evaluation.) The function "delete from NEWCANDO" now deletes the recorded activity list from NEWCANDO, indicating that those activities whose components do not have gravity support should not be scheduled.

The final rule type uses the word *precedes* in the following format:

'Activity_A' precedes 'Activity_B'

This notation is a simplification of an *if-then* rule. The effect of the above rule is the same as the effect of the following rule:

If
    DONE does not contain 'Activity_A'
then
    delete from NEWCANDO 'Activity_B'

A knowledge base is made current (within the workspace) by selecting it from a menu. Deletions, additions, and revisions can be made to the most recently retrieved knowledge base. Then it can be reassigned to its old or a new name for future retrieval. The knowledge base which exists in the Schedule module can also be copied, assigned a name, and recorded. This approach allows simultaneous access to all knowledge bases without changing workspaces.

This chapter has shown how the user inputs project components, their relationships, resources, and scheduling knowledge; and has shown how this information in represented within ACP. With this information ACP can generate activities—the subject of the following chapter.

As noted earlier, current research indicates that much of the component and relationship information will soon be retrievable via communications links to programs such as AUTOCAD. Much of the remaining input—resources and scheduling knowledge—will be typical within a wide range of construction projects. These points will significantly reduce the quantity of input on subsequent projects without losing the knowledge-based advantages.

6. ACTIVITIES

The Activities module performs a straight-forward mapping from project components to activities, based on user selected or input attributes and crew names. Refer to Chapter 4 for an overview of the system and to Appendix 2 for a schematic of the system menus.

Project components are sometimes considered to be synonymous with activities in simple schedules. Although this is occasionally a valid assumption, it often leads to a overly simplified view of construction since it precludes the scheduler from representing any of the necessary temporary materials and equipment and restricts further breakdown of the activity into subactivities. Consideration of only the activity *construct footing X* instead of the obvious subactivities indicates the benefit of this richer representation. This mapping of project assemblies, crews, and methods is a complex part of the scheduling process; the treatment here offers some solutions, but further research is required.

6.1. ACTIVITY GENERATION

ACP activities are generated in three steps.

- The user selects Initialize in the Activity Attributes menu and then enters the default attribute values from the next menu. These attributes and their default values are discussed in detail below.

- The user edits the default values to reflect the project status and construction methods. This step allows the user to input knowledge about the project in the form of attributes. Automation of this step offers fertile ground for further research.

- Each attribute list is converted into an activity. This is accomplished by applying a function or functions to the remaining attributes. This function(s) is selected by the system based on the activity type attribute.

An example of the typical representation of an activity is shown in Figure 6.1. The words next to the left margin followed by colons are the activity slot names and the words and numbers which follow them are the slot values.

Name:	Const_COLUMN_A1_1
Components:	COLUMN_A1_1
Initial State:	0
Duration:	6
Supported_By:	FOOTING_A1
Add to Status:	COLUMN_A1_1
Crew:	CREW_COLUMN
Procure:	0
Temporary:	
Cando Test:	

Figure 6.1 Typical Activity Representation

Activity Attributes are used to determine the resulting activities. The following seven Activity Attributes have been implemented:

- Component
- Activity Type
- Crew
- Initial State
- Duration
- Temporary
- Procure

This is not intended to be a comprehensive list. As more diverse projects are analyzed and more concise attribute combinations are discovered, the attribute list should be modified.

As noted above, a default list is generated, primarily by the system, and then the user adds status and construction methods knowledge to the activities by editing the list. Figure 6.2 shows an edited attribute list for a small pier construction project. Each line of this figure is an attribute list.

COMPONENTS	ACTIVITY TYPE	CREW	INIT. STATE	DUR.	PROCURE	TEMPORARY
EARTH	CONSTRUCT	CREW_0	1	0	0	
THRUST_BEAM	EXCAVATE	CREW_O	0	6	0	
FENDER_PILES	CONSTRUCT	CREW_PILES	0	3	1	
CONCRETE_PILES	CONSTRUCT	CREW_PILES	0	14	1	
BACKFILL	CONSTRUCT	CREW_0	0	1	0	
RIPRAP	CONSTRUCT	CREW_0	0	5	1	
PILE_CAPS	MANUFACT	CREW_0	0	4	0	
TEMPORARY_GIRTS	TEMPORARY	CREW_0	0	12	0	CIP_CONCRETE
DECK_PANELS	MANUFACT	CREW_0	0	7	0	
CIP_CONCRETE	CONCRETE	CREW_0	0	8	0	
RUBBER_BUMPERS	CONSTRUCT	CREW_0	0	8	1	
PIER_HARDWARE	CONSTRUCT	CREW_0	0	1	0	
MECH_SERVICE	CONSTRUCT	CREW_0	0	4	0	
EXISTING_PIER	DEMOLISH	CREW_0	0	6	0	

Figure 6.2 Typical Attribute List

6.2. COMPONENT ATTRIBUTE

The component list is taken directly from the project description information. The length of the component list determines the number of attribute lists which are created by the system. It does not, however, determine the number of activities which will be created. There are three other considerations which can influence the number of project activities: a) the activities Mobilization and Demobilization are created for all projects, b) some activity types result in more than one activity for one project component, and c) it is also possible to map more than one project component to one activity, but this has not been implemented in the current ACP system.

6.3. Activity Type Attribute

Six activity type attribute values have been implemented:

- CONSTRUCT
- CONCRETE
- MANUFACTURE
- DEMOLISH
- TEMPORARY
- EXCAVATE

The default value is CONSTRUCT. The name assigned to an activity which is created based on a CONSTRUCT activity type attribute is Const_<component>[1], e.g., Const_COLUMN_A1_1. The component associated with this activity, the initial state of the activity, and the activities duration are obtained directly from the attribute list and are stored within the activity representation. The parents of the <component> in the support relationship (discussed in Section 5.1.) are inserted in the Supported_By slot. The <component> is entered in the Add to Status slot. The Crew and Procure slot values are taken directly from the respective attributes. Neither the Temporary nor the Cando Test slots are used for Const activities. This information is summarized in Figure 6.3.

It is clear how this simple activity is scheduled: the *supported by* rule is fired, it looks in the Supported_By slot and checks to see if the parents of this component (in the supported by relationship) exist in the STATUS, if they do not exist the activity is deleted from the NEWCANDO list, (see Section 5.3.). All other rules will also have an opportunity to constrain and prioritize this activity, but the simplest case is straight forward. Constraints for other activity types are more complicated. In discussion of these other activities, reference will be made to slots constraining the scheduling of an activity; in reality slots do not constrain activities—a rule uses the value of a slot to constrain the activity. This statement is used only when the rule is obvious and not germane to the discussion.

[1]In the remainder of this Chapter character strings within angle brackets refer to attribute values. Also, generic activities will be referred to by the initial portion of their name, eg., Const, Form, etc. See Figure 6.4.

Name:	Const_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	<duration>
Supported_By:	<component> parents in support relationship
Add to Status:	<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	

Figure 6.3 Const Activity Representation

Most of the other activity type choices result in the creation of more than one activity. Figure 6.4 indicates the mapping of the various activity type attribute choices to the activities which they create. In other respects, the remaining activities are very much the same as Const activities. Figures 6.5 through 6.9 summarize the variation of activities which result from these choices. Appendix 4 provides the full representation of these activities. There are a very large number of possible activity configurations based on these six activity types and even a modest number of other activity attributes. I will not attempt to enumerate all of these, but I will highlight each of the activity types in order to indicate the capabilities which are available.

The CONCRETE activity type (shown in Figure 6.5) causes five activities to be created. In the first of these, Form_<component>, the Supported_By slot may constrain the activity from being scheduled. The remaining activities, Rebar_<component> through Strip_<component>, are constrained by the completion of the previous activity. Strip_<component> adds the component's name to the STATUS.

If MANUFACTURE is chosen as the activity type, two activities are created along with the typical Const activity. The first of these activities, Estab_Facil_<component>, provides

field personnel an opportunity to set up a facility for manufacturing the component (eg. precast concrete members) and can be started whenever the project has been mobilized. The second, Manuf_<component>, allows for the actual manufacturing of the component and is constrained from starting by the completion of the first.

The Demo activity is created for existing project components that must be demolished and is allowed to begin whenever the Mobilize activity is complete. The use of TEMPORARY as an activity type is described within the explanation of the TEMPORARY slot, Section 6.7. The EXCAVATE activity type causes an excavation activity to be created and completed prior to the the construction of the component. Note that if there is only one excavation activity, a simpler solution is to use EARTH as the component to be excavated without an associated Const activity.

Although Chapter 4 attests to the full effect of these selections, these effects will be clear only after the other attributes and activity slots are described and are seen in action in Chapter 8.

Activity Type	Activity Name	Figures	
CONSTRUCT	Const_<component>	6.3	A4.1
CONCRETE	Form_<component>	6.5	A4.2
	Rebar_<component>	6.5	A4.3
	Conc_<component>	6.5	A4.4
	Cure_<component>	6.5	A4.5
	Strip_<component>	6.5	A4.6
MANUFACTURE	Estab_Facil_<component>	6.6	A4.7
	Manuf_<component>	6.6	A4.8
	Const_<component>	6.6	A4.1
DEMOLISH	Demo_<component>	6.7	A4.9
TEMPORARY	Const_<component>	6.8	A4.1
	Remove_<component>	6.8	A4.10
EXCAVATE	Excav_for_<component>	6.9	A4.11
	Const_<component>	6.9	A4.1

Figure 6.4 Expansion of Activity Type to Activities

Name	Form_ <component>	Rebar_ <component>	Conc_ <component>	Cure_ <component>	Strip_ <component>
Components	<component>	<component>	<component>	<component>	<component>
Initial State	<initial state>	<initial state>	<initial state>	<initial state>	<initial state>
Duration	3/8 x<duration>	2/8 x<duration>	1/8 x<duration>	1/8 x<duration>	1/8 x<duration>
Supported By	<component> support parents	<component> support parents	<component> support parents	<component> support parents	<component> support parents
Add to Status	FORM_ <component>	REBAR_ <component>	CONC_ <component>	CURE_ <component>	<component> <component>
Crew	<crew>	<crew>	<crew>	<crew>	<crew>
Procure	<procure>	<procure>	<procure>	<procure>	<procure>
Temporary					
Cando Test		DONE contains Form_ <component>	DONE contains Rebar_ <component>	DONE contains Conc_ <component>	DONE contains Cure_ <component>

Figure 6.5 Summary of Concrete Activities

Name	Estab_Facil_ <component>	Manuf_ <component>	Const_ <component>
Components	<component>	<component>	<component>
Initial State	<initial state>	<initial state>	<initial state>
Duration	2	10	<duration>
Supported By			<component> support parents
Add to Status	FACILITY_ ESTABLISHED_ <component>	MANUFAC-TURED_ <component>	<component>
Crew	<crew>	<crew>	<crew>
Procure	<procure>	<procure>	<procure>
Temporary			
Cando Test	DONE contains Mobilize <component>	DONE contains Estab_Facil_ <component>	DONE contains Manuf_ <component>

Figure 6.6 Summary of Manufacture Activities

Name	Demo_ <component>
Components	<component>
Initial State	<initial state>
Duration	<duration>
Supported By	
Add to Status	DEMOLISHED_ <component>
Crew	<crew>
Procure	<procure>
Temporary	
Cando Test	DONE contains Mobilize

Figure 6.7 Summary of Demolition Activities

Name	Const_ <component>	Remove_ <component
Components	<component>	<component>
Initial State	<initial state>	<initial state>
Duration	<duration>	<duration>
Supported By	<component> support parents	
Add to Status	<component>	REMOVED_ <component>
Crew	<crew>	<crew>
Procure	<procure>	<procure>
Temporary		<temporary>
Cando Test		STATUS contains <temporary>

Figure 6.8 Summary of Temporary Activities

Name	Excav_for_ <component>	Const_ <component
Components	<component>	<component>
Initial State	<initial state>	<initial state>
Duration	1	<duration>
Supported By		<component> support parents
Add to Status	EXCAVATED_ FOR_ <component>	<component>
Crew	<crew>	<crew>
Procure	<procure>	<procure>
Temporary		
Cando Test	DONE contains Mobilize	

Figure 6.9 Summary of Excavate Activities

Two additional activities are created for every project—Mobilize and Demobilize (See Figure 6.10 and Figures A4.12 and A4.13).

6.4. CREW ATTRIBUTE

The list of crews which have been defined in the resource module and filed are obtained by choosing List Available Crews from the Activity Attributes menu. The user can select and enter the crews when editing the attribute list, Figure 6.2. If a more flexible method of assigning crews is desirable, the system can be easily modified to prompt the user for the crew name when the activity is being created; this approach was followed when implementing the example in Chapter 8. CREW_0 is a dummy crew and is referred to in this thesis as a generic crew.

Name	Mobilize	Demobilize
Components		
Initial State	0	0
Duration	2	2
Supported By		
Add to Status	MOBILIZED	DEMOBILIZED
Crew	CREW_0	CREW_0
Procure	0	0
Temporary		
Cando Test		

Figure 6.10 Summary of Mobilize and Demobilize Activities

6.5. INITIAL STATE ATTRIBUTE

The initial state attribute is used by the scheduling module to determine the activities which are already done and the activities which remain to be done. This attribute slot allows the user to start at an intermediate point in the schedule; activities with an initial state of 1 (meaning complete) are all immediately added to the list of COMPLETED activities and only the activities with an initial state of 0 (meaning not complete) are added to the TODO list.

6.6. DURATION ATTRIBUTE

Durations are entered directly by the user when entering the activity attributes. Certain activities obtain their duration directly from the duration attribute, e.g., Const and Demo. Other activities factor the duration attribute, e.g., Form, Rebar, Conc. The final option is to prompt the user for an absolute duration entered when the activity is created, e.g., Mobilize and Remove; this option proved very flexible and was used for CONCRETE activity types and Fab activities in Chapter 8. An obvious extension to this research is to provide the system with specialized knowledge and procedures to estimate and propose

more accurate durations for each activity. This can readily be done with rules as demonstrated by Hendrickson [87] and by Zozaya [88].

6.7. TEMPORARY ATTRIBUTE

The temporary attribute is used for activities which create and remove a temporary structure such as scaffolding or bracing. It is implicit that temporary structures are utilized to advance the project between their creation and removal. This makes it intuitive to designate a STATUS item which prompts the removal of temporary structures. For example, if scaffolding is required to build a masonry wall, the user specifies SCAFFOLDING as a project component and specifies MASONRY_WALL as the temporary attribute for SCAFFOLDING. This initiates the creation of activities Const_SCAFFOLDING and Remove_SCAFFOLDING. Const_SCAFFOLDING can be scheduled based on either a rule [Section 5.3.] or a Cando Test. Since MASONRY_WALL is designated as the temporary attribute, whenever MASONRY_WALL exists in the STATUS, the Remove_SCAFFOLDING activity will begin. Assuming there is a Const activity associated with the masonry wall, MASONRY_WALL will automatically be added to STATUS whenever it is completed.

6.8. PROCURE ATTRIBUTE

If the procure attribute within an attribute list is 1, it means that the component associated with this attribute list must be procured. This indicates that a Procure activity should be created. The creation of Procure activities is slightly different than other activities, since it is based on an attribute other than activity type. The result is that a Procure activity can be created along with any other activity type, at the users discretion. Of the activity types implemented, the most likely candidates to have an associated Procure activity are CONSTRUCT and TEMPORARY.

After all activities have been created from the attribute list, they can be individually refined using the Activities Edit menu option. Finally, the activities should be filed to enable access from the Schedule module—the subject of the next chapter.

Name:	Procure_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	15
Supported_By:	
Add to Status:	PROCURED_<component>
Crew:	CREW_0
Procure:	<procure>
Temporary:	
Cando Test:	

Figure 6.11 Procure Activity Representation

7. SCHEDULE GENERATION

As noted in Appendix 3 Figures A3.1 and A3.2, ACP is composed of five modules—Project Description, Resource, Activities, Knowledge, and Scheduling modules. This chapter describes the Scheduling module. Chapters 5 and 6 described the four modules which precede schedule generation. Although Chapters 5 and 6 are essential, this Chapter is the kernel of the system.

7.1. ACTIVITY CATEGORIES

ACP incrementally cycles all activities through six categories. These categories are applicable to all construction projects and provide useful concepts when considering the status of activities. The categories are named and defined as follows:

- TODO — All activities which cannot be started yet.
- NEWCANDO — Prospective CANDO activities.
- CANDO — Activities on which it is possible to work this cycle, based on the STATUS and the CANDO knowledge.
- DOING — Activities which, based on the STATUS and ASSIGNMENT knowledge, have been selected to be worked on this cycle.
- ENDED — Activities that were completed in the previous cycle.
- DONE — Activities that were completed in all previous cycles.

