Automation and Robotics in Construction XI
D.A. Chamberlain (Editor)

Automated Case-Based Scheduling for Power Plant Boiler Erection: Use of Annotated Schedules

I.D. Tommelein[a] and R.-J. Dzeng[b]

[a, b] Civil and Environmental Engineering Department,
The University of Michigan, Ann Arbor, MI 48109-2125, USA

Abstract

Contractors who repeatedly build the same kind of facilities accumulate experience in scheduling the needed construction work. When parts of a facility's design are copied from one project to the next, the previously developed schedules could be reused to schedule future work. The aim of the presented research is to articulate project characteristics and describe associated schedules, to accomplish such reuse. This paper describes the need for annotating a schedule with constraints pertaining to the facility design and construction resources, in addition to providing traditional scheduling data. It also presents a case-based reasoning system, called CasePlan. CasePlan automates the generation of construction schedules for power plant boiler erection by reusing annotated schedules. Power plant boilers have a more or less standardized design, notwithstanding design variations to optimize combustion of the fuel used in each plant. A boiler product model serves as the basis for comparing designs and assessing their similarities, in order to select the associated construction schedules for reuse. This automated scheduler tries to mimic the scheduling activity of human planners. Its case-based reasoning capability is expected to be useful to support robot planners which could learn from experience as people do through the use of cases. Case-based reasoning avoids the need to encode fundamental theories of physics and default logic that are otherwise required by automated schedulers that plan from scratch.

1. INTRODUCTION

Contractors who repeatedly build the same kind of facility accumulate experience in scheduling the needed construction work. When parts of a facility's design are copied from one project to the next, the corresponding parts of previously developed schedules could possibly be reused to schedule future work. The aim of the present work is to articulate project characteristics and describe associated schedules, to facilitate such reuse.

This paper describes parts of CasePlan, a case-based reasoner [9] that automates the generation of construction schedules for power plant boiler erection. Essential issues that must be addressed by CasePlan are: (1) Which schedule should be chosen for reuse? and (2) How can the chosen schedule be reused? The present discussion focuses on the expressiveness and reusability of information represented in construction schedules.

2. BACKGROUND

Contractors typically plan construction work by generating a schedule from scratch before the project starts. The resulting *planned schedule* must be updated as construction progresses and it becomes the *as-built schedule* upon project completion. For a given project, the planned and the as-built schedule can differ substantially, as unforeseen conditions arise and changes must be dealt with throughout the duration of construction. It is not a-priori clear, however, which of those two schedules lends itself better for reuse.

2.1. Planned Schedule

Traditional scheduling tools that use the critical path method (CPM) represent only the effects of applying constraints that govern schedule generation. Research using artificial intelligence programming techniques to automate the schedule generation task (e.g., [2, 7]) heavily relied on tying the schedule back to the facility design, so that activity sequencing relationships could be inferred from physical component relationships. E.g., Echeverry et al. [5] focused on scheduling installation activities and articulated rules for activity sequencing based on: (1) physical relationships among building components (e.g., supported by, weather protected, enclosed in); (2) trade interaction; (3) path interference; and (4) code regulations (e.g., safety). These rules provide one type of useful knowledge to complement the traditional CPM schedule.

Other useful knowledge stems from understanding construction methods (e.g., which combination of resources are required for each activity) and resource constraints within which a contractor has to work (e.g., use of a single crane). This knowledge can also complement the traditional CPM schedule [7, 10].

2.2. As-built Schedule

As construction progresses, changes will be made to the planned schedule to accommodate unforeseen project conditions. Examples of changes are: an activity that starts before its scheduled start (which might happen if the activity's precedence link that governed its planned early start was not a true finish-to-start link), an activity that takes longer than expected (which might happen if productivity is hampered by bad weather), or an activity that gets interrupted (which might happen as the result of equipment failure or a labor strike).

Some of these schedule changes reflect haphazard events. Others occur to handle change orders. Yet others may be corrections of mistakes in the planned schedule. The as-built schedule, which reflects the combined outcome of these changes, may thus contain useful information, but judgment must be applied to determine which of the changes constitute reusable scheduling knowledge. Comparing and contrasting planned with as-built schedules can provide meaningful insights (as needed in litigation), and this is in part how schedulers acquire their expertise. Accordingly, planned and as-built schedules could be annotated with reason specifications, to document and rationalize why changes happened. With this added knowledge, a case-based reasoner should be able to judge which reasons apply to the new project.

