

4 -WBS, Predecessors & Scheduling

In this chapter we're going to use the work packages and estimates we developed earlier as the building blocks for our detailed project plan. As you will recall, we crafted these building blocks from our requirements. We "bundled" achievements and specifications to craft assignments for our project team members. Then we met with the team members (at least the known members of the team) and secured their agreement that the assignment was feasible. Then we negotiated an estimated amount of work, not just a duration or a completion date, with each individual. So now we have our work packages in measured achievement form and a reasonable level of commitment to a work estimate.

Next we'll take these work packages and build a simulation of the project in the software. By simulation we mean using the software's resource leveling algorithms and calculations to give us a model of the project plan that updates and optimizes itself whenever we make changes. We need to become comfortable with modeling the project as opposed to developing a static plan. When we use these techniques we make it relatively easy to model alternative ways of doing the project. We also give ourselves a tool to quickly assess the changes that executives or customers want to make in the project. This modeling approach is an ideal tool for considering trade-offs between the level of achievements, risk, budget and the project duration.

BUILDING OUR PROJECT MODEL

With the information we have in our work packages, we'll go through the following three-step process to build a model of the project.

- ❑ *Enter our achievement network and the supporting work packages as our Work Breakdown Structure (WBS)*
- ❑ *Create predecessor relationships between the work packages.*
- ❑ *Enter our resource availability and cost data (labor and materials)*

This will give us a model of the project plan that automatically adjusts itself whenever we make a change. It's a very different process than the one to which you may be accustomed. For most assignments, we're not going to enter start and finish dates; we're going to let the software calculate them from a combination of our predecessor relationships, the amount of work and the resources' availability. As an example, a task will have a start date of May 1st because its predecessors are finished by that date. It will have a duration of 10 working days because the task requires 40 hours worth of work and our resource is available half-time (the person can do 20 hours worth of work a week). If we make changes to the staffing or work estimates or the predecessors of this task, that may allow it to start earlier and the software will automatically make that change. Similarly, if we lose some of this person's availability so they are available only quarter-time (can do ten hours of work each week) the duration of our example task will increase from 10 working days to 20 working days. The software will make this change automatically as well as altering the start dates of the successors of our example task so we can see the full impact of the lost productivity.

There's no question that just entering start and finish dates is simpler than dealing with work estimates, capacity and predecessors. But over the course of the project, entering start and finish dates will take much more time. In fact, on larger projects, entering start and finish dates is the equivalent of engraving our project plan on a stone tablet. It usually takes so much time to adjust the plan that we don't do it.

By building our plan as a simulation, we use the software's resource leveling algorithms. Resource leveling ensures that no resource is assigned more work than they can do and schedules the project to finish as early as possible, given the data we have entered. So if we have made an arrangement with a person's boss that makes them available to the project half time, the software will adjust all of their assignments so that we never schedule them to work more than 20 hours each week. This does not mean we can't ask people to work overtime nor does it mean that no one on the team can work more than 40 hours per week. But we consciously set the limits on each resource's availability and once we have set those limits, the software enforces them for us. Resource leveling takes some getting used to because it is different than other PC applications with which you may work. In a word processing program the software will not go back and change the first paragraph automatically while you are typing the third paragraph. But in Microsoft Project®, resource leveling may automatically change the start and stop dates for task 17 when you enter the resources, work estimates or predecessors for task 73.

There are people in the project management world who suggest that PMs should never use resource leveling because it will increase the duration of a project. Instead they say we should just add resources. While resource leveling can increase the project duration, using it reflects the real world of project management. We don't always have the option of going out and getting more resources. As a refresher, let's take a look at our work packages for the project before building our work breakdown structure (WBS).