The following sentences elucidate the meaning of the first two categories. Prior to scheduling, all project activities are in the TODO list; after the project has been successfully scheduled, the TODO list is empty. The NEWCANDO list is initialized each cycle as the TODO list; as the CANDO knowledge is applied to the NEWCANDO list each rule has the opportunity to delete activities, thereby leaving only those activities which meet all constraints. (As a simplification, NEWCANDO was omitted from the overview given in Chapter 4 since it is used only to allow ready access both to the activities which were active in the last cycle and to the activities which are being considered for this cycle.) The meaning of the other categories are self-explanatory

7.2. STATUS

Keeping track of the status of the project and using this status to determine subsequent scheduling, is a central notion of this thesis.

The categories of activities discussed in the previous section are special status items. These categories were discussed separately due to their unique role in the model. Other status items include:

- Conditions external to the project
  - the day of the week
  - weather
  - labor conditions
- Conditions within the project
  - existence of project components
  - milestones attained
- Available resource quantities
  - workers
  - equipment

As ACP cycles through the project, the system updates the status by such actions as revising the day-of-the-week, adding completed project components, and adjusting the available level of reusable resources.

7.3. ACP KNOWLEDGE BASES

Sections 4.3. and 5.3. provide examples of knowledge and an explanation of how knowledge is input to the system. The following subsection recapitulates earlier discussion and provides further explanation.

7.3.1. System Knowledge

ACP uses three knowledge bases—STATUS, CANDO, and ASSIGNMENT knowledge. The STATUS knowledge base primarily checks for cumulative effects or milestones such as:

If
the roof is complete
and
the walls are complete
and
the glazing is complete

then
a controlled environment exists.

Or, in executable form.

If
STATUS contains all {of the} ROOF
and
STATUS contains all {of the} WALLS
and
STATUS contains all {of the} GLAZING

then
add to STATUS 'Controlled Environment'

Here the terms ROOF, WALLS, and GLAZING are user-defined variable names as described in Section 5.3.

The CANDO knowledge base includes principles which (a) constrain the order of activities and (b) are rarely, if ever, violated.

If
the members which provide gravity support for member X do not exist in the STATUS

then
member X cannot be started.

The ASSIGNMENT knowledge base contains constraints which typically embody the scheduler's preferences. The following three rules indicate how a number of rules can work together to specify preferences. The first rule reorders the list of CANDO activities

from most important to least important (based on the assumption that it is more efficient to construct footings progressively starting at one corner of a building, rather than choosing footings randomly or alphabetically). The second rule updates the available quantity of resources and the third rule deletes the lower priority activities for which adequate resources are not available.

If

CANDO contains Footings
and
the CANDO Footing list length is greater than 1

then

{the} CANDO Footing list {will be} reordered {based on the} 'adjacent_to Footings contained in STATUS'.

If

activity X consumes Q units of resource R
and
greater than or equal to Q units of resource R are available

then

decrease the available quantity of resource R by Q units.

If

activity X consumes Q units of resource R
and
less than Q units of resource R are available

then

activity X cannot be started.

The last two rules represent a generic resource constraint. Resources are usually considered to be more than preferences, but nonetheless are set by the scheduler for planning purposes and therefore fit within ASSIGNMENT knowledge.

7.3.2. Constraint versus Opportunistic Knowledge

The above CANDO knowledge is configured in the form of constraints. That is, in the general form:

If
condition X is not true

then
do not allow activity Y to be scheduled.

This constraint-based approach starts by assuming that the list of *all* activities should be scheduled and then deletes from this list those activities which *do not* meet the stated conditions. An opportunistic approach can be used to obtain equivalent results; it starts by assuming that *no* activities (an empty list) should be scheduled and then adds to this list those activities which *do* meet the stated conditions. The general form of opportunistic rules is as follows:

If
condition X is true

then
allow activity Y to be scheduled

The constraint-based approach has been chosen for this implementation because I feel that it will result in fewer rules. This has not been tested and may in fact vary across different types of construction or different projects. The opportunistic approach is equally viable within ACP as is seen by its use in the STATUS rule given above.

7.4. SCHEDULING ALGORITHM

Section 4.4. lists six steps which summarize the algorithm used by ACP to schedule activities. Figure 7.1 provides a more detailed perspective of this algorithm.

```
INITIALIZE:
  initialize the durations of activities
  initialize the ENDED activities and TODO activities
  initialize the Gantt chart

BEGIN CYCLE:
  display selected output

  find the effects of the ENDED activities
  add these effects to the STATUS using the STATUS knowledge

  add to DONE the ENDED activities
  delete from CANDO the ENDED activities

  assign to NEWCANDO all activities in TODO
  apply the CANDO knowledge to the NEWCANDO activities
    (i.e., delete from NEWCANDO all activities which do not meet
    the CANDO constraints)
  add the resulting NEWCANDO activities to CANDO
  delete the resulting NEWCANDO activities from TODO

  apply the ASSIGNMENT knowledge to the CANDO activities
    (i.e., delete from CANDO all activities which do not meet the
    ASSIGNMENT constraints)
  add the resulting CANDO activities to DOING

  display selected output
  branch to END if the DOING list is empty

  find the DOING activity with the shortest remaining duration
  assign this duration to TICK
  decrease the remaining durations of all DOING activities by TICK

  add to the Gantt chart the DOING activities and TICK periods
  assign to ENDED all DOING activities which have a remaining
    duration of zero
  reassign to DOING all DOING activities which have a remaining
    duration which is not equal to zero
  branch to BEGIN CYCLE

END:
  display end-of-project output
```

Figure 7.1 ACP's Algorithm—Detailed

A comparison indicates that the algorithm in Figure 7.1 provides far more steps than the one in Section 4.4. and that the cycle is started and ended at different points. Both

algorithms are essentially the same; the above entrance and exit points were chosen to minimize repetition of code.

The algorithm employs some of the concepts used in the parallel approach to resource allocation, as described in Paulson [75 p.42], but significantly extends this approach to include the documentation of status and the utilization of constraint knowledge.

7.5. INTERFACE

Implementation of interface issues were omitted from the previous section since they are not germane to the algorithm. The interface alternatives are discussed in this Section and their place in the implementation is shown in Appendix 3.

7.5.1. Flags

The scheduler can use a variety of flags to modify the interface. These flags can be revised prior to scheduling a project by editing the zeros and ones in the Flag Interface Screen which is reproduced in Figure 7.2. This screen is obtained by choosing *Flags* from the first menu in the Scheduling module. A flag is set to "on" when it has a value of "1."

```
                FLAG INTERFACE SCREEN

Q       <-  0   Query Mode
U       <-  0   User-Directed Mode

K       <-  1   Apply Knowledge

G       <-  0   Gantt Chart display
S       <-  0   STATUS display
T       <-  0   TODO display
N       <-  0   NEWCANDO display
C       <-  1   CANDO display
D       <-  1   DOING display
E       <-  1   ENDED display
DE      <-  0   DONE display
```

Figure 7.2 Flag Interface Screen

In order to select query mode, the user must insert a "1," instead of the "0" shown, as the value to be assigned to Q. (Query mode as well as the other influences of flags are described in the following subsections.) A "1" on the second line of the screen shown in

Figure 7.2 selects user-directed mode; a "0" selects automatic mode. The K flag is used during prototyping to apply or not apply the knowledge bases. This is useful to ensure that when the knowledge bases are not applied, all activities are scheduled in parallel. The remaining flags simply specify the information which will be displayed each cycle during scheduling.

Most of the flags can only be revised via the Flag Interface Screen; this restricts the user's control of the interface to be before or after a project is scheduled. In the case of flags Q, U, and G, it is often convenient to revise the flag during scheduling. This can be done by pressing the letter Q, U, or G and thereby toggling the flag value and dynamically revising the interface at the end of the current cycle.

7.5.2. Query Mode

If the query mode flag is set, the user enters the query facility at the end of each cycle. An alternative method of reaching the query facility is to select Query from the first Scheduling menu. Upon entering the query facility the user is presented with the alternatives shown in Figure 7.3.

The query facility allows the user to ask questions about current and previous cycles and is a significant result of this research. The ability to ask questions about what the system status is, about when the status changed, and about why it changed, is a natural product of most knowledge based systems. The concept of querying a computer is not new, but it has not been applied in this form to construction scheduling systems.

Upon entering the query facility, the user has the six menu options shown in Figure 7.3. The following paragraphs describe each of these options.

The first option is to view the Gantt chart. This Gantt chart documents the schedule as it is developed and provides a record for future use.

The second option is to view the activities. The user selects one of the six activity categories; this results is a list of the activities which are currently in that category. Finally one of these activities can be selected and viewed, although the list of activities are usually of most interest.

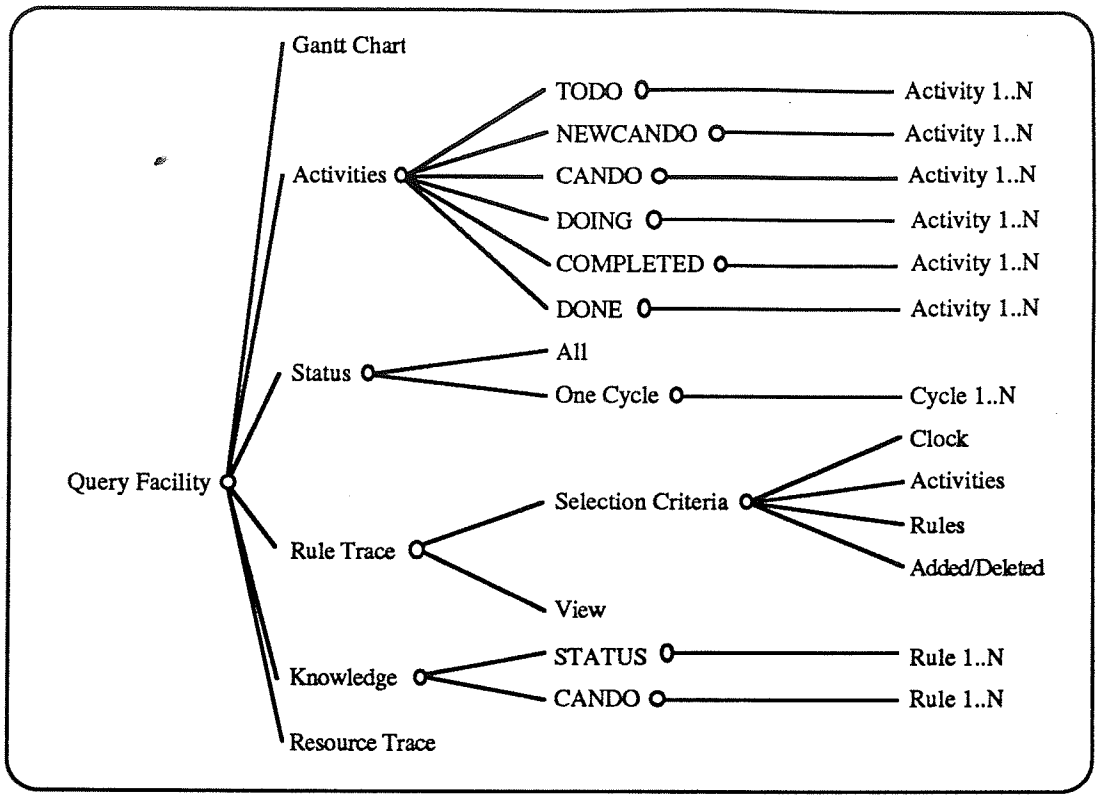

Figure 7.3 Query Menus

Next the STATUS can be chosen. This is where the items described in Section 7.2 are documented. This information can be viewed as a whole or indexed by cycle.

The rule trace is best reduced in size by using one or more of the four selection criteria. These criteria provide ample opportunity to probe the system. All criteria are selected from menus. The Clock criterion allows the user to specify one or more cycles of interest by selecting the clock time at the beginning of the cycle(s). The default is all previous cycles. The Activities and Rules criteria each allow the user to select one or more activities or rules respectively for viewing. Again the default is "all." If Deleted is chosen under the Added/Deleted criterion, then only those activities which were deleted by a rule are displayed. The default for this option is to display both added and deleted activities.

As will be seen from the example below, the user may want to view the rules after doing a rule trace. Easy access to the rules without leaving the query facility is the purpose of the Knowledge option from the main Query menu. The resource trace allows the user to view documentation of resource usage.

The following is an example of a query facility interaction. A user might view the Gantt chart and find that the second-floor slabs were not scheduled until day 30 (cycle 9) of the project, whereas the user expected them to be scheduled on day 24 (cycle 8). (This information could also have been obtained by viewing the DOING list under Activities during cycle 8.) By selecting one or more activities (i.e., the second-floor slab activities) and the Deleted criterion for a rule trace, the user can view all rules which deleted the specified activities on all cycles. Suppose this indicates that the Supported_By rule (see Section 5.3.) and the following Safety_Hazard rule were constraints until cycle 8 and then only the latter rule constrained starting the second-floor slabs until cycle 9. The user can then review the Safety_Hazard rule and presumably will find either from the text of the rule or an associated comment (as indicated below) that this rule was inserted to constrain overhead concrete activities while workers are busy completing structural components below.

```
If
    STATUS does not contain 'FLOOR_1'

then
    delete from NEWCANDO FLOOR_2_SLABS
{the purpose of this rule is to eliminate the safety hazard from
FLOOR_2_SLABS while workers are working below on FLOOR_1}
```

7.5.3. Back-Up Mode

Back-up mode allows the user to return to the previous cycle, i.e., to undo activities which have been scheduled. Just as the user can enter the query or the user-directed modes during scheduling by typing a Q or U, the user can also enter the back-up mode by pressing a B at any time during scheduling.

After backing up one cycle, the user is asked whether or not she would like to back-up again or go forward. By repeatedly choosing back-up again, the user can return all the way to the initial project state.

This mode is most useful in conjunction with the query mode. The result of a query may inspire the user to return to a previous cycle to investigate further, or may inspire the user to override ACP's decision at an earlier cycle by using the user-directed mode which is described in the next subsection.

The function "back-up" simply deletes all system changes which were made during the previous cycle. It uses the variable CLOCK and the Gantt chart to identify the latest changes. This function is shown in Appendix 3 and is clear enough for the reader to understand.

7.5.4. User-Directed versus Automatic Mode

This mode allows the user to override any of ACP's activity scheduling decisions. If this flag is set (has a value of 1), the user is prompted to confirm the activities which the system is about to convert into the DOING list. This confirmation is a two-step process; the user may:

1. Postpone activities which the system suggests should be scheduled, or

2. Select additional activities to be scheduled which the system has not suggested.

7.6. SUMMARY

This chapter has explained how ACP generates a schedule. The algorithm, its characteristic use of status information, and the interface have been described in general terms. The use of status as an integral part of determining what activities should be scheduled next is a unique capability of ACP. The opportunity to query a construction scheduling system about its intermediate decisions is also unique. The following chapter exercises ACP on a typical construction project.

8. ACP APPLICATION

Chapters 5, 6, and 7 described the operation of ACP; this Chapter applies ACP to a construction project. The first Section of this Chapter describes the project and the second Section applies the typical network diagraming techniques (NDTs). Sections 8.3. and 8.4. apply ACP to the project.

8.1. WAREHOUSE EXAMPLE

The project chosen is one which has been used by J.W. Fondahl in a course (CE 241) he has taught for many years at Stanford's Construction Engineering and Management Program; the project was originally drawn from [Seabee circa 1960]. This project was chosen because it is sufficiently complex and student solutions over many years provide some confidence in the range of "good" alternative solutions to this problem which cannot be solved for optimality. Figure 8.1 lists the activities, their durations, their predecessors, and their required resources. (Note that activities 10 and 27 are subcontracted and therefore do not require any of the specified resources.) Figure 8.2 shows a network diagram for the project. In the ACP treatment of this project one additional constraint is added between activities 22 and 29; this modification is described in Section 8.3.

8.2. NETWORK DIAGRAMING TECHNIQUES

Figure 8.3 summarizes the results of forward and backward pass calculations and float calculations; these calculations and references for them are discussed in Section 2.2.