3. PRODUCT MODEL

If a new project to be scheduled is in some way similar to an old one with known schedule, this old schedule might be reusable. For this to be possible, the similarity between projects must be determined. A generic boiler product model is therefore being developed for CasePlan, to establish a basis for comparing projects with one another. CasePlan's boiler product model follows an industry guideline [6] that presents a framework for data sharing among computer applications for different processes (e.g., design, construction), which constitute a power plant's engineering life cycle. Using a model to represent a project's design and relating it to the construction schedule is also in keeping with the aforementioned artificial intelligence research work.

Figure 1 shows part of the generic boiler product model, which is represented as a graph in CasePlan. Each node in the graph is a class. A class pointed at by a *has-design-component* or a *has-construction-component* link from other classes, is called a *component*. A class with no arrows pointing at it, is called a *product*. Thus, "Boiler" is the only product shown. Each component or product has attributes, which can have a single value or a list of values. A value can be a number, keyword, text, expression, or pointer to another object. Figure 2 illustrates the attributes of the *Economizer* boiler component and their values.

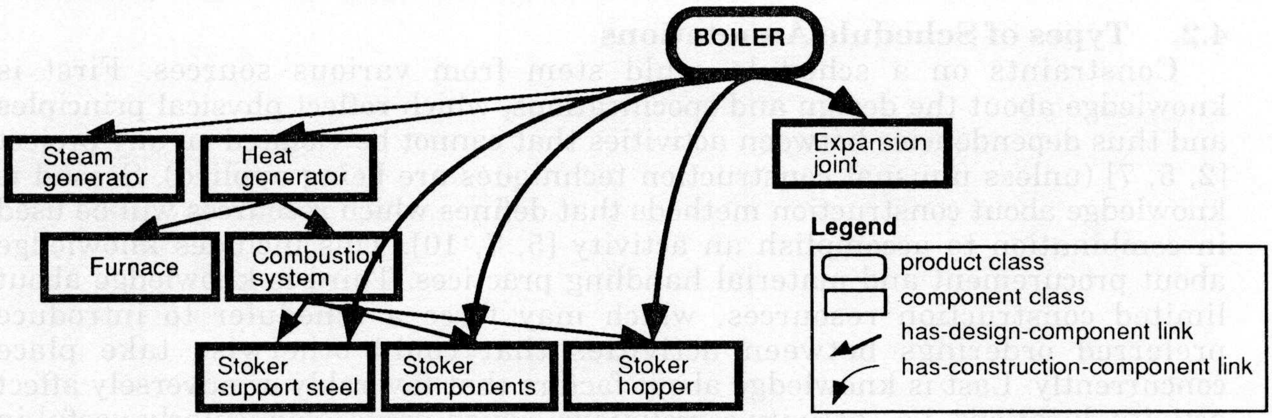

Figure 1. Product Model for Power Plant Boiler

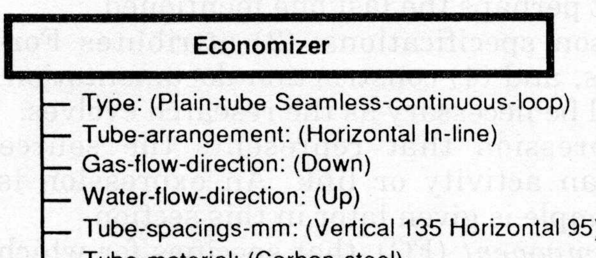

Figure 2. Attributes of Economizer (adapted from [1, 3])

4. MAKING SCHEDULES REUSABLE

4.1. Need for Schedule Annotations: Example Scenario

In a traditional scheduling scenario, a user generates a CPM schedule that has only activities with durations and precedence links. When the user changes a duration or link, the network algorithm will update all timing information (early start (ES), early finish (EF), late start (LS), late finish (LF), and floats) to reflect the change.