MS Project exhibit

ID	WBS	Task Name
1	1	\$17 million Revenue
2	1.1	7.4m Space Capacity Plant
3	1.1.1	Site purchased meet JITD 30x30
4	1.1.1.1	REC contracted
5	1.1.1.2	Site & alternates meet JITD spec
6	1.1.1.3	Site location apprvd by VPO
7	1.1.1.4	Site purchase executed by CEO
8	1.1.2	Plant apprvd for construction
9	1.1.2.1	Architect contracted
10	1.1.2.2	Building design apprvd by VPM
11	1.1.2.3	GC contracted per design spec
12	1.1.2.4	Permits Obtained
13	1.1.3	Plant structure meets design spec
14	1.1.3.1	Site prepared for Foundation
15	1.1.3.2	Foundations set
16	1.1.3.3	Underground utilities installed
17	1.1.3.4	Floor set
18	1.1.3.5	State Inspect. & A/E approve Foundation
19	1.1.3.6	Steel frame & roof 100%
20	1.1.3.7	Exterior shell & insulation 100%
21	1.1.3.8	Main Electric Xfmr & Service 100%
22	1.1.3.9	Rooftop HVAC 100%
23	1.1.3.10	State Inspect. & A/E approve Shell
24	1.1.3.11	Rooftop HVAC supply heat
25	1.1.3.12	Electric distribution prod.area A
26	1.1.3.13	Electric distribution system 100%
27	1.1.3.14	HVAC air distribution prod.area A
28	1.1.3.15	HVAC air distribution 100%
29	1.1.3.16	Interior office 100%
30	1.1.3.17	Fire & life safety systems 100%
31	1.1.3.18	Cert-of-Occupancy issued by State Inspector
32	1.1.3.19	Architect signs-off on As-built doc.s
33	1.1.4	Prod. line 1-2-3 environment meet spec.s
34	1.1.4.1	Air quality exceeds spec.s
35	1.1.4.2	Power quality exceeds spec.s
36	1.1.4.3	Compressed air provides specified volume
37	1.2	284k Production Program
38	1.2.1	Management staff hired
39	1.2.2	Production plan approved by VPO
40	1.2.3	Prod. Equip. accepted by plant engr.
41	1.2.4	Raw material inventory filled
42	1.2.5	Pilot product passes QA test
43	1.2.6	Sales orders gr.than 32.4k units
44	1.2.7	5.4k prod.units in warehouse
45	1.3	3.4m/284k JIT Delivery
46	1.3.1	Distribution plan apprvd by VPO

WORK BREAKDOWN STRUCTURE (WBS)

The software appendix contains step-by-step instructions for creating the work breakdown structure and all of the other keystrokes necessary to build and track your plan. We've separated the use of the information from the data entry instructions so you can get the big picture first and reference the details later.

ID	Task Name	Fixed Cost	Notes	Ma
1	\$17 million Revenue	\$0		
2	7.4m Space Capacity Plant	\$0		
3	Site purchased meet JITD 30x30	\$0		
8	Plant apprvd for construction	\$0		
13	Plant structure meets design spec	\$0		
33	Prod. line 1-2-3 environment meet spec.s	\$0		
37	284k Production Program	\$0		
48	3.4m/284k JIT Delivery	\$0		
54	Project Management	\$0		

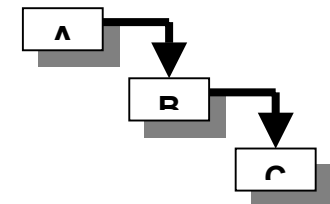
In overview, the WBS is a hierarchical listing of the high-level achievements (HLA) that lead to our project measure of success. All of our HLAs are summary tasks which means they have other tasks indented under them. Summary tasks serve a special purpose; they are "holding tanks" for the work, duration and cost of the tasks indented under them. We will use our summary tasks to "rollup" the detail underneath them and give ourselves a way of looking at the project plan in summarized form. We won't enter task information such as duration, work, start or finish dates, predecessor or resource assignments for a summary

task. We enter them only for tasks. The system aggregates and summarizes information from each task into information in the summary task.

DESIGNING PREDECESSOR NETWORKS

Our next step is to take the Work Breakdown Structure (WBS) and detail the interactivity of the tasks. This process will link the dependencies of each task with another, forming the project beginning and end in a logical fashion of what comes first, what can be done together and what comes last. We will define the predecessor (leader) and successor (follower) relationships between tasks inherent in the nature of the work.

It's important to remember that we specify only the immediate predecessor of a task. If task C can not start until task B finishes and B can't start until task A finishes, then C's predecessor is B, and B's predecessor is A. There is no reason to tell the system that both A and B must finish before C starts.



PREDECESSOR AND SUCCESSOR RELATIONSHIPS

When you link tasks in a project, you define a dependency between their start and finish dates. There are four kinds of task dependencies:

<u>Task dependency</u>	<u>Description</u>
Finish to start (FS)	Task (B) cannot start until task (A) finishes.
Start to start (SS)	Task (B) cannot start until task (A) starts.
Finish to finish (FF)	Task (B) cannot finish until task (A) finishes.
Start to finish (SF)	Task (B) cannot finish until task (A) starts.