Following the steps given in Section 2.2.2., resource allocation techniques were applied resulting in the resource allocation chart shown in Figure 8.4. A parallel resource allocation algorithm was used with the following priorities for scheduling activities constrained by resources:

1. Largest Late Start
2. Smallest Total Float (in the case of a tie)
3. Input Order (in the case of a remaining tie)

Act. No.	Description	Dur-ation	Prede-cessor(s)	Resources
1	Site Survey and Material List	1		0
2	Materials and Equipment to Site	3	1	0
3	Fasten Wood Strips to Purlins	2	2	2C
4	Fabricate Interior Wall Partitions	2	2	4C
5	Fabricate Timber Trusses	4	2	4C
6	Excavate	1	2	4L
7	Construct Plywood Exterior Wall Panels	3	2	4C
8	Fabricate Exterior Doors	2	2	2C
9	Forms, Rebar, Bolts for Column Ftgs	1	6	4C,2I,2L
10	Place Drain Pipes	1	6	0
11	Pour Footings	1	9	1C,2L
12	Strip and Cure Footings	4	11	1C,1L
13	Erect Steel Columns and Vertical Brace	1	12	4I
14	Backfill, Compact, and Grade Slab	2	10, 12	2L
15	Erect Trusses	1	5, 13	4I
16	Forms, Rebar, and Drains Floor Slab	1	14	4C,2I
17	Fabricate and Erect Frames at Truss Pans	1	15	2C,2I
18	Pour Slab	1	13, 16	1C,2L
19	Erect Steel Purlins	1	3, 17	4I
20	Cure Slab	6	18	0
21	Place Rod Bridging and Weld	1	19	4I
22	Erect Exterior Wall Panels	2	7, 20	4C
23	Nail Planks to Purlins	2	21	2C
24	Waterproof and Insulate Joints	1	22	2L
25	Erect Doors	1	8, 22	2C
26	Lay Tar Paper on Planks	1	23	2L
27	Paint Exterior Walls	2	24, 25	0
28	Erect Corrugated Roofing	2	26	2C
29	Erect Interior Wall Partitions	1	4, 20, 26	4C
30	Cleanup	2	27, 28, 29	4L

Figure 8.1 Warehouse Activities and Precedence

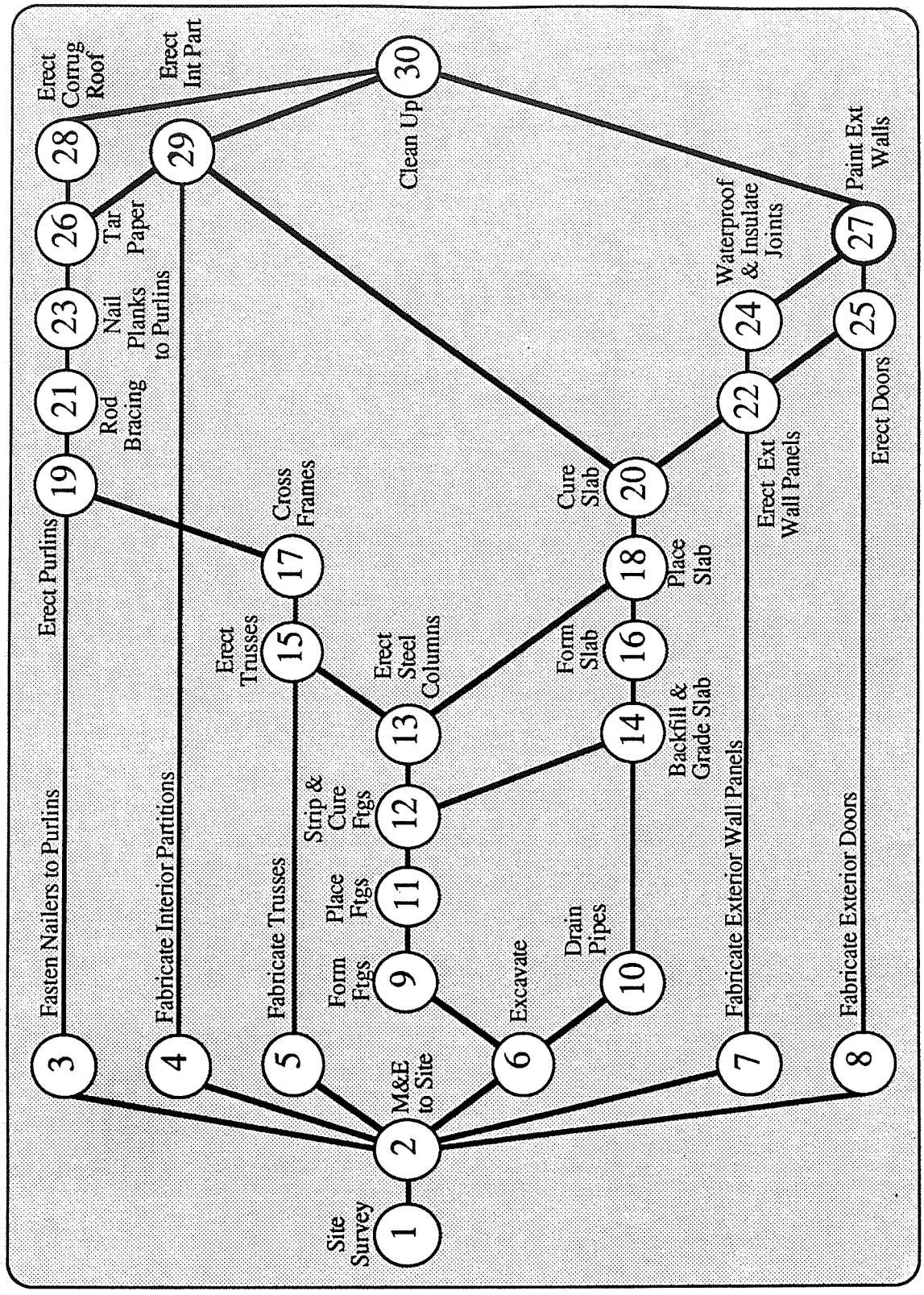

Figure 8.2 Warehouse Network Diagram

Act No	Description	Crit	Dur	ES	LS	EF	LF	TF	FF
1	Site Survey and Material List	1	1	1	1	2	2	0	0
2	Materials and Equipment to Site	1	3	2	2	5	5	0	0
3	Fasten Wood Strips to Purlins	0	2	5	18	7	20	13	8
4	Fabricate Interior Wall Partitions	0	2	5	24	7	26	19	15
5	Fabricate Timber Trusses	0	4	5	14	9	18	9	4
6	Excavate	1	1	5	5	6	6	0	0
7	Construct Plywood Exterior Wall Panels	0	3	5	19	8	22	14	14
8	Fabricate Exterior Doors	0	2	5	22	7	24	17	17
9	Forms, Rebar, Bolts for Column Ftgs	1	1	6	6	7	7	0	0
10	Place Drain Pipes	0	1	6	11	7	12	5	5
11	Pour Footings	1	1	7	7	8	8	0	0
12	Strip and Cure Footings	1	4	8	8	12	12	0	0
13	Erect Steel Columns and Vertical Brace	0	1	12	14	13	15	2	0
14	Backfill, Compact, and Grade Slab	1	2	12	12	14	14	0	0
15	Erect Trusses	0	1	13	18	14	19	5	0
16	Forms, Rebar, and Drains Floor Slab	1	1	14	14	15	15	0	0
17	Fabricate and Erect Frames at Truss Pans	0	1	14	19	15	20	5	0
18	Pour Slab	1	1	15	15	16	16	0	0
19	Erect Steel Purlins	0	1	15	20	16	21	5	0
20	Cure Slab	1	6	16	16	22	22	0	0
21	Place Rod Bridging and Weld	0	1	16	21	17	22	5	0
22	Erect Exterior Wall Panels	1	2	22	22	24	24	0	0
23	Nail Planks to Purlins	0	2	17	22	19	24	5	0
24	Waterproof and Insulate Joints	1	1	24	24	25	25	0	0
25	Erect Doors	1	1	24	24	25	25	0	0
26	Lay Tar Paper on Planks	0	1	19	24	20	25	5	0
27	Paint Exterior Walls	1	2	25	25	27	27	0	0
28	Erect Corrugated Roofing	0	2	20	25	22	27	5	5
29	Erect Interior Wall Partitions	0	1	22	26	23	27	4	4
30	Cleanup	1	2	27	27	29	29	0	0

Figure 8.3 Forward Pass, Backward Pass, and Float Calculations

Act	Res	Dur	LS	TF	0 10 20 30
1	0	1	1	0	X... ....
2	0	3	2	0	.XXX ....
3	2C	2	18	13	.... .XX. ....
4	4C	2	24	19	....XX... ....
5	4C	4	14	9	....X---XXX... ....
6	4L	1	5	0	....X.... ....
7	4C	3	19	14	.... .XX--X... ....
8	2C	2	22	17	.... .XX. ....
9	4C,2I,2L	1	6	0	.... X... ....
10	0	1	11	5	.... X... ....
11	1C,2L	1	7	0	.... .X.. ....
12	1C,1L	1+3	8	0	.... ..X---... ....
13	4I	1	14	2	.... .X.. ....
14	2L	2	12	0	.... .XX. ....
15	4I	1	18	5	....X. ....
16	4C,2I	1	14	0	....X ....
17	2C,2I	1	19	5	....X.... ....
18	1C,2L	1	15	0	....X.... ....
19	4I	1	20	5	.... X... ....
20	0	6	16	0	.... XXXXXX... ....
21	4I	1	21	5	.... .X.. ....
22	4C	2	22	0	.... .XX. ....
23	2C	2	22	5	....XX ....
24	2L	1	24	0	....X ....
25	2C	1	24	0	....X ....
26	2L	1	24	5	....X.... ....
27	0	2	25	0	....XX...
28	2C	2	25	5	....XX....
29	4C	1	26	4	.... X...
30	4L	2	27	0	.... .XX.
Carpenters	C	Max=4			00004433444444342424444424 00
Ironworkers	I	Max=4			00000200000442244000000000
Laborers	L	Max=4			00004221000220200002000200 44

Figure 8.4 Warehouse Resource Allocation Chart (NDT)

8.3. ACP PRECEDENCE CONSTRAINTS

The following tasks are necessary in order to prepare ACP for scheduling:

- specify the resources which are available
- enter the project components and their relationships
- create the project activities, and
- enter the sequencing constraints.

The following ten crews are named to represent each unique combination of resources for the Warehouse project:

CREW_2C	CREW_4I
CREW_2L	CREW_4L
CREW_1C1L	CREW_2C2I
CREW_1C2L	CREW_4C2I
CREW_4C	CREW_4C2I2L

The crew representation is the same as is described in Section 5.2.

The project contains the eighteen components noted below:

EARTH	FOOTINGS
STEEL_COLUMNS	DRAIN_PIPES
BACKFILL	SLAB
TRUSSES	CROSS_FRAMES
PURLINS	ROD_BRACING
PLANKS	TAR_PAPER
CORRUG_ROOF	INTERIOR_PART
EX_WALL_PANELS	EXTERIOR_DOORS
PANEL_JOINTS	PAINT_WALL_PAN

Fourteen of these components belong in the *supported_by* relationship shown in Figures 8.5 and 8.6. The internal representation is a Boolean matrix similar to the one shown in Figure 5.1.

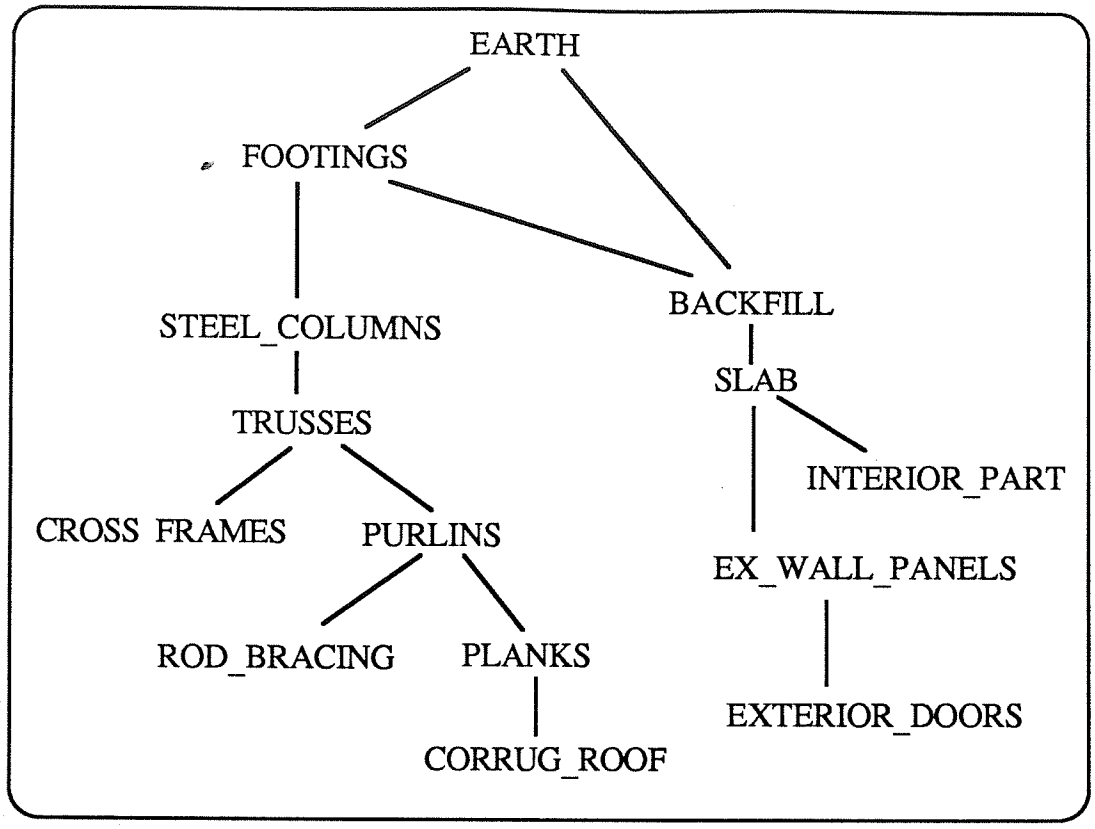

Figure 8.5 Graph Representation of the *supported_by* Relationship

```
EARTH
    BACKFILL
        SLAB
            EX_WALL_PANELS
                EXTERIOR_DOORS
            INTERIOR_PART
    FOOTINGS
        BACKFILL
            SLAB
                EX_WALL_PANELS
                    EXTERIOR_DOORS
                INTERIOR PART
        STEEL_COLUMNS
            TRUSSES
                PURLINS
                    PLANKS
                        CORRUG_ROOF
                    ROD_BRACING
                CROSS_FRAMES
```

Figure 8.6 Indented Representation of the *supported-by* Relationship

The Warehouse Attribute List, shown in Figure 8.7, is generated based on these components. During the generation of this list the user enters the durations and selects the other information associated with these components from menus. From this list ACP then creates the activities. In cases where activity elaboration occurs, such as with CONCRETE and Fabrication activities, the user must again enter durations and select crews. This results in the activities listed in Figure 8.8.

COMPONENTS	ACTIVITY_ TYPE	CREW	INITIAL_ STATE	DUR	FAB
EARTH	EXCAVATE	CREW_4L	0	1	0
FOOTINGS	CONCRETE	CREW_0	0	0	0
STEEL_COLUMNS	CONSTRUCT	CREW_4I	0	1	0
DRAIN_PIPES	CONSTRUCT	CREW_0	0	1	0
BACKFILL	CONSTRUCT	CREW_2L	0	2	0
SLAB	CONCRETE	CREW_0	0	0	0
TRUSSES	CONSTRUCT	CREW_4I	0	1	1
CROSS_FRAMES	CONSTRUCT	CREW_2C2I	0	1	0
PURLINS	CONSTRUCT	CREW_4I	0	1	1
ROD_BRACING	CONSTRUCT	CREW_4I	0	1	0
PLANKS	CONSTRUCT	CREW_2C	0	2	0
TAR_PAPER	CONSTRUCT	CREW_2L	0	1	0
CORRUG_ROOF	CONSTRUCT	CREW_2C	0	2	0
INTERIOR_PART	CONSTRUCT	CREW_4C	0	1	1
EX_WALL_PANELS	CONSTRUCT	CREW_4C	0	2	1
EXTERIOR_DOORS	CONSTRUCT	CREW_2C	0	1	1
PANEL_JOINTS	CONSTRUCT	CREW_2L	0	1	0
PAINT_WALL_PAN	CONSTRUCT	CREW_0	0	2	0

Figure 8.7 Warehouse Attribute List

As described in Chapter 7, sequence constraints are stored as STATUS knowledge and CANDO knowledge. The STATUS knowledge for this project is shown in Figure 8.9 and the CANDO knowledge is shown in Figure 8.10.

The first two rules in the STATUS knowledge are used to signify the completion of the FOOTINGS and SLAB respectively. Figure 6.5 provides a more standardized approach to concrete activities, but does not match the network diagramming approach used for the Warehouse project.

In Figure 8.4, different resources are specified for stripping and curing footings; this requires that two separate activities be created. The slab, on the other hand, only uses one crew for curing—presumably including the minor task of stripping or leaving the

formwork in place. This is an atypical situation, but is easily handled by the first two STATUS rules. In addition, the first rule also displays the ability to represent the parallel nature of the constraint between curing and stripping, within a single rule rather than as four separate precedence constraints.