Suppose that the user knows that the ES of activity *Install-upper-drum* depends on the delivery date of the drums in addition to the EFs of the activity's current predecessors. This delivery date was originally not shown in the network. The user tries to change the activity's ES to the confirmed delivery date, but this date happens to be earlier than the original ES. Assuming that the drums can be stored on site from their delivery until installation, the ES and the schedule need no modification because the EFs of the activity's predecessors govern the calculation.

Later, the user is assigned to a new project that is similar to the previous one. Parts of the old schedule are therefore reused to construct the new schedule. Of the activity *Install-upper-drum*, only the name and duration are reused; the ES can presumably be recalculated. Thus, the dependency of *Install-upper-drum* on the delivery of the drums is overlooked. This illustrates that a piece of reusable information may be lost if one does not articulate constraints on a schedule. (Note that one could have introduced a procurement activity in the planned schedule to show this reusable dependency). In fact, many schedules used in practice are so terse that they are virtually useless in case-based reasoning [4].

4.2. Types of Schedule Annotations

Constraints on a schedule could stem from various sources. First is knowledge about the design and specifications, which reflect physical principles and thus dependencies between activities that cannot be violated on any project [2, 5, 7] (unless unusual construction techniques are being applied). Second is knowledge about construction methods that defines which resources will be used in combination to accomplish an activity [5, 7, 10]. This includes knowledge about procurement and material handling practices. Third is knowledge about limited construction resources, which may force a scheduler to introduce preferred orderings between activities that could otherwise take place concurrently. Last is knowledge about factors that favorably or adversely affect activity durations, i.e., activity correlations, which proves particularly useful in schedule updating [8]. Our aim in CasePlan is to annotate schedule with each of these different kinds of knowledge, except perhaps the last one mentioned.

At present, CasePlan uses: (1) reason specifications, (2) attributes For-component and Due-to, (3) link priorities, and (4) construction documentation. Refinements of and additions to these will be necessary as the research evolves.

A *reason specification* is an expression that represents the source determining a user-specified value of an activity or link. An expression is composed of functions and values. An example is given later in this section.

Each activity has the attribute *For-component* (FC), that specifies for which product component(s) the activity is performed. Most activities are directly or indirectly related to one or several product components or the product as a whole. E.g., *Set-economizer-frame* is an activity for the boiler component *Economizer*. *Mobilization* is an activity for the *Boiler* as a whole.

Each activity and each link has the attribute *Due-to* to specify the reason for their existence when they cannot be attributed to components. Activities or links may be needed because of project specifications, site conditions, or regulations. E.g., inspection and testing activities usually fall in the latter category. A reason specification can be used as the value for attribute *Due-to*. E.g., activity *Install-ID-fan* is needed for the component induced draft fan. Thus, its *For-component* can have *Component-ID-Fan* as value. For boilers with ground-supported ID-fans, an additional activity *Install-support-foundation* is needed. Thus, the activity's *For-component* can be assigned to *Component-ID-fan*, and *Due-to* to an expression like *(need-ground-support Component-ID-fan)* .

A *priority* indicates a link's necessity. A high-priority link indicates that the priority link typically indicates a resource constraint or scheduler preference, to be reconsidered upon reuse. E.g., the link between activities *Install-column* and *Install-beam* is of high priority because the column must be in place to support the beam. The link between *Install-east-wall* and *Install-north-wall*, to reflect the position of a crane or the availability of a crew, is of low priority.

Construction documentation describes how the actual construction occurred and why it deviated from the schedule.

4.3. Use of Schedule Annotations

Annotation helps with schedule reuse as follows. If all the user knows is "*Install-upper-drum* has an IES constraint with value X," it is unclear whether the constraint or the constraint value should be reused in a new schedule. X may not apply even though the reasoning behind it may. If X was based solely on the old delivery date of the drums, then the appropriate constraint value for the new project should be the new delivery date. A reason specification (shown underlined) can thus annotate the value X as follows:

```
(X (equal (delivery-date Component-upper-drum)) )
```

where *delivery-date* is a function that retrieves the value of the delivery date attribute for the upper drum component. This constraint thus represents a procurement principle.