NDT Act. No.	ACP Activity Name	CREWS	DUR
1,2	Mobilize	CREW_0	4
3	Fab_PURLINS	CREW_2C	2
4	Fab_INTERIOR_PART	CREW_4C	2
5	Fab_TRUSSES	CREW_4C	4
6	Excav_EARTH	CREW_4L	1
7	Fab_EX_WALL_PANELS	CREW_4C	3
8	Fab_EXTERIOR_DOORS	CREW_2C	2
9	Form_FOOTINGS	CREW_4C2I2L	1
10	Const_DRAIN_PIPES	CREW_0	1
11	Place_FOOTINGS	CREW_1C2L	1
12	Strip_FOOTINGS	CREW_1C1L	1
12	Cure_FOOTINGS	CREW_0	4
13	Const_STEEL_COLUMNS	CREW_4I	1
14	Const_BACKFILL	CREW_2L	2
15	Const_TRUSSES	CREW_4I	1
16	Form_SLAB	CREW_4C2I	1
17	Const_CROSS_FRAMES	CREW_2C2I	1
18	Place_SLAB	CREW_1C2L	1
19	Const_PURLINS	CREW_4I	1
20	Cure_SLAB	CREW_0	6
21	Const_ROD_BRACING	CREW_4I	1
22	Const_EX_WALL_PANELS	CREW_4C	2
23	Const_PLANKS	CREW_2C	2
24	Const_PANEL_JOINTS	CREW_2L	1
25	Const_EXTERIOR_DOORS	CREW_2C	1
26	Const_TAR_PAPER	CREW_2L	1
27	Const_PAINT_WALL_PAN	CREW_0	2
28	Const_CORRUG_ROOF	CREW_2C	2
29	Const_INTERIOR_PART	CREW_4C	1
30	Demobilize	CREW_4L	2

Figure 8.8 ACP Warehouse Activities

```
FOOTINGS_COMPLETE:
   If
         (STATUS contains 'Stripped_FOOTINGS')
         and
         (STATUS contains 'Cured_FOOTINGS')
   then
         add to MILESTONES 'FOOTINGS'


SLAB_COMPLETE:
   If
         STATUS contains 'Cured_SLAB'
   then
         add to MILESTONES 'SLAB'


TRUSS_ASSEMBLY_STAT:
   If
         (STATUS contains 'TRUSSES')
         and
         (STATUS contains 'CROSS_FRAMES')
   then
         add to MILESTONES 'TRUSS_ASSEMBLY')


PURLIN_ASSEMBLY_STAT:
   If
         (STATUS contains 'PURLINS')
         and
         (STATUS contains 'ROD_BRACING')
   then
         add to MILESTONES 'PURLIN_ASSEMBLY'


CONTROLLED_ENV_STAT:
   If
         STATUS contains all ENVELOPE
   then
         add to MILESTONES 'CONTROLLED_ENVIRONMENT'
```

Figure 8.9 Warehouse STATUS Knowledge

The purpose of the third and fourth STATUS rules is clear when considered in conjunction with their counterparts in the CANDO knowledge. Finally the last rule draws on a user defined variable called ENVELOPE which includes the following project components:

SLAB
EX_WALL_PANELS
PLANKS
TAR_PAPER
EXTERIOR_DOORS

The completion of these components signifies that the environment within the Warehouse is protected from the weather by stating that when the STATUS contains all of the ENVELOPE components, a MILESTONE called CONTROLLED_ENVIRONMENT has been reached. As will be seen below, when this milestone is reached it allows INTERIOR_PARTITIONS to begin.

The first two rules (SUPPORT and CANDO_TEST) in the CANDO knowledge shown in Figure 8.10 are described in Section 7.3. The third rule, COLUMNS_EMBEDDED, is an example of a precedence constraint typical of NDTs. As noted above, the fourth rule, TRUSS_ASSEMBLY_CNDO, constrains starting the purlins until after the milestone TRUSS_ASSEMBLY is reached; reasons for this and the fifth rule lie in need for lateral support and access. The *enclosed_by* rules constrain the left argument from starting until one of the objects in the right argument is complete, then, when the left argument is complete, the second of the two right arguments can commence. The *applied_to* rules constrain the left argument object from beginning until all of the right argument objects are complete. Finally as noted above, the last rule constrains work on the INTERIOR_PART until the CONTROLLED_ENVIRONMENT milestone has been reached.

Figure 8.11 shows the Gantt chart which results from the constraints noted above. The schedule indicates that the project will be complete at the end of the 28th day; this corresponds to the information given in Figure 8.3 (here completion is at the beginning of the 29th day). The results are essentially the same except for INTERIOR_PART which ACP schedules later due to the controlled environment constraint and the need for the completion of the EXTERIOR_DOORS to reach this constraint.

```
SUPPORT:
   delete from NEWCANDO
         (NEWCANDO which do not meet 'STATUS contains all support for X')


CANDO_TEST:
   delete from NEWCANDO
         (NEWCANDO which do not meet 'cando test X')


COLUMNS_EMBEDDED:
   'Const_STEEL_COLUMNS' precedes 'Place_SLAB'


TRUSS_ASSEMBLY_CNDO:
   If
         STATUS does not contain 'TRUSS_ASSEMBLY'
   then
         delete from NEWCANDO 'Const_PURLINS'


PURLIN_ASSEMBLY_CNDO:
   If
         STATUS does not contain 'PURLIN_ASSEMBLY'
   then
         delete from NEWCANDO 'Const_PLANKS'


ENCLOSED_DRAIN_PIPE:
   'DRAIN_PIPE' enclosed_by 'EARTH BACKFILL'


ENCLOSED_TAR_PAPER:
   'TAR_PAPER' enclosed_by 'CORRUG_ROOF PLANKS'


APPLIED_JOINTS:
   'PANEL_JOINTS' applied_to 'EX_WALL_PANELS'


APPLIED_PAINT:
   'PAINT_WALL_PAN' applied_to
   'EX_WALL_PANELS  PANEL_JOINTS  EXTERIOR_DOORS'


CONTROLLED_ENV_CNDO:
   If
         STATUS does not contain 'CONTROLLED_ENVIRONMENT"
   then
         delete from NEWCANDO 'Const_INTERIOR_PART'
```

Figure 8.10 Warehouse Cando Knowledge

Activity	Res	Dur	0 10 20 30
Mobilize	0	4	OOOO
Fab_PURLINS	2C	2	....OO...
Fab_INT_PART	4C	2	....OO...
Fab_TRUSSES	4C	4	....OOOO.
Excav_EARTH	4L	1	....O....
Fab_EX_WALL_PAN	4C	3	....OOO..
Fab_EXT_DOORS	2C	2	....OO...
Form_FOOTINGS	4C2I2L	1	.... O...
Const_DRN_PIPES	0	1	.... O...
Place_FOOTINGS	1C2L	1	.... .O..
Strip_FOOTINGS	1C1L	1	.... ..O.
Cure_FOOTINGS	0	4	.... ..OOOO...
Const_STEEL_COL	4I	1	.... .O..
Const_BACKFILL	2L	2	.... .OO.
Const_TRUSSES	4I	1	....O.
Form_SLAB	4C2I	1	....O
Const_CROSS_FR	2C2I	1	....O
Place_SLAB	1C2L	1	....O....
Const_PURLINS	4I	1	....O....
Cure_SLAB	0	6	.... OOOOOO...
Const_ROD_BRACING	4I	1	.... O..
Const_EX_WAL_PAN	4C	2	.... .OO.
Const_PLANKS	2C	2	.... .OO.
Const_PAN_JOINTS	2L	1	....O
Const_EXT_DOORS	2C	1	....O
Const_TAR_PAPER	2L	1	....O
Const_PAINT_PANEL	0	2	....OO...
Const_CORRUG_ROOF	2C	2	....OO...
Const_INT_PART	4C	1	....O....
Demobilize	4L	2	.... .OO.
Carpenters	Max=4		0000[AB]95000006102202244240000
Iron Workers	Max=4		00000200000444440000000000000
Laborers	Max=4		00004221000220200020000200044

Figure 8.11 Warehouse Gantt Chart without Resource Constraints

[A] Twenty carpenters are required on day 5.
[B] Sixteen carpenters are required on day 6.

8.4. ACP Resource Constraints

ACP applies resource constraints within each scheduling cycle rather than applying them after a network has been developed. This is done at the ASSIGNMENT stage of Figure 4.1. Assignment constraints for the Warehouse project simply consist of a maximum resource usage of :

4 Carpenters
4 Ironworkers
4 Laborers

Figure 8.12 shows the result of applying these resource constraints with all priority weights equal. The result is a 39 day project duration. As can be seen, inefficient use of resources has delayed the project significantly. NDTs address this problem of inefficient use of resources by using heuristics which draw on the previous network calculation. Since one of the goals of this research is to avoid the two-stage treatment of non-resource and resource constraints, this approach is not desirable. The approach taken by ACP is to use knowledge about the project and other similar past projects to set priorities. All activities are assigned a default priority chosen by the user. The user then has four methods of refining the default priorities to improve resource peaks and valleys:

- multiply priorities by a factor
- divide priorities by a factor
- add a constant to priorities
- individually edit the priorities.

The first three methods rely on specifying a group of activities or project components (which are associated with activities) and applying the factor or constant to the group. Using these priorities, ACP reorders the CANDO activities in descending order of their priorities and selects the DOING activities from this list within the resource constraints.

From the non-resource-constrained schedule of Figure 8.11, it can be seen that day 5 requires the peak resource demand of 20 carpenters; it is also evident that the Fabrication activities all begin on day 5. This, and the assumption that Fabrication activities will have float, make these activities prime candidates to have a reduced priority. The ENVELOPE and the STRUCTURE are candidates to have increased priorities since they are crucial to

Activity	Res	Dur	0 10 20 30 40
Mobilize	0	4	OOOO ....
Fab_PURLINS	2C	2	....OO... ....
Fab_INT_PART	4C	2	....OO... ....
Fab_TRUSSES	4C	4	.... .OOOO.... ....
Excav_EARTH	4L	1	....O.... ....
Fab_EX_WALL_PAN	4C	3	.... .OOO ....
Fab_EXT_DOORS	2C	2	....OO... ....
Form_FOOTINGS	4C2I2L	1	.... O... ....
Const_DRN_PIPES	0	1	.... O... ....
Place_FOOTINGS	1C2L	1	.... .O.. ....
Strip_FOOTINGS	1C1L	1	....O. ....
Cure_FOOTINGS	0	4	....OOOO... ....
Const_STEEL_COL	4I	1	.... .O.. ....
Const_BACKFILL	2L	2	.... .OO. ....
Const_TRUSSES	4I	1	....O. ....
Form_SLAB	4C2I	1	....O.... ....
Const_CROSS_FR	2C2I	1	....O ....
Place_SLAB	1C2L	1	.... O... ....
Const_PURLINS	4I	1	.... O... ....
Cure_SLAB	0	6	.... .OOOOOO.. ....
Const_ROD_BRACING	4I	1	.... .O.. ....
Const_EX_WAL_PAN	4C	2	....OO ....
Const_PLANKS	2C	2	....OO ....
Const_PAN_JOINTS	2L	1	....O....
Const_EXT_DOORS	2C	1	....O....
Const_TAR_PAPER	2L	1	....O.... ....
Const_PAINT_PANEL	0	2	.... OO..
Const_CORRUG_ROOF	2C	2	.... OO.. ....
Const_INT_PART	4C	1	.... O...
Demobilize	4L	2	....OO
Carpenters	Max=4		00004444444444441100000241022022442400O
Iron Workers	Max=4		00000000000000020000044224400000000000
Laborers	Max=4		00004000000000221000220020002000020044

Figure 8.12 Warehouse Gantt Chart with Resources

the completion of the project. The ENVELOPE components are listed in Section 8.3. and the STRUCTURE was defined as including the following components:

EARTH	TRUSSES
FOOTINGS	PURLINS
STEEL_COLUMNS	ROD_BRACING
CROSS_FRAMES	

This provided an improvement to 31 days. Further trial-and-error testing resulted in the selection of ROOF_STRUCTURE priority being increased instead of the whole STRUCTURE. ROOF_STRUCTURE only includes TRUSSES, PURLINS, and ROD_BRACING.

The final priority choices were as follows based on a default priority of 500:

Add -300 to Fabrication activities.

Add +100 to ENVELOPE components.

Add +200 to ROOF_STRUCTURE components.

Figure 8.13 is the resulting schedule after the above modifications. This is the minimum project time as indicated by the non-resource-constrained schedule shown in Figure 8.11. (It is noteworthy that empirical evidence indicated that it is impossible to obtain a 28-day schedule for this project without interrupting activities and that neither Primavera nor MacProject currently have this capability.)

Fondahl has also tested a variation of the Warehouse project which entails the activity duration extensions shown in Figure 8.14. This is a particularly interesting extension since all activity duration changes are within that activity's float, but in order to maintain resource constraints the total project duration is significantly extended. From the perspective of applying ACP, the interest is solely a matter of how this time extension can be minimized.

If the extended project is run with all activities having the same priority the result is a 61 day schedule. Figure 8.15—which has a 50 day duration—is produced by using the final priorities noted above. This duration is the minimum which has been obtained by use of fixed resource leveling heuristics.

Activity	Res	Dur	0 10 20 30
Mobilize	0	4	OOOO
Fab_PURLINS	2C	2	....O.O..
Fab_INT_PART	4C	2	....O....O....
Fab_TRUSSES	4C	4	...OOOO..
Excav_EARTH	4L	1	....O....
Fab_EX_WALL_PAN	4C	3	..O. OO..
Fab_EXT_DOORS	2C	2	....O..O.
Form_FOOTINGS	4C2I2L	1	O...
Const_DRN_PIPES	0	1	O...
Place_FOOTINGS	1C2L	1	.O..
Strip_FOOTINGS	1C1L	1	..O.
Cure_FOOTINGS	0	4	..OOOO...
Const_STEEL_COL	4I	1	.O..
Const_BACKFILL	2L	2	.OO.
Const_TRUSSES	4I	1	..O.
Form_SLAB	4C2I	1	...O
Const_CROSS_FR	2C2I	1	....O....
Place_SLAB	1C2L	1	....O....
Const_PURLINS	4I	1	O...
Cure_SLAB	0	6	OOOOOO...
Const_ROD_BRACING	4I	1	.O..
Const_EX_WAL_PAN	4C	2	.OO.
Const_PLANKS	2C	2	..OO
Const_PAN_JOINTS	2L	1	...O
Const_EXT_DOORS	2C	1	...O
Const_TAR_PAPER	2L	1	....O....
Const_PAINT_PANEL	0	2	....OO...
Const_CORRUG_ROOF	2C	2	....O...O
Const_INT_PART	4C	1	.....O...
Demobilize	4L	2	.OO.
Carpenters	Max=4		00004433444444344224244444400
Iron Workers	Max=4		00000200000442244000000000000
Laborers	Max=4		00004221000220200020002004400

Figure 8.13 Warehouse Gantt Chart with Resources and Priorities

Activity	Original Duration	Extended Duration
Fab_INT_PART	2	8
Fab_EX_WALL_PANELS	3	12
Fab_EXTERIOR_DOORS	2	7
Const_INTERIOR_PART	1	4

Figure 8.14 Warehouse Project Extension

Activity	Res	Dur	0 10 20 30 40 50
Mobilize	0	4	OOOO ....
Fab_PURLINS	2C	2	....O.O.. ....
Fab_INT_PART	4C	8	.... .OOOOOOOO ....
Fab_TRUSSES	4C	4	....OOOO.. ....
Excav_EARTH	4L	1	....O.... ....
Fab_EX_WALL_PAN	4C	12	....O..OOOOOOOOOO. ....
Fab_EXT_DOORS	2C	7	....O..O. ..O..OOOO ....
Form_FOOTINGS	4C2I2L	1	.... O... ....
Const_DRN_PIPES	0	1	.... O... ....
Place_FOOTINGS	1C2L	1	.... .O.. ....
Strip_FOOTINGS	1C1L	1	.... ..O. ....
Cure_FOOTINGS	0	4	.... ..OOOO... ....
Const_STEEL_COL	4I	1	.... .O.. ....
Const_BACKFILL	2L	2	.... .OO. ....
Const_TRUSSES	4I	1	....O. ....
Form_SLAB	4C2I	1	....O ....
Const_CROSS_FR	2C2I	1	....O.... ....
Place_SLAB	1C2L	1	....O.... ....
Const_PURLINS	4I	1	.... O... ....
Cure_SLAB	0	6	.... OOOOOO... ....
Const_ROD_BRACING	4I	1	.... .O. ....
Const_EX_WAL_PAN	4C	2	....OO.... ....
Const_PLANKS	2C	2	....OO ....
Const_PAN_JOINTS	2L	1	.... O... ....
Const_EXT_DOORS	2C	1	.... O... ....
Const_TAR_PAPER	2L	1	....O.... ....
Const_PAINT_PANEL	0	2	.... .OO. ....
Const_CORRUG_ROOF	2C	2	.... OO.. ....
Const_INT_PART	4C	4	....OOOO.
Demobilize	4L	2	....OO
Carpenters	Max=4		00004433444424322444224444444444442444444444444400
Iron Workers	Max=4		00004221000220200002000000000000002000000000000044
Laborers	Max=4		00000200000442244000000000000000000000000000000000

Figure 8.15 Extended Gantt Chart with Priority Revisions

If the priority for Fab_EXTERIOR_DOORS is individually reduced from 300 to 200, then a 48 day duration can be obtained and is shown in Figure 8.16. The rationale for this revision is to save this two carpenter activity, which does not need to be completed until near the end of the project, to fill in some of the subsequent hollows in the carpenter resource histogram.

By observing the carpenter resource usage in Figure 8.16 it can be seen that days 33 and 34 are a problem. Activities Fab_EXTERIOR_DOORS and Const_EXTERIOR-_DOORS are each responsible for one day of the two carpenter hollow in the histogram. Since these activities are sequential, two other carpenter activities must be found to put in parallel with them. This can be done by pushing forward the Const_PLANKS and the Const_CORRUG_ROOF activities. A further complication is that these latter two activities are separated by Const_TAR_PAPER; this is solved by allowing the Fab_INTERIOR-_PART activity to run in parallel with Const_TAR_PAPER. This is all accomplished by the following additional modifications:

- decreasing the priority of Const_PLANKS from 600 to 200
- postponing the Place_SLAB activity from execution on day 15 (thereby allowing the Fab_EXTERIOR_DOOR to run in parallel with Const_CROSS-_FRAMES on day 15 and with Place_SLAB on day 16)
- postponing Fab_EXTERIOR_DOORS on day 32 (thereby allowing Fab-_INTERIOR_PART to run in parallel with Const_TAR_PAPER).

Postponing activities in done through the user-directed mode described in Section 7.5.4. In this case, other activities, which would normally have been delayed since they have lower priorities than the postponed activities, are automatically selected during the ASSIGNMENT stage and pulled backward to fill the gap left by the postponed activities. The result is Figure 8.17. As can be seen from a close look at Figure 8.17, this is the minimum time which can be achieved for the project.

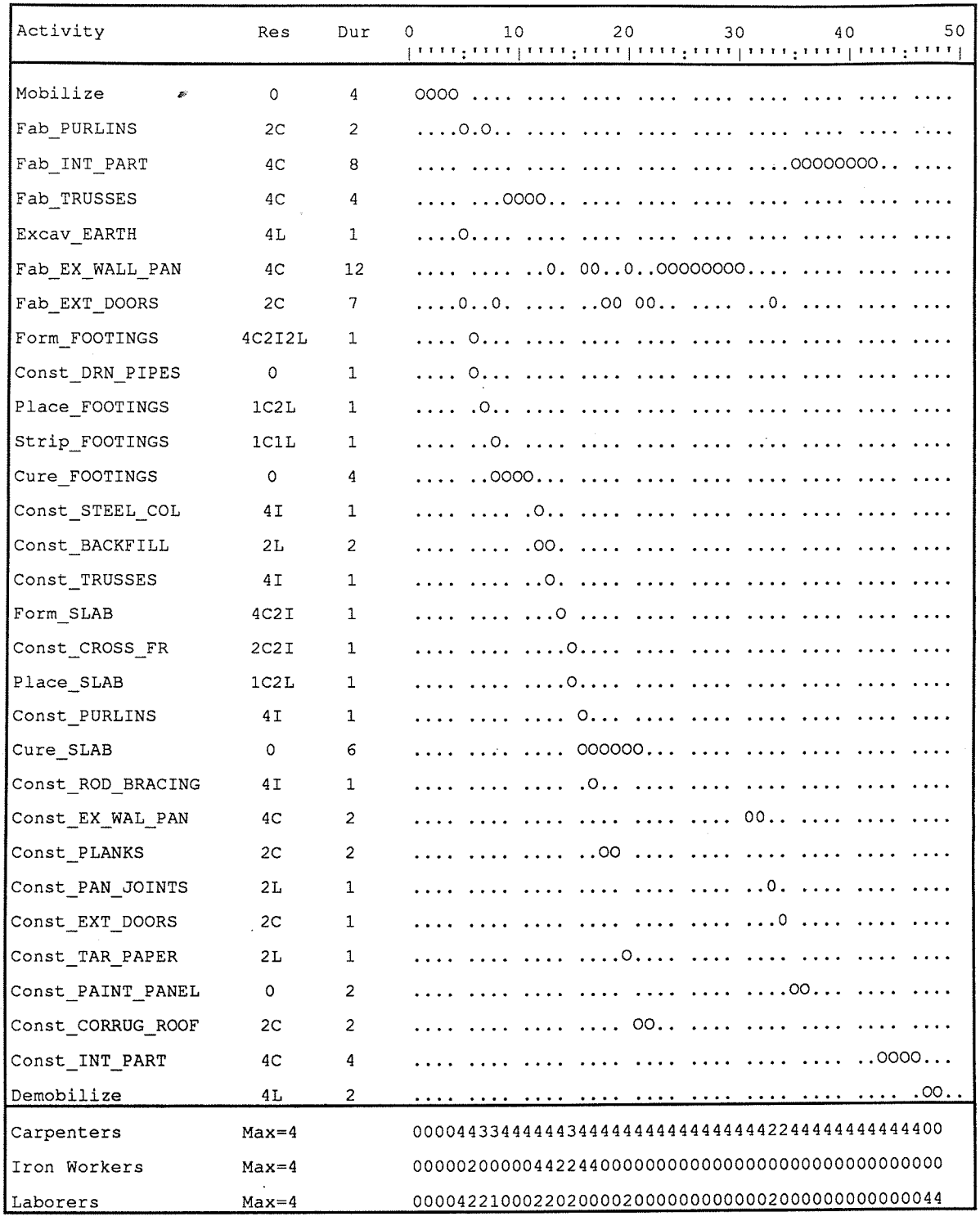

Figure 8.16 Extended Gantt Chart with Individual Priority Revisions

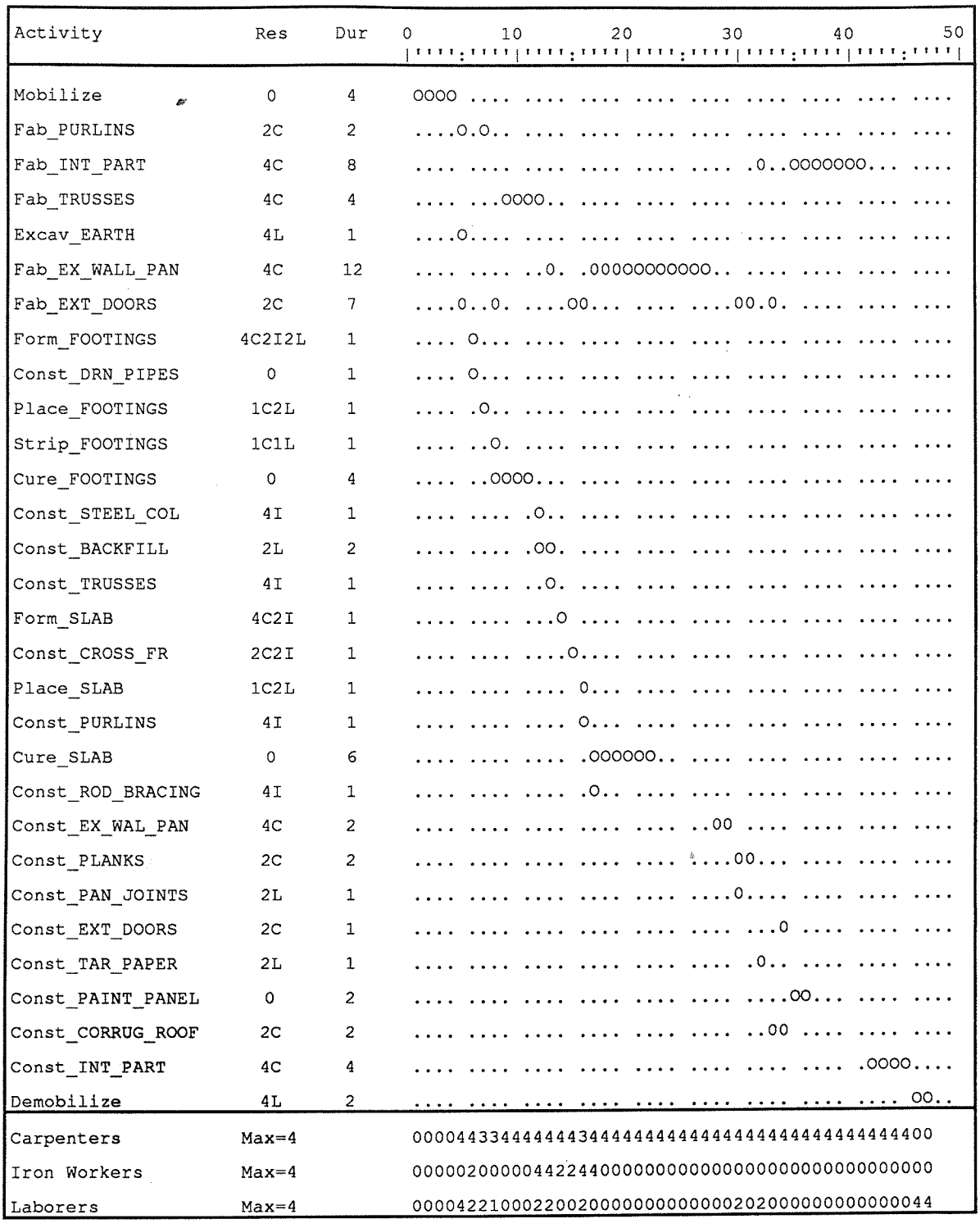

Figure 8.17 Extended Gantt Chart with Postponed Activities

9. CONCLUSIONS

The first three Sections of this chapter highlight ideas from previous chapters which help to define and support the contributions of the research listed in the fourth Section. The restrictions to network diagraming techniques (NDT), noted in Section 9.1., are discussed in detail in Chapter 2. Section 9.2. is a summary of Chapters 3 through 7. Section 9.3. compares ACP with other similar efforts. The final section of this Chapter suggests opportunities for future research.

9.1. OPPORTUNITIES FOR IMPROVING SCHEDULING TECHNIQUES

The ordering of activities within network techniques such as PERT and CPM is based on user defined precedence among activities. This precedence is clearly and concisely conveyed by traditional network diagrams. Milestones are also used in network techniques. Resource constraints are incorporated in the second stage of a two-stage approach. In this approach, resource limitations are imposed on the traditional network. This results in additional constraints among activities which are neither explicitly communicated to the user nor available for review—the black-box effect. Furthermore, NDTs are designed around the representation of precedence as a relationship between activities; the concept of conditions or states (such as weather) which constrain activities do not fit within the NDT approach.

The ordering of activities in existing expert system construction planners is derived from relationships among project components as well as from certain user-specified activity precedence. Traditional network techniques do not deduce activity precedence from component relationships; expert system construction planners developed to date do not base precedence on resources. Neither traditional network techniques nor expert system planners developed to date document for the user their bases for ordering of activities.

9.2. MODEL SUMMARY

9.2.1. Representation

The model explicitly represents the following:

- project components
- component relationships
- activities
- activity categories
- constraints
- status

The modular structure of ACP was chosen to facilitate the importing of project component and component relationship information from the database and the Computer-Aided Design system. This importing of information has not been implemented, but can be done using CIFECAD and PMAP [Ito 89] in the same way as two previous construction planners (see Section 2.3.3. and 2.3.4. for descriptions of OARPLAN and SIPEC). Activities are generated from the project components and additional attributes supplied by the scheduler (see Section 6.1.). All activities start in the TODO category, and as constraints are applied they are cycled through NEWCANDO, CANDO, DOING, ENDED, and when the project is complete they all reside in DONE. The use of constraints is discussed further below. The installed components, milestones, and resource usage are continuously documented in the project status.

9.2.2. Reasoning

The reasoning structure is iterative and chronological. The model uses a constraint-based approach, where it is assumed that all activities can be scheduled unless a constraint delays an activity's execution. Reasoning is directed by rules. These rules can be used to represent several kinds of constraints supported by the underlying activity and component representations which the scheduler can express in an *if-then* format. These constraints draw on the project status.

The model a) combines the bases for ordering activities from network techniques (other activities, milestones, and resources) and from expert system construction planners

(component relationships and activities), b) extends these bases to include activity categories and installed project components, and c) generalizes resources, installed components, and milestones into *status* where other conditions (e.g., weather, labor conditions) can also be documented. The identification of activity categories and installed components adds to the representation of a construction project. Inferred milestone *status* information provides a generic, new basis for project representation. Both extensions expand the basis for deducing precedence.

9.2.3. User Interface

The user interacts with the model in several ways:

- by defining English-like rules and then applying them to the activity list
- by monitoring the model as it iterates chronologically (forward and, if requested, backward)
- by overriding model decisions
- by dynamically querying the model about the activities, the status, and the rule trace, and finally
- by viewing the resulting Gantt chart

9.3. Comparison to Previous Efforts

This Section compares ACP to three somewhat parallel approaches and differentiates between them.

9.3.1. ACP versus NDTs

NDTs have the following goals:

- to analyze the timing (early and late start and finish times and float times) and resource implications of a project schedule developed by a human planner and
- to apply resource leveling techniques.

NDTs perform these functions very well, but require that all activities, sequence constraints, durations, and resources be explicitly defined by the user.

Since a major goal of ACP is to derive project activities and their precedence, it is imperative that ACP represent much more project information (see Section 9.2.1.). If this information can be obtained directly from a Computer-Aided Design or carried forward

from previous project schedules, then additional manual input is not necessary; in other situations, considerable input could be required. In addition the processing of this symbolic information is far slower than the conventional scheduling of primarily numeric data. So, if one's goal were only to obtain the output of NDTs, then ACP offers no advantage.

On the other hand, ACP was developed in order to generate plans (including precedence information) from first principles. ACP uses generic activity constraints which draw on the project components and their relationships, and determines sequencing (or precedence) of activities; this is a goal which is not and could not be achieved by NDTs, since they receive precedence information as input.

The current structure of ACP develops a schedule in a single pass. This precludes the opportunity for the system to learn which activities are "critical with infinite resources" or which resources are scarce, and then to use this information on a second pass in allocating constrained resources. Resource leveling techniques which use heuristics such as total float and late start are, in effect, using this two-pass approach to give priority to specific activities. However, ACP brings non-resource and resource constraints together in a one-stage process. This process is one-stage in the sense that at the beginning of a time period, both non-resource and resource constraints are considered rather than generating a non-resource constrained schedule and then further constraining this schedule with resources. It brings together all constraints and thereby eases their coordination.

The final difference between ACP and NDTs is that NDTs do not offer the scheduler information about why a particular precedence constraint exists or, in the case of resource constraints, when it was applied. ACP addresses these needs through English-like constraints and a query facility. Experimentation with ACP in the future will shed additional light on how the query facility can most appropriately be utilized.

9.3.2. ACP versus Expert Construction Planners

SIPEC and OARPLAN (see Sections 2.3.3. and 2.3.4.) are initiated by a goal such as "construct the project," then this goal is expanded into subgoals such as "construct first floor and second floor." These subgoals are similarly pursued and expanded as far as possible. The plan or project network is then generated from the subgoal satisfaction trace. Since this trace is the opposite to the order in which the work will be executed, it is reversed and a start node is used to tie together the trace extremities. This approach is quite

different than the one used by ACP. GHOST (see Section 2.3.2.) starts with a totally parallel network and uses critics to force precedence. Again, this is a different approach than the one used by ACP.

The major difference between these planners and ACP is due to ACP's representation of the temporal aspects of construction schedules. The planners derive an activity sequence, but do not attach this sequence to a time scale. (See Section 3.3. for some implications of this approach.)

9.3.3. ACP versus CONSTRUCTION PLANEX

PLANEX (see Section 2.3.6.) is the most comprehensive expert system for construction scheduling available today. Contrary to the above planners, it addresses the timing of activities. It also computes quantities, selects crews, calculates durations, and estimates costs; these capabilities are not included in ACP. Nonetheless, the emphasis of ACP is on generating the sequencing of activities from first principles, whereas PLANEX accesses predefined precedence relationships among activities. Finally, ACP levels resources, while PLANEX does not.

9.4. CONTRIBUTIONS

1. ACP considers resource and non-resource constraints concurrently. The two-stage approach which is currently used within NDT requires the user to do all the coordination between the network development stage and the resource leveling stage. If this task is to be automated, we must get the coordination out of the scheduler's mind and into a computer. The single-stage approach takes a step in this direction.