A small activity network illustrates the use of schedule annotations. Figure 3 shows the notation used in subsequent figures. Each bold-lined box represents an activity, with its name and duration. Above each activity is the CPM timing information. Below it is the user-specified background information and annotation. A line between two activities represents a sequential link, which can be of type start-to-start (SS), start-to-finish (SF), finish-to-start (FS), and finish-to-finish (FF). The use of these four types removes the ambiguity introduced by using the FS link only. Above each link is the minimum lead time, below it the maximum lead time, each with its annotation. A solid line represents a high-priority and a dashed line represents a low-priority link. Links between critical activities are shown in bold.

Figure 4 shows a traditional schedule. Activities A1, A2, and A4 are critical. A1 has a FF link to A2 with minimum lead time equal to 1. A2 has a FS link to A4 with minimum lead time 1 and maximum lead time 5. Thus, A4's ES should be later than (6 + 1 = 7), but earlier than (6 + 5 = 11).

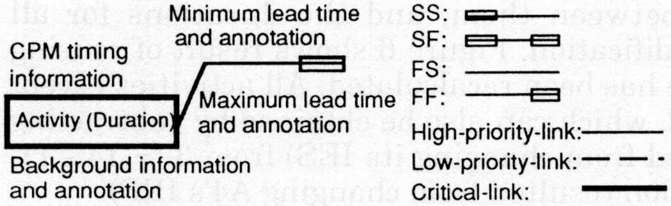

Figure 3. Annotated Schedule Notation

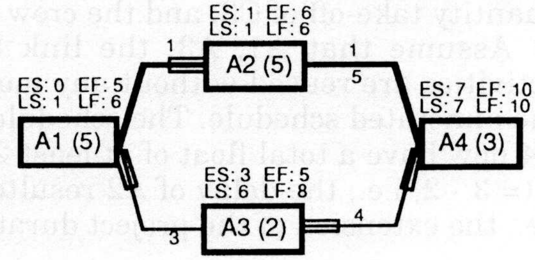

Figure 4. Traditional Schedule

Figure 5 shows the annotated schedule after it has been recalculated. For brevity, only activities A2 and A4 will be discussed. A2 has an IES of 2, annotated with reason specification R1, which delays its original ES from 1 to 2. A2 is required for component C3. A4 has an IEF of 6, annotated with R6, which does not affect its original ES of 8. A4 is also required for component C3. The minimum lead time of 1 for the link between A2 and A4 is derived from some reason specification R2, and the maximum of 5 from some R3.

The critical path changes after the schedule is recalculated. A1 is no longer critical. Thus, the user-specified constraint (i.e., A2's IES = 2) results in a schedule that does not have a continuous critical path. A part of the critical path is implied by the constraint. I.e., if A2's IES were replaced by adding a dummy predecessor with EF 2, a continuous critical path would have existed.

All links except the one between A3 and A4 are of high priority, so they are reused. The low priority A3-A4 link requires reconsideration upon reuse.

When the schedule is chosen for reuse in a project that comprises C3, A2 and A4 will need to be included in the new schedule. All reason specifications need to be reevaluated to determine proper values for the user-specified constraints in the new project. Suppose R1 is:

```
(if (Require Component-C3 Material-M3)
    (delivery-date Material-M3)     nil)
```

where *Require* is a keyword, and *delivery-date* is an attribute of the *Material* class. It says: "If component C3 requires material M3, the value will be the delivery date of material M3. Otherwise, the value is nil (i.e., IES will not be imposed)." If C3 in the new project requires the same type of material and will be delivered on day 3, IES = 3 is imposed on A2 as shown in Figure 6. Suppose R6 is:

```
(if (and (earliest-project-completion-date Specifications)
         (not (successors Activity)))
    (earliest-project-completion-date Specifications)   nil)
```

where *earliest-project-completion-date* is an attribute of *Specifications* class, and *successors* is an attribute of *Activity* class.

An IEF was imposed on A4 because the project specs. required the project to be completed no earlier than day 6. Such constraint may occur when the project is subcontracted, and the subcontractor cannot finish the last piece of work (e.g., enclosing a building) until other contractors have made certain progress (e.g., moving in large equipment). If the new project also specifies the earliest-project-completion-date and A4 has no successors, a new IEF will be used. Otherwise, no IEF will be imposed. In this example, assume a new value 14 is assigned.

The A2-A4 link may reflect some safety concern, R5: `(Safety-concern)`. This high-priority link is preserved for the new A2 and A4. R2 and R3 are also reevaluated to determine the values for the link's minimum and maximum lead time. Assume the values remain the same.