2. ACP determines the sequencing of activities based on constraints. These constraints are represented as heuristics or *if-then* rules. An advantage of this technique is that it is as easy to represent a precedence constraint which has a state (defined by exogenous variables, completion of groups of tasks, etc.) as its premise, as it is to represent a precedence constraint defined by the completion of another activity.

3. ACP addresses the need to record and communicate the fundamental basis for precedence through English-like rules and comments associated with these

rules. It also allows the scheduler to interrogate the system through the query facility.

9.5. FUTURE RESEARCH

This planning model opens many areas for future research. Automated assistants for all phases of construction planning are envisioned. Several obvious extensions to ACP include:

- more activity types
- more activity attributes
- expansion of the query facility to include resource constraints

More substantial changes such as the mapping of project components and resources to activities would also enhance the model. Selecting from alternative resource levels or crews and determining the durations of activities would also be major improvements.

Finally, the representation of generic construction methods as hierarchical accumulations of primitive actions, such as transport, position, fasten, or apply, would be a significant achievement. If ACP had the ability to reason about selecting methods or choosing between methods it would represent the construction scheduling domain far more robustly.

The above research, together with ACP's emphasis, will lead toward more accurate, more intelligent, and more easily automated construction planning and scheduling tools.

APPENDIX 1: GLOSSARY

action	A sequence of operations or a procedure.
activity	An activity has a definable beginning, ending, and therefore duration. The number of activities into which a project is divided is chosen by the scheduler. Generally an activity brings the completion of the project one step closer. Typically an activity applies a crew, using a method, in order to construct a project component.
assembly	A selection of one or more project components or other assemblies. Assemblies can be combined into a multi-level hierarchy; project components will be at the extremities of this hierarchy.
attribute	A feature of an object which can be described, e.g., color, supports.
characteristic	A fact which describes an object, e.g., color: red; supports: column X. A specific attribute and an associated value.
components	Elements and systems which when combined constitute the project.
constraints	Knowledge which causes the activity network to be nonparallel.
crew	A selection of one or more resources or other crews. In addition to having associated resources, a crew has attributes which describe its resource consumption and production. Crews can be combined into a multi-level hierarchy; resources will be at the extremities of this hierarchy.
critical path	The longest (based on time) path through the activity network. This path has the least float of all paths through the network.
end user	The person(s) who uses the schedule for monitoring purposes.
event	An event represents a change of state which occurs at a point in time with some significance. The beginning and end of an activity are events.
executable constraints	Executable knowledge which constrains the schedule (see sections 2.2.3. and 2.2.5.). Example: If gravity support is not available for project component X, then do not schedule the activity associated with component X.
executable knowledge	Knowledge which is in an executable form or algorithm, e.g., a function, a procedure, or an if-then rule.

facts Attributes and values.

hierarchy A set of objects which are connected by an is_a, or conversely a subclass_of, relationship.

knowledge Facts which have been refined, distilled, or organized into a more concise and usable form.

method A selection of one or more actions or other methods. Methods can be combined into a multi-level hierarchy; actions will be at the extremities of this hierarchy.

milestone An event which is of particular interest to the scheduler. Typically the completion of one or more related activities such as closing in a building.

network Traditional usage is a representation of activities and sequence constraints among activities which the scheduler currently believes will be the most constraining (see Section 2.2.7.).

objects Entities which can have associated facts, e.g. project components, activities.

planning Determining the project activities and, by selecting methods, determining the order of these activities.

precedence Unmodified, precedence refers collectively to all project sequence constraints. Precedence between two activities is synonymous with a sequence constraint between the same activities.

project 1. A physical structure or facility.
2. The construction of a physical structure or facility.

project state or condition One or more project characteristics, e.g., Completed_Project_Components: FOOTING_A FOOTING_B FOOTING_C COLUMN_A; Weather: FINE; Carpenters_Available: 4; Day_of_the_Week: FRIDAY.

relationships A physical or conceptual connection between two objects. Relationships are special combinations of facts. Relationships between objects result when one object has another object as one of its attribute values. Example:

object name: column X
characteristic: supports
value: beam_X1 beam_X2

representation The formulation of data, information, or knowledge into a physical symbol system (e.g., English, PASCAL, or LISP; see Rich [83 p.3] for definition) for the purpose of communication, manipulation, or reasoning.

resource Generally, anything which is employed or consumed in order to reach the goal of completing a project. Typically workers, temporary and permanent materials, and equipment.

schedule or plan A model which dynamically describes the expected activities and events which, when completed, signify the project's completion.

scheduler or planner The person(s) who develops the original schedule or network diagram.

scheduler or planner v.s. end user This differentiation is intentionally made because the available software is just becoming user friendly enough for a manager also to be the scheduler. It is a goal of this research to overcome this problem and gain the benefits of the scheduler and the end user being the same person.

scheduling Determining the timing of the project activities. Resources choices and resource quantities are inherently tied to methods and durations; this in turn, ties resources to planning and scheduling. Therefore, if the results of planning and scheduling produce an unacceptable project duration and a new resource selection is chosen to rectify this problem, then a new planning and scheduling cycle may be necessary.

slot The location of a characteristic within the representation of an object.

symmetric matrix A matrix in which each i-j item is equal to the corresponding j-i item.

this research The document you are currently reading and the computer system which it describes.

values A constant or object which defines the state of an attribute, e.g., red, 42, column X.

workspace The APL execution environment in which computation takes place and in which names have meaning. A workspace contains the variables, functions, and control information for an APL session. A repository for a collection of functions and data. [STSC 88 p.GL-41]

APPENDIX 2: MENUS

The figures in this Appendix show the hierarchy of all menus within ACP. This hierarchy is provided as a roadmap for the system user and as a guide for the reader in chapters 5 through 8.

To move from left to right in Figure A2.1 the user selects from a menu. Typically, to exit from a submenu to the previous menu (i.e., move from right to left in Figure A2.1), the user presses key F10. An exception to this rule, occurs when exiting from a second level menu (i.e., a module) back to ACP; here the user types an uppercase "Q" for Quit.

Figure A2.2 and Figure A2.3 respectively show the relationship menus and the query facility menus. These menus are subject to the same rules as Figure A2.1.

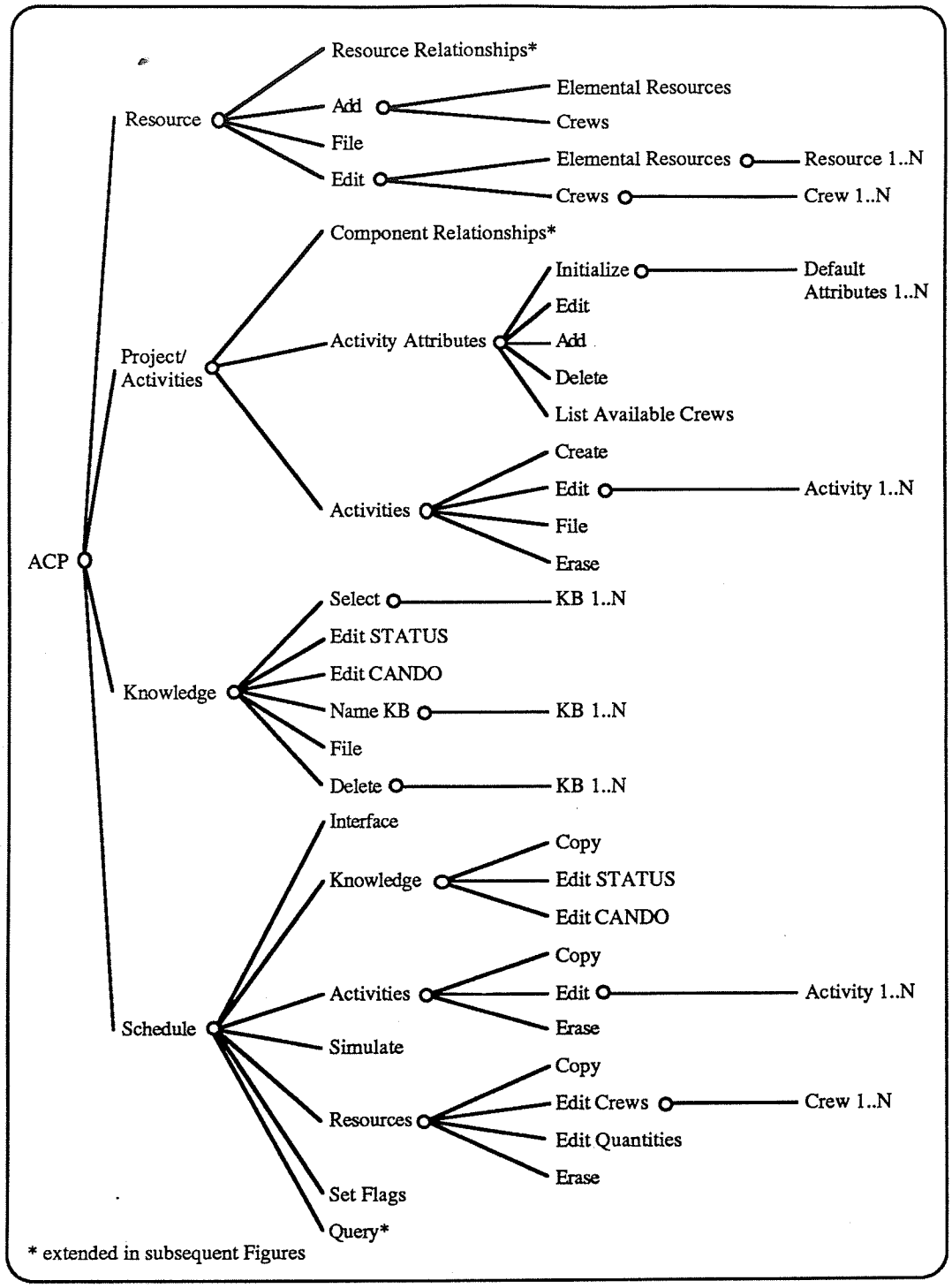

Figure A2.1 System Menus

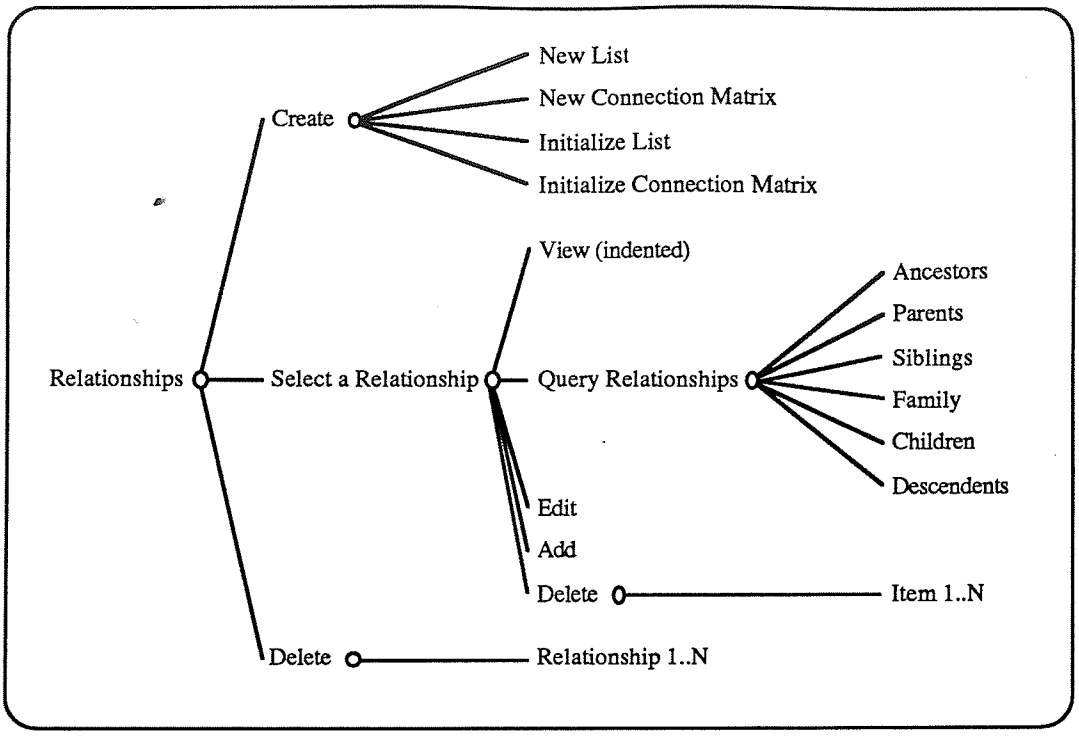

Figure A2.2 Relationship Menus

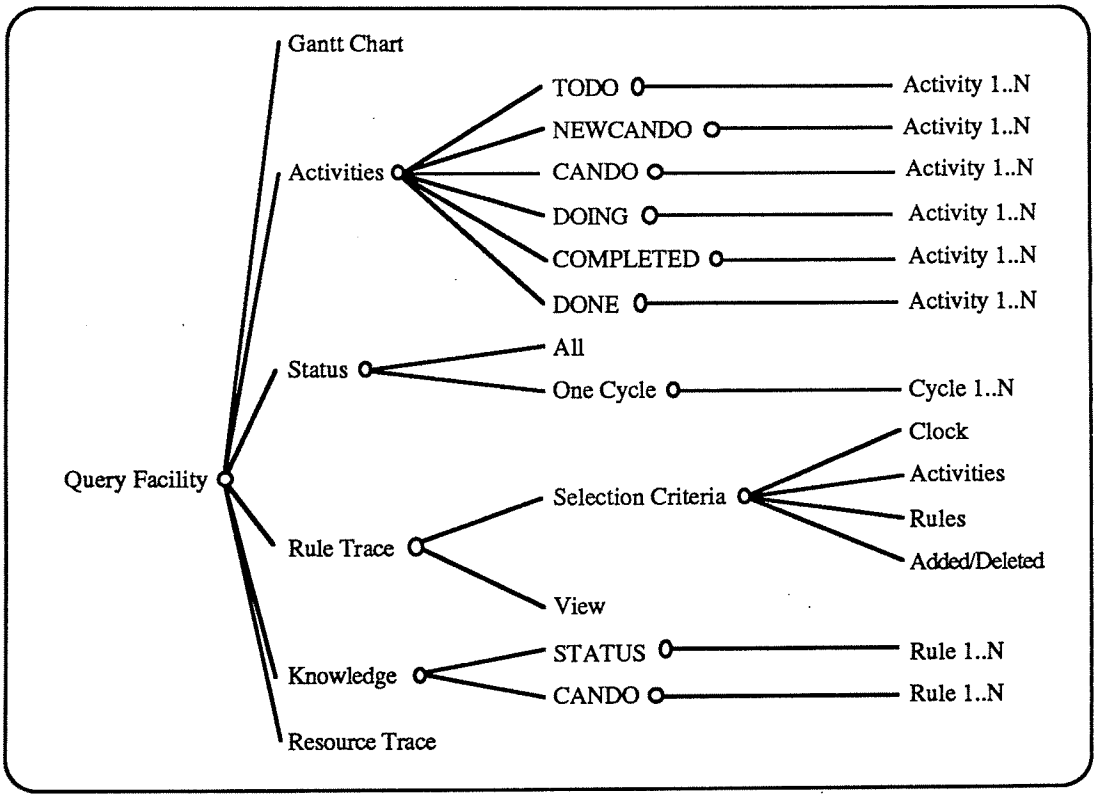

Figure A2.3 Query Menus

APPENDIX 3: IMPLEMENTATION

A3.1. OVERVIEW

The system has been implemented on an Intel 80286 microprocessor under DOS and can be set up using an autoexec.bat file so that it will automatically be loaded when the computer is turned on.

The system is implemented in APL, which is well known for its ability to represent and manipulate numeric and character arrays [Gilman 84; Peelle 86], and has been used more recently for expert system applications [Alfonseca 89; Broadbent 89; Engelmann 89; Eusebi 86; Hagamen 89; Jantzen 89; Rodriguez 89; Sullivan 86; Surkan 89]. In APL, the execution environment is called a workspace; functions and variables are contained within workspaces, but can be temporarily stored in files. Version 8.0 of STSC APL*PLUS [STSC 88] was used for this implementation.

The name ACP is used for the main workspace since entry to the system must be via this workspace and since it plays a central role in the system. In the text where ambiguities could exist, the term *ACP workspace* will be used to avoid confusion between the workspace name and the system name.

The five modules contained in the system are listed in Figure A3.1.

Module	Chapter
Project Description	5
Resource	5
Scheduling Knowledge	5
Activities	6
Scheduling	7

Figure A3.1 System Modules

Two of these modules, Project Description and Activities, have been combined into one workspace since they both need access to the relationships between project components. This results in Figure A3.2 and the four choices from the top level menu in Figure A2.1.

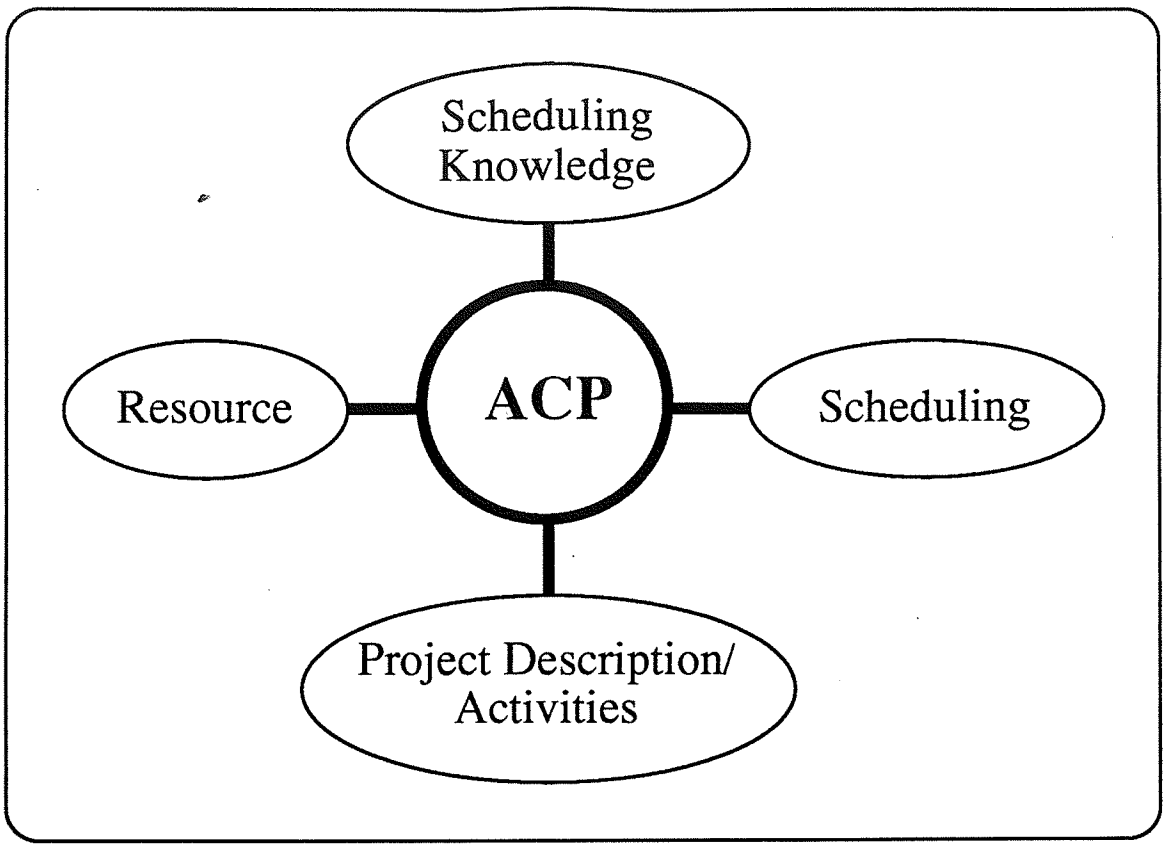

Figure A3.2 System Workspaces

The Resource, Project, and Scheduling Knowledge modules are used to select, enter, assemble, and manipulate input. The Activity module uses the project components, component relationships, crews, and activity attributes to create project activities. The Resource, Activities, and Scheduling Knowledge modules each record their refined information in temporary files. These files are retrieved by the Scheduling module which then generates the order and timing of the activities.

The temporary files are called TEMPA.ASF, TEMPR.ASF, and TEMPK.ASF for Activities, Resources, and scheduling Knowledge and have the common extension .ASF which stands for APL System File. These files include only the information which the Scheduling module uses from each of the other workspaces. The purpose of the temporary files is to speed access to this information; it is equally possible to copy the information into the Scheduling workspace from the other workspaces, but it takes longer.

In order to record various resources and resource combinations, the user can store as many versions of the resource workspace as desired and still maintain access to them all from the ACP menu. This is also true for project description and activities workspaces.

As noted at the end of Chapter 5, the scheduling knowledge bases are all stored within the scheduling knowledge workspace.

A3.2. REPRESENTATION

APL workspaces include functions and variables. The next section describes the use of functions; this section describes variables.

There are two types of variables—numeric and character. N-dimensional arrays of numbers or characters can be represented. The terms scalar, vector, and matrix are used to refer to variables with 0, 1, and 2 dimensions respectively. A single number or a single character is a scalar. Digits can be stored as numbers or characters, but cannot be mathematically manipulated as characters.

Boolean matrices are used to represent relationships as described in Section 5.1. and shown in Figure 5.1. Gantt charts are also stored as Boolean matrices.

Lists of names are usually stored as character matrices. Examples include project components (see Figure 5.1), elemental resources (Figure 5.2), and activity categories (Figure 4.1). Figure 6.2 shows four lists and three numeric vectors combined as a character matrix.

Typical examples of the most structured character matrices used in this implementation are shown in figures 5.3 and 6.1. These crew and activity representations are similar to frames which were suggested and originally described by Minsky [75] and used in OARPLAN [Darwiche 88], PIPPA [Marshall 87], and SIPEC [Kartam 89]. The similarity of these structured objects to frames arises from the concept of slots and values. On the other hand, these structured objects do not have attached procedures to perform tasks such as restricting possible values of slots, finding default values, or searching for values through inheritance. Such procedures are typical of frames.

Rules are a final character matrix representation which require special note. These are written in English-like terms (see Section 5.3.), the terms are converted into APL code using the vocabulary described by Figure 5.4, a header is added, and then the character matrix is converted into a function by a system function called *define*.

A3.3. REASONING

APL has three types of functions—system functions, primitive functions, and user-defined functions. System functions are described in the STSC APL*PLUS manual [STSC 88] and primitive functions are described in references on APL [Gilman 84; Peelle 86]. These two types of functions are used within user-defined functions.

The system menu hierarchy shown in Appendix 2 is consistent with the function hierarchy. For example in the scheduling workspace, there are seven options from the main menu, each choice leads to a further menu or allows the user to execute a selection. Here each menu item corresponds to an APL function.

The most complicated and interesting function in the scheduling workspace in *simulate* which, when executed, generates a schedule. The remainder of this Appendix is devoted to a synopsis of the *simulate* function. The purpose of this overview is to give the interested reader a more in-depth view of function organization and variable representation within the system.

simulate calls three other functions—*siminit, sim,* and *simend. siminit* initializes the trace variables, the activity category variables, the clock, the status, the Gantt chart, asks the scheduler to set the priorities, and resets the activity durations to their initial values. *simend* informs the scheduler it there were activities which were not scheduled and deletes various temporary variables.

The function *sim,* then, does most of the work. This function is shown in Figure A3.3 in keyword mode without comments. The APL code associated with flags is described in Section 7.5.1. and has been omitted for simplicity. Words which consist of all uppercase letters are variables or line labels; all other words are functions. All functions which begin with a # symbol are primitive APL functions; I defined the remaining functions. The paragraphs below briefly describe the functions in, and the operation of *sim.* One cycle through *sim* corresponds to one cycle through Figure 4.1.

```
  sim
BEGIN: output 1

 addtoSTATUS effects ENDED
 upSTATUS
 addtoDONE ENDED
 delfmCANDO ENDED
 NEWCANDOP #is priorget NEWCANDO #is cando
 CANDO #is CANDO prioritize NEWCANDO
 DOING #is choose CANDO
 delfmTODO NEWCANDO

 REMDUR #is DOING durget 1
 output 2
 #goto (empty DOING)/END
 uptime
 DOING duradd REMDUR #is REMDUR-TICK
 resused
 resrplc DOING

 ENDED #is DOING[where 0=REMDUR;]
 DOING #is DOING[where 0 #ne REMDUR;]
 TICK gantt DOING over ENDED
 #goto BEGIN

END:
```

Figure A3.3 The Function sim

:	indicates that the characters to its left are a line label.
(...)	force evaluation of the included statement.
/	select the number of items in its left argument from its right argument.
-	minus (applies to arrays as well as scalars).
=	equals (when used to compare a scalar to a vector, the result is Boolean and is the same length as the vector).
[L;C]	index into the Lines and or Columns of a matrix (if C is omitted, all Columns of the specified Lines are selected).
#is	assigns its right argument to its left argument.
#goto	branches to its right argument (a line label).
#ne	not equals (applied the same as equals).
output	is called twice each cycle and provides different information based on its right argument. All output information can be activated or suppressed via flags.
effects	gets from each activity in the ENDED list the value of its "Add to status" attribute. These effects are then added to the STATUS.
upSTATUS	updates the status using the status knowledge described in Section 5.3. The status knowledge checks for, and adds milestones which have been reached as a result of ENDED activities.
addtoDONE	adds to DONE the ENDED activities.
delfmCANDO	deletes from CANDO the ENDED activities.
cando	applies the cando knowledge (also described in Section 5.3.) in order to generate the list of NEWCANDO activities.
priorget	gets the priorities of the NEWCANDO list. This is then assigned to NEWCANDOP.
prioritize	accepts the two arguments CANDO and NEWCANDO; it combines these activity lists and reorders the new list according to the activities' priorities. This is then assigned to CANDO.
choose	chooses from CANDO the activities to be DONE this cycle.
delfmTODO	deletes from TODO the NEWCANDO activities.

Figure A3.4 *sim* Functions and Description

durget	gets the remaining durations of the DOING activities. (When *durget* is called in *siminit* with a right argument of 0 it gets the initial durations.) The remaining durations of the DOING activities are then assigned to REMDUR.
	The second portion of output is then displayed on the screen for the scheduler.
empty	checks to see if DOING contains any activities; it returns a value of 0 if not, a 1 if so. *sim* ends if the value is 1, but does not branch if the value is 0.
uptime	finds the minimum value in REMDUR and assigns it to TICK; it then increments the clock by this value. An option exists to always increment the clock by one time period.
	The REMDUR is then assigned the old REMDUR minus TICK.
duradd	opens each of the DOING activities and concatenates the current remaining duration to the Duration slot.
resused	documents the usage of resources for each time period.
resrplc	replaces the resources which were being used this cycle in the pool of available resources.
where	finds the indices of REMDUR which are equal to 0. These indices are then used to select the activities from within DOING which have 0 as their remaining duration. These activities are assigned to ENDED. Similarly the activities in DOING which have a remaining duration not equal to 0 are reassigned to DOING.
gantt	updates the Gantt chart by indicating that all the activities which are currently in either DOING or ENDED were being executed for the last TICK time periods.
	Finally *sim* branches back to BEGIN.

Figure A3.4 *sim* Functions and Description, continued

Obviously, there are many other functions which are called by the functions within *sim*. In total, the workspace contains 150 functions, of which 10 were not written by the author. Many of these functions are used for interface and development purposes. Most functions are far simpler than *sim*—typically containing a few lines of code.

APPENDIX 4: ACTIVITY REPRESENTATION

The use of activities in the schedule module is described in chapters 4, 7, and 8. The generation of activities and their attributes are described in Chapter 6. Although Chapter 6 provides the representation of activities in summary form, this Appendix displays each individual activity as it would be seen in ACP. As in Chapter 6, character strings within angle brackets refer to attribute values and generic activities are referred to by the initial portion of their name.

Name:	Const_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	<duration>
Supported_By:	<component> parents in support relationship
Add to Status:	<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	

Figure A4.1 Const Activity Representation

Name:	Form_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	(3/8)x<duration>
Supported_By:	<component> parents in support relationship
Add to Status:	FORM_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	

Figure A4.2 Form Activity Representation

Name:	Rebar_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	(2/8)x<duration>
Supported_By:	
Add to Status:	REBAR_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	DONE contains Form_<component>

Figure A4.3 Rebar Activity Representation

Name:	Conc_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	(1/8)x<duration>
Supported_By:	
Add to Status:	CONC_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	DONE contains Rebar_<component>

Figure A4.4 Conc Activity Representation

Name:	Cure_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	(1/8)x<duration>
Supported_By:	
Add to Status:	CURE_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	DONE contains Conc_<component>

Figure A4.5 Cure Activity Representation

Name:	Strip_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	(1/8)x<duration>
Supported_By:	
Add to Status:	<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	DONE contains Cure_<component>

Figure A4.6 Strip Activity Representation

Name:	Estab_Facil_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	2
Supported_By:	
Add to Status:	FACILITY_ESTABLISHED_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	DONE contains Mobilize

Figure A4.7 Estab_Facil Activity Representation

Name:	Manuf_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	10
Supported_By:	
Add to Status:	MANUFACTURED_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	DONE contains Estab_Facil_<component>

Figure A4.8 Manuf Activity Representation

Name:	Demo_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	<duration>
Supported_By:	
Add to Status:	DEMOLISHED_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	DONE contains Mobilize

Figure A4.9 Demo Activity Representation

Name:	Remove_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	2
Supported_By:	
Add to Status:	REMOVED_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	<temporary>
Cando Test:	STATUS contains _<temporary>

Figure A4.10 Remove Activity Representation

Name:	Excav_for_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	1
Supported_By:	
Add to Status:	EXCAVATED_FOR_<component>
Crew:	<crew>
Procure:	<procure>
Temporary:	
Cando Test:	DONE contains Mobilize

Figure A4.11 Excav_for Activity Representation

Name:	Mobilize
Components:	
Initial State:	0
Duration:	2
Supported_By:	
Add to Status:	MOBILIZED
Crew:	CREW_0
Procure:	0
Temporary:	
Cando Test:	

Figure A4.12 Mobilize Activity Representation

Name:	Demobilize
Components:	
Initial State:	0
Duration:	2
Supported_By:	
Add to Status:	DEMOBILIZED
Crew:	CREW_0
Procure:	0
Temporary:	
Cando Test:	

Figure A4.13 Demobilize Activity Representation

Name:	Procure_<component>
Components:	<component>
Initial State:	<initial state>
Duration:	15
Supported_By:	
Add to Status:	PROCURED_<component>
Crew:	CREW_0
Procure:	<procure>
Temporary:	
Cando Test:	

Figure A4.14 Procure Activity Representation

REFERENCES

[Adams 87] Adams, J.R., and Martin, M.D., *Professional Project Management: A Practical Guide*, Universal Technology Corporation, Dayton, OH, 1987.

[Adrian 85] Adrian, J.J., *Microcomputers in the Construction Industry*, Reston Publishing Company, Inc., Reston, VA, 1985.

[Alfonseca 89] Alfonseca, M., "Object Oriented Programming in APL2," *APL QuoteQuad*, Vol. 19, No. 4, 1989.

[Allam 88] Allam, S.I.G., "Multi-Project Scheduling: A New Categorization for Heuristic Scheduling Rules in Construction Scheduling Problems," *Construction Management and Economics*, Vol. 6, No. 2, 1988.

[Anderson 87] Anderson, S.D. and Woodhead, R.W., *Project Manpower Management: Decision-Making Processes in Construction Practice*, John Wiley & Sons Inc., New York, NY, 1987.

[Antill 82] Antill, J.M., and Woodhead, R.W., *Critical Path Methods in Construction Practice*, 3rd ed., Wiley-Interscience, New York, NY, 1982.

[Assad 86] Assad, A.A., and Wasil, E.A., "Project Management using a Microcomputer," *Computers and Operations Research*, Pergamon Press, Oxford, England, Vol. 13, No. 2/3, 1986.

[AutoCAD 88] AutoCAD Reference Manual, *Autodesk, Inc.*, Sausalito, CA, July 1988.

[Barrie 84] Barrie, D.S., and Paulson, B.C., Jr., *Professional Construction Management*, 2nd ed., McGraw-Hill Book Company, New York, NY, 1984.

[Bergen 86] Bergen, S.A., *Project Management: An Introduction to Issues in Industrial Research and Development*, Basil Blackwell, Oxford, UK, 1986.

[Birrell 87] Birrell, G.S., discussion of "Criticism of CPM for Project Planning Analysis," by A. Jaafari, *Journal of Construction Engineering and Management*, Vol. 113, No. 2, June, 1987.

[Broadbent 89] Broadbent, H.A., and Lucas, J., "A Neural Network Model of Serial Learning," *APL QuoteQuad*, Vol. 19, No. 4, 1989.

[Cormican 85] Cormican, D., *Construction Management: Planning and Finance*, Construction Press, London, UK, 1985.

[Crowston 67] Crowston, W., and Thompson, G.L., "Decision CPM: A Method for Simultaneous Planning, Scheduling and Control of Projects," *Operations Research*, Vol. 15, No. 3, May-June, 1967.

[Currie 85] Currie, K., and Tate, A., "O-Plan—Control in the Open Planning Architecture," *AIAI-TR-12*, Artificial Intelligence Applications Institute, University of Edinburgh, UK, 1985.