The activity durations may also be reusable if they were annotated. E.g., the reason specification can be a formula that determines the duration based on the quantity take-off of C3, and the crew and equipment used by the activities.

Assume that A1, A3, the link between them, and the durations for all activities are reused without any modification. Figure 6 shows result of reusing the annotated schedule. The schedule has been recalculated. All activities except A4 now have a total float of at least 2, which can also be obtained by subtracting 1 (= 3 - 2, i.e., the delay of A2 resulted from changing its IES) from 3 (= 14 - 11, i.e., the extension of the project duration resulted from changing A4's IEF).

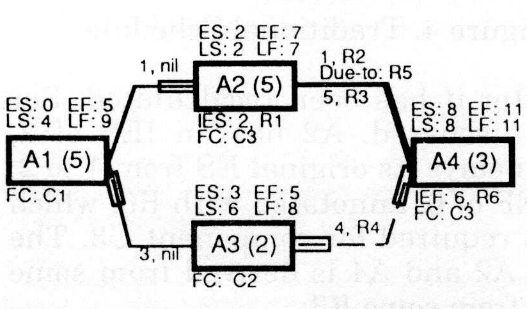

Figure 5. Annotated Schedule

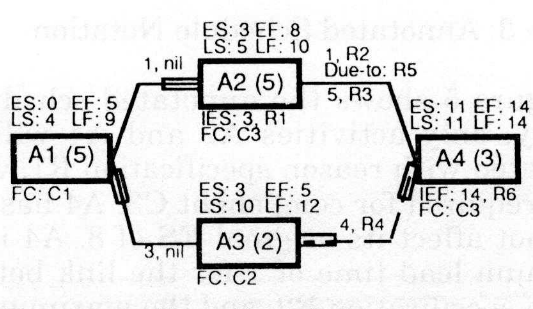

Figure 6. New Schedule

5. CASEPLAN'S AUTOMATED SCHEDULING

5.1. Ingredients of a Case

A CasePlan case comprises a product, a site, project specifications, and an annotated schedule. A *site* has attributes that describe the place where construction of the product occurs. The *project specifications* specify the constraints that are imposed by the owner, architect, or local government, etc., beyond those pertaining to the product itself.

To build a case, the user must: (1) define the product, site, and project specifications, (2) provide an annotated schedule, and (3) categorize the case. A more detailed description of (1) and (3) can be found in [11].

5.2. Providing an Annotated Schedule

When an old schedule is provided to CasePlan, it is important that no seemingly redundant links be removed from it. Traditional CPM algorithms remove dependency links that are implied by others, to facilitate computations. Removing those dependency links would hamper schedule reuse, however.

After the user has annotated activities and links, CasePlan breaks down the provided schedule into subnetworks based on the components that activities are associated with (i.e., value of *For-component*). This makes it possible to reuse part of a schedule when only some components are present in a new project design. The user may inspect each subnetwork and modify it if necessary. The subnetworks (termed *component technologies)* are stored with their associated components in the case and tied to the generic product model.

Figure 7 shows an annotated schedule given to CasePlan. Each activity is denoted by its name and associated component, e.g., activity A1 is associated with C1 and C2. The dashed line between A6 and A5 represents a redundant link. Figure 8 shows the component technologies generated by CasePlan for the individual components C1, C2, and C3. A component technology can be further abstracted into what is termed a *product technology*. Such abstraction would result in a single activity between (S) and (F) for each component in Figure 8.

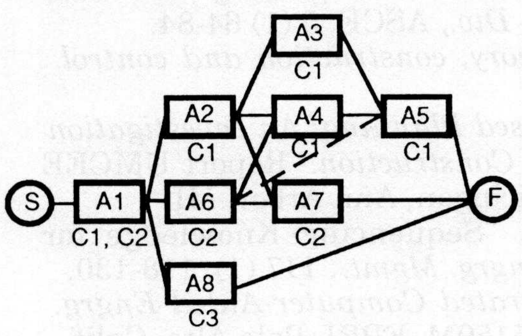

Figure 7. Annotated Schedule

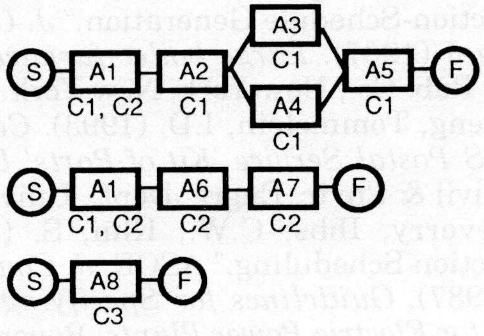

Figure 8. Component Technology

5.3. Automated Scheduling

CasePlan performs three tasks to reuse an annotated schedule and produce a new one: (1) planning: matching components of the new with the old product, determining a product technology and a component technology for each component in the new product, specializing the link between technologies, and removing redundant links; (2) scheduling: performing quantity take-off, determining the crew, calculating duration and CPM data for each activity; and (3) archiving: creating the new case and storing it. The details of this procedure are beyond the scope of this paper.

6. SUMMARY

CasePlan uses product models and annotated schedules to automate scheduling through case-based reasoning. Product models provide a consistent representation of design information from one project to the next, which makes it possible to assess the degree of similarity between projects. Design-, resource-, method-, and preference annotations help a scheduler understand the logic of a schedule and why actual construction may differ from what was planned. They support schedule reuse by maintaining proper values for user-specified constraints. With this knowledge, a scheduler will be able to produce a more realistic schedule when reusing old ones. Presumably, the resulting schedule will be more accurate for predicting and controlling project execution and resource use as parts of it have been tried, but this will be the case only if those parts can be pieced together well. The annotation will also be useful to inexperienced schedulers who wish to study real constraints imposed on construction schedules and to companies to capture and formalize their standard practice.

7. ACKNOWLEDGMENTS

This research is funded by grant MSS-9215935 from the National Science Foundation (NSF), whose support we gratefully acknowledge. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of NSF.

8. REFERENCES

1. CEAGI (1982). *Standard design of thermal power station with 200/210 MW units having CE boiler and KWU turbine*. Central Electr. Authority, Govt. of India, Central Board of Irrigation and Power, New Delhi, India, pp. 29-30.
2. J.M. Cherneff, Logcher, R., and Sriram, D. (1991). “Integrating CAD with Construction-Schedule Generation,” *J. Constr. Div.*, ASCE, 5 (1) 64-84.
3. R. Dolezal (1967). *Large boiler furnaces: theory, construction and control.* Elsevier Pub. Co., New York, New York.
4. R.-J. Dzeng, Tommelein, I.D. (1993). *Case-based Planning: An Investigation of the US Postal Service 'Kit of Parts' Use in Construction.* Report UMCEE 93-18, Civil & Envir. Engrg. Dept., Univ. of Michigan, Ann Arbor, MI.
5. D. Echeverry, Ibbs, C.W., Kim, S. (1991). "Sequencing Knowledge for Construction Scheduling." ASCE, *J. Constr. Engrg. Mgmt.*, 117 (1), 118-130.
6. EPRI (1987). *Guidelines for Specifying Integrated Computer-Aided Engrg. Applics. for Electric Power Plants.* Report NP-5159M, EPRI, Palo Alto, Calif.
7. C.Hendrickson, Zozaya-Gorostiza, C., Rehak, D., Baracco-Miller, E., Lim, P. (1987). "Expert System for Construction Planning." ASCE *J. Comp. in Civil Engrg.*, 1(4), 253-269.
8. R.E. Levitt, Kunz, J.C. (1985). "Using Knowledge of Construction and Project Management for Automated Schedule Updating", *Proj. Mgmt. J.*, 14 (5) 57-76.
9. C.K. Riesbeck, Schank, R.C. (1989). *Inside Case-Based Reasoning*. Lawrence Erlbaum Associates, Publishers, Hillsdale, New Jersey.
10. I.D. Tommelein, Carr, R.I., Odeh, A.M. (1994). "Assembly of simulation networks using designs, plans, and methods." ASCE, *J. Constr. Engrg. Mgmt.*, in press.
11. I.D. Tommelein, Dzeng, R.-J. (1993). "Product Modeling to Structure a Case Library for Case-Based Construction Planning." CIB W78, *The Mgmt. of Information Technology in Constr., Proc. 1st Intl Conf.* Si