[Darwiche 88] Darwiche, A., Levitt, R.E., and Hayes-Roth, B., "OARPLAN: Generating Project Plans in a Blackboard System by Reasoning about Objects, Actions and Resources," *Artificial Intelligence in Engineering Design, Analysis and Manufacturing*, Vol. 2, No. 3, 1988. (OARPLAN)

[Dym 91] Dym, C.L., and R.E. Levitt, *Knowledge-Based Systems in Engineering*, McGraw-Hill, In Press, 1991.

[Easa 89] Easa, S.M., "Resource Leveling in Construction by Optimization," *Journal of Construction Engineering and Management*, Vol. 115, No. 2, June, 1989.

[Echeverry 89] Echeverry, D., Ibbs, C., and Simon, K., "Generation of Construction Schedules A Knowledge Based Approach," *Proceedings of the Sixth International Symposium on Automation and Robotics in Construction*, San Francisco, CA, June, 1989.

[Elmaghraby 77] Elmaghraby, S.E., *Activity Networks*, John Wiley & Sons, New York, NY, 1977.

[Engelmann 89] Engelmann, U., Gerneth, T., and Meinzer, H.P., "Implementation of Predicate Logic in APL2," *APL QuoteQuad*, Vol. 19, No. 4, 1989.

[Eusebi 86] Eusebi, E., and Brown, J.A., "APL2 and AI: A Study of Search," *APL Quote Quad*, Vol. 16, No. 4, 1986.

[Fabrycky 84] Fabrycky, W.J., Ghare, P.M., and Torgersen, P.E., *Applied Operations Research and Management Science*, Prentice-Hall, Inc., Englewood Cliffs, NY, 1984.

[Fikes 71] Fikes, R.E., and Nilsson, N.J., "STRIPS: a New Approach to the Application of Theorem Proving to Problem Solving," *Artifical Intelligence*, Vol. 2, No. 3, 1971.

[Fondahl 61 62] Fondahl, J.W., "A Noncomputer Approach to the Critical Path Method for the Construction Industry," *Technical Report No. 9*, The Construction Institute, Department of Civil Engineering, Stanford University, Stanford, CA, 1st ed. 1961, 2nd ed., 1962.

[Fondahl 68] Fondahl, J. W., "Let's Scrap the Arrow Diagram," *Western Construction*, Aug., 1968.

[Fondahl 75] Fondahl, J.W., "Some Problem Areas in Current Network Planning Practices and Related Comments on Legal Applications," *Technical Report No. 193*, The Construction Institute, Department of Civil Engineering, Stanford University, Stanford, CA, Apr., 1975.

[Fox 87] Fox, M.S., *Constraint-Directed Search: A Case Study of Job-Shop Scheduling*, Morgan Kaufmann Publishers, Inc., Los Altos, CA, 1987.

[Gilman 84] Gilman, L., and Rose, A.J., *APL: An Interactive Approach*, 3rd ed., John Wiley & Sons Inc., New York, NY, 1984.

[Gray 86] Gray, C., "Intelligent Construction Time and Cost Analysis," *Construction Management and Economics*, Vol. 4, No. 2, Aut., 1986.

[Gupta 85] Gupta, S., and Taube, L.R., "A State of the Art Survey of Research on Project Management," *Project Management: Methods and Studies*, B.V. Dean, ed., North-Holland, Amsterdam, 1985.

[Hagamen 89] Hagamen, W., Berry, P.C., Iverson, K.E., and Weber, J.C., "Processing Natural Language: Syntactic and Semantic Mechanisms," *APL QuoteQuad*, Vol. 19, No. 4, 1989.

[Halpin 76] Halpin, D.W., and Woodhead, R.W., *Design of Construction and Process Operations*, John Wiley & Sons Inc., New York, NY, 1976.

[Harris 78] Harris, R.B., *Precedence and Arrow Networking Techniques for Construction*, John Wiley & Sons, New York, NY, 1978.

[Harris 83] Harris, F., and McCaffer, R., *Modern Construction Management*, 2nd ed., Granada, London, UK, 1983.

[Hendrickson 87] Hendrickson, C., Martinelli, D., and Rehak, D., "Hierarchical Rule-Based Activity Duration Estimation," *ASCE Journal of Construction Engineering and Management*, Vol. 113, No. 2, June, 1987. (MASON)

[Hendrickson 89] Hendrickson, C., and Au, T., *Project Management for Construction*, Prentice Hall, Englewood Cliffs, NJ, 1989.

[Howard 89] Howard, H.C., Levitt, R.E., Paulson, B.C., Pohl, J.G., and Tatum, C.B., "Computer Integration: Reducing Fragmentation in the AEC Industry," *Journal of Computing in Civil Engineering*, Vol. 3, No. 1, Jan., 1989.

[Ito 89] Ito, K., Ueno, Y., Levitt, R.E., and Darwiche, A., "Linking Knowledge-Based Systems to CAD Design Data with an Object-Oriented Building Product Model," *CIFE Technical Report No. 17*, Center for Integrated Facilities Engineering, Stanford University, Stanford, CA, Aug., 1989. (CIFECAD)

[Jaafari 84] Jaafari, A., "Criticism of CPM for Project Planning Analysis," *Journal of Construction Engineering and Management*, Vol. 110, No. 2, June, 1984.

[Jantzen 89] Jantzen, J., "Inference Planning Using Digraphs and Boolean Arrays," *APL QuoteQuad*, Vol. 19, No. 4, 1989.

[Kalk 80] Kalk, A., "Insight: Interactive Simulation of Construction Operations using Graphical Techniques," *Technical Report No. 238*, The Construction Institute, Department of Civil Engineering, Stanford University, Stanford, CA, Dec., 1980.

[Kartam 89] Kartam, N.A., "Investigating the Utility of Artificial Intelligence Techniques for Automatic Generation of Construction Project Plans," thesis presented to Stanford University, Stanford, CA, in 1989, in partial fulfillment of the requirements for the degree of Doctor of Philosophy. (SIPEC)

[Kelly 61] Kelly, J., "Critical Path Planning and Scheduling: Mathematical Basis," *Operations Research*, Vol. 9, No. 3, May-June, 1961.

[Lee 85] Lee, S.M.,and Olson, D.L., "Project Scheduling for Multiple Objectives," *Project Management: Methods and Studies*, B.V. Dean, ed., North-Holland, Amsterdam, 1985.

[Lester 82] Lester, A., *Project Planning and Control*, Butterworths, London, UK, 1982.

[Levitt 85] Levitt, R.E., and Kunz, J.C., "Using Knowledge of Construction and Project Management for Automated Schedule Updating," *Project Management Journal*, Vol. 26, No. 5, Dec., 1985.

[Levitt 87] Levitt, R.E., and Kunz, J.C., "Using Artifical Intelligence Techniques to Support Project Management," *AI EDAM*, Vol. 1, No. 1, 1987.

[Levitt 88] Levitt, R.E., Kartam, N.A., and Kunz, J.C., "Artificial Intelligence Techniques for Generating Construction Project Plans," *Journal of Construction Engineering and Management*, Vol. 114, No. 3, Sep., 1988.

[Levy 87] Levy, S.M., *Project Management in Construction*, McGraw-Hill Book Company, New York, NY, 1987.

[Lichtenberg 74] Lichtenberg, S., *Project Planning: A Third Generation Aproach*, Polyteknisk Forlag, Copenhagen, Denmark, 1974.

[MacProject II 87] MacProject II Manual, *Claris Corporation*, Mountain View, CA, 1987.

[Malcolm 59] Malcolm, D.G., Roseboom, J.H., Clark, C.E., and Fazar, W., "Applications of a Technique for R and D Program Evaluation," *Operations Research*, Vol. 7, No. 5, 1959.

[Marshall 59] Marshall, A.W., and Meckling, W.H., "Predictability of the Costs, Time and Success of Development," *Report P-1821,* RAND Corporation, Dec., 1959.

[Marshall 87] Marshall, G., Barber, T.J., and Boardman, J.T., "Methodology for Modelling a Project Management Control Environment," *IEE Proceedings*, Vol. 134, No. 4, July, 1987. (PIPPA)

[Minsky 88] Minsky, M.L., *The Society of Mind*, Simon and Schuster Inc., New York, NY, 1988.

[Moder 83] Moder, J.J., Phillips, C.R., and Davis, E.W., *Project Management with CPM, PERT, and Precedence Diagramming*, 3rd ed., Van Nostrand Reinhold Company, New York, NY, 1983.

[Moder 85] Moder, J.J., and Crandall, K.C., "Precedence Diagramming: Time Computations, Anomalies and Remedies," *Project Management: Methods and Studies*, B.V. Dean, ed., North-Holland, Amsterdam, 1985.

[Navinchandra 88] Navinchandra, D., Sriram, D., and Logcher, R.D., "GHOST: Project Network Generator," *Journal of Computing in Civil Engineering*, Vol. 2, No. 3, July, 1988. (GHOST)

[Newell 63] Newell, A., and Simon, H.A., "GPS, A Program That Simulates Human Thought," *Computers and Thought*, E.A. Feigenbaum and J. Feldman, eds, McGraw-Hill, New York, NY, 1963.

[Paulson 75] Paulson, B.C., Jr., "Continuing Research in the Development of Interactive Man-Computer Systems for Engineering-Construction Projects," *Technical Report No. 200*, The Construction Institute, Department of Civil Engineering, Stanford University, Stanford, CA, Sep., 1975.

[Paulson 78] Paulson, B.C., Jr., "Interactive Graphics for Simulating Construction Operations," *Journal of the Construction Division*, ASCE, Vol. 104, No. C01, Mar., 1978.

[Peck 62] Peck, M.J., and Scherer, F.M., *The Weapons Acquisition Process: An Economic Analysis*, Division of Research, Graduate School of Business Administration, Harvard University, Cambridge, MA, 1962.

[Peelle 86] Peelle, H.A., *APL: An Introduction*, Holt, Rinehart and Winston, New York, NY, 1986.

[Peer 74] Peer, S., "Network Analysis and Construction Planning," *Journal of the Construction Division,* ASCE, Vol. 100, No. C03, Sep. 1974.

[Primavera 88] Primavera Project Planner Manual, *Project Management and Control Software*, Version 3.2, Primavera Systems, Inc., Bala Cynwyd, PA, 1988.

[Pritsker 66] Pritsker, A.A.B., and Happ, W.W., "GERT: Graphical Evaluation and Review Technique, Part I: Fundamentals," *Journal of Industrial Engineering*, Vol. 17, No. 5, May, 1966.

[Rich 83] Rich, E., *Artificial Intelligence*, McGraw-Hill Book Company, New York, NY, 1983.

[Ritchie 85] Ritchie, E., "Network Based Planning Techniques: A Critical Review of Published Developments," *Further Developments in Operational Research*, G.K. Rand and R.W. Eglese ed., Pergmon Press, Oxford, UK, 1985.

[Rodriguez 89] Rodriguez, P., Rojas, J., Alfonseca, M., and Burgos, J.I., "An Expert System in Chemical Synthesis Written in APL2/PC," *APL QuoteQuad*, Vol. 19, No. 4, 1989.

[Sacerdoti 74] Sacerdoti, E.D., "Planning in a Hierachy of Abstraction Spaces," *IJCAI-73*, Palo Alto, CA, 1973. (ABSTRIPS)

[Sacerdoti 75] Sacerdoti, E.D., "The Non-Linear Nature of Plans," *IJCAI-75*, Tbilisi, USSR, 1975. (NOAH)

[Sacerdoti 77] Sacerdoti, E.D., *A Structure for Plans and Behavior*, Ellsevier North-Holland, Inc., New York, NY, 1977.

[Sathi 85] Sathi, A., and Fox, M.S., "Representation of Activity Knowledge for Project Management," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 7, No. 5, Sep., 1985.

[Seabee circa 1960] "Seabee Planner's and Estimator's Handbook," NavDocks-P-405, Department of the Navy, Bureau of Yards and Docks, Washington 25, DC, circa 1960.

[Simons 88] Simons, K.L., Thornberry, H.L., and Wichard, D.A., "Simulation System for Construction Planning and Scheduling," presented at the Sep. 25-29, 1988, Joint ASME/IEEE Power Generation Conference, held at Philadelphia, PA.

[Stevens 90] Stevens, J.D., *Techniques for Construction Network Scheduling*, McGraw-Hill Publishing Company, New York, NY, 1990.

[STSC 88] STSC APL*PLUS Reference Manual, *STSC, Inc.*, Rockville, MD, 1988.

[Sullivan 86] Sullivan, Gerald, and Fordyce, K., "A Boolean Array Based Algorithm in APL for Forward Chaining in Rule-Based Production Expert Systems," *APL Quote Quad*, Vol. 16, No. 3, 1986.

[Surkan 89] Surkan, A., "APL Descriptions of Functional Building Blocks for Connectionist Computer Models," *APL QuoteQuad*, Vol. 19, No. 4, 1989.

[Tamimi 88] Tamimi, S., and Diekmann, J., "Soft Logic in Network Analysis," *Journal of Computing in Civil Engineering*, Vol. 2, No. 3, July, 1988.

[Tate 75] Tate, A., "Interacting Goals and Their Use," *IJCAI-75*, Tbilisi, USSR, 1975. (INTERPLAN)

[Tate 76] Tate, A., "Project Planning Using a Hierarchial Non-Linear Planner," *Report No. 25*, Department of Artificial Intelligence, University of Edinburgh, UK, Aug., 1976. (NONLIN)

[Tate 77] Tate, A., "Generating Project Networks," *IJCAI-77*, Boston, MA, 1977.

[Tate 85] Tate, A., "A Review of Knowledge-Based Planning Techniques," *Knowledge Engineer's Review*, British Computer Society Specialist Group on Expert Systems, Vol. 1, No. 2, June, 1985.

[Teicholz 63] Teicholz, P.M., *A Simulation Approach to the Selection of Construction Equipment*, thesis presented to Stanford University, Stanford, CA, in 1963, in partial fulfillment of the requirements for the degree of Doctor of Philisophy.

[Tenah 85] Tenah, K.A., and Guevara, J.M., *Fundamentals of Construction Management and Organization*, Reston Publishing Co., Inc., Reston, VA, 1985.

[Vere 83] Vere, S., "Planning in Time: Windows and Durations for Activities and Goals," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 5, No. 3, May, 1983. (DEVISER)

[Vere 83] Vere, S.A., "Planning in Time: Windows and Durations for Activities and Goals," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 5, No. 3, May, 1983.

[Walker 59] Walker, M.R., and Sayer, J.S., *Project Planning and Scheduling*, Report 6959, E.I. duPont de Nemours and Co., Wilmington, DE, Mar., 1959.

[Waugh 88a] Waugh, L.M., *Construction Planning Research Status Report*, unpublished report submitted to Professor R.E. Levitt, Department of Civil Engineering, Stanford University, Mar., 1988.

[Waugh 88b] Waugh, L.M., *A Construction Planner*, unpublished Ph.D. Proposal, Department of Civil Engineering, Stanford University, Stanford, CA, May, 1988.

[Waugh 89a] Waugh, L.M., "Knowledge-Based Structural Concrete Scheduling," presented at the Mar. 20-21, 1989 CSCE/CPCA Structural Concrete Conference, held at Montreal, Canada.

[Waugh 89b] Waugh, L.M., and Levitt, R.E., "Computerized Construction Scheduling," presented at the June 8-10, 1989 Annual Conference, Canadian Society for Civil Engineering, held at Saint John's, NFLD, Canada.

[Waugh 89c] Waugh, L.M., "Knowledge-Based Construction Scheduling," *Proceedings of the Sixth Conference on Computing in Civil Engineering and Symposium on Expert Systems in Civil Engineering*, Atlanta, GA, Sep. 11-13, 1989.

[Whiteman 88] Whiteman, W.E., and Irwig, H.G., "Disturbance Scheduling Technique for Managing Renovation Work," *Journal of Construction Engineering and Management*, Vol. 114, No. 2, June, 1988.

[Wiest 77] Wiest, J.D., and Levy, F.K., *A Management Guide to PERT/CPM*, 2nd ed., Prentice-Hall, Inc., Englewood Cliffs, NJ, 1977.

[Wiest 85] Wiest, J.D., "Gene-Splicing PERT and CPM: The Engineering of Project Network Models," *Project Management: Methods and Studies*, B.V. Dean, ed., North-Holland, Amsterdam, 1985.

[Willis 86] Willis, E.M., *Scheduling Construction Projects*, John Wiley & Sons, New York, NY, 1986.

[World Book 77] World Book Dictionary, *Field Enterprises Educational Corporation*, Chicago, IL, 1977.

[Zozaya 88] Zozaya-Gorostiza, C., and Hendrickson, C.T., "Knowledge-Based Planning for Construction Projects," *Report No. R-88-170*, Department of Civil Engineering, Carnegie-Mellon University, Pittsburg, PA, Apr., 1988. (CONSTRUCTION PLANEX)