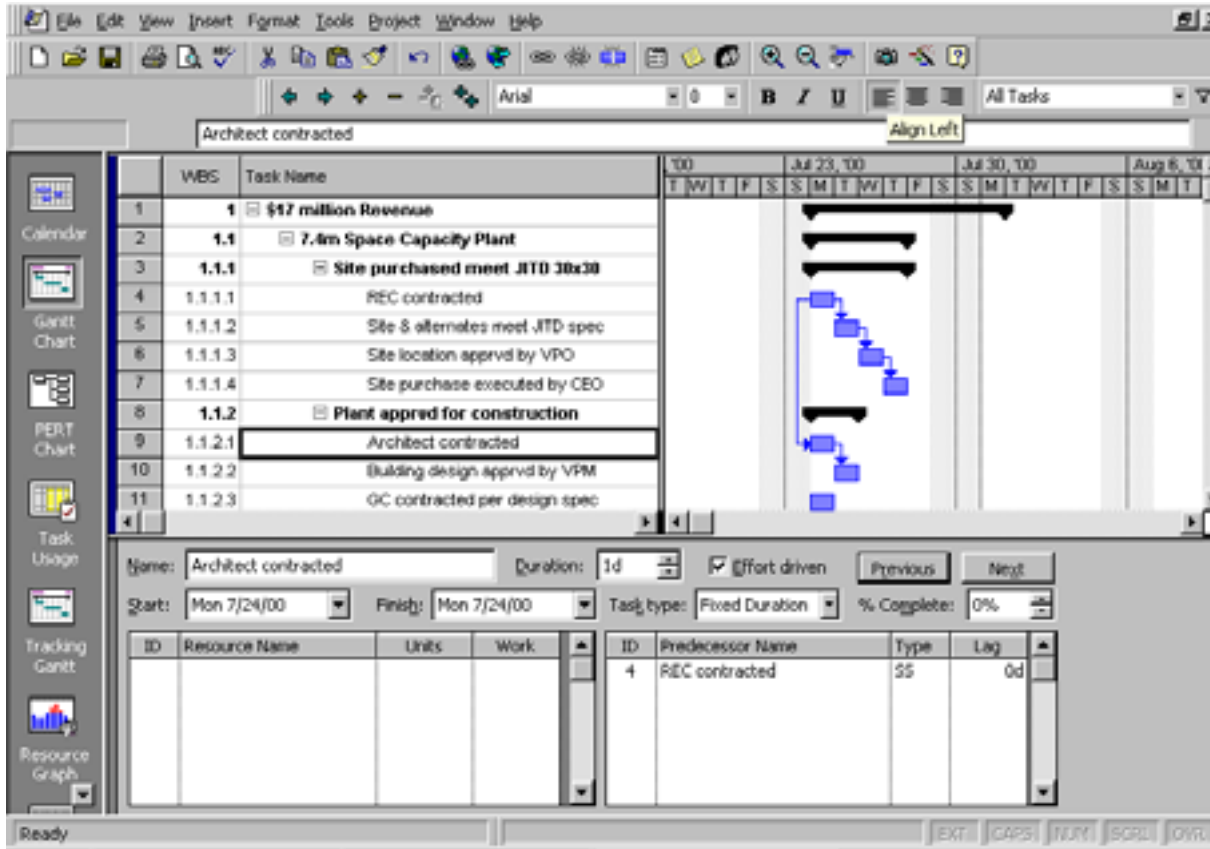
The diagrams show four types of task dependencies between task A and task B:

- (FS)**: Task A is above task B. An arrow points from the right side of task A to the left side of task B.
- (SS)**: Task A is above task B. An arrow points from the left side of task A to the left side of task B.
- (FF)**: Task A is above task B. An arrow points from the right side of task A to the right side of task B.
- (SF)**: Task A is above task B. An arrow points from the left side of task A to the right side of task B.

The advantage of linking tasks is that when your task schedule changes during planning or later when tracking work progress, the rest of the project will re-schedule itself automatically. We gain this advantage by:

- ❑ *Not entering task start and stop dates without links. Or ...*
- ❑ *Not setting constraint dates in tasks. Constraints are created by fixing task start or finish dates.*

The preference is to define all tasks to start “as soon as possible”. Usually the only constraint set is the project start or finish date. To add a time-phasing component to linked tasks we utilize lag and lead times functions to the task dependency interaction. Lag-time provides a delay between tasks that have a dependency. For example, if you need a 2-day delay between the finish of one task and the start of another, you can establish a finish-to-start dependency and specify a 2-day lag-time. You enter lag-time as a positive value. Lead-time provides an overlap between tasks that have a dependency. For example, if a task can start when its predecessor is half finished, you can specify a finish-to-start dependency with a lead-time of 50 percent of the successor task. You enter lead-time as a negative lag value.



Let's examine our WBS and specify the predecessor relationships, again keeping in mind that we need specify only the immediate predecessor(s) for a task, not all the predecessors. An example of this is to the left. The Gantt chart in the upper half of the screen graphically shows task #7 has a finish to start (FS) predecessor of task #6, task #6 has a finish to start (FS) predecessor of task #5 and task #5 has a finish to start (FS) predecessor of task #4. In the lower half of the screen the figure also shows the detail of task #9 (the one we have clicked on). We see that task #9 has a start-to-start (SS) predecessor of task #4.

Let's begin the process of assigning predecessors. In practice we assign predecessors after we have entered the WBS into MS Project®. At the end of this chapter (and in the Appendix) there are step-by-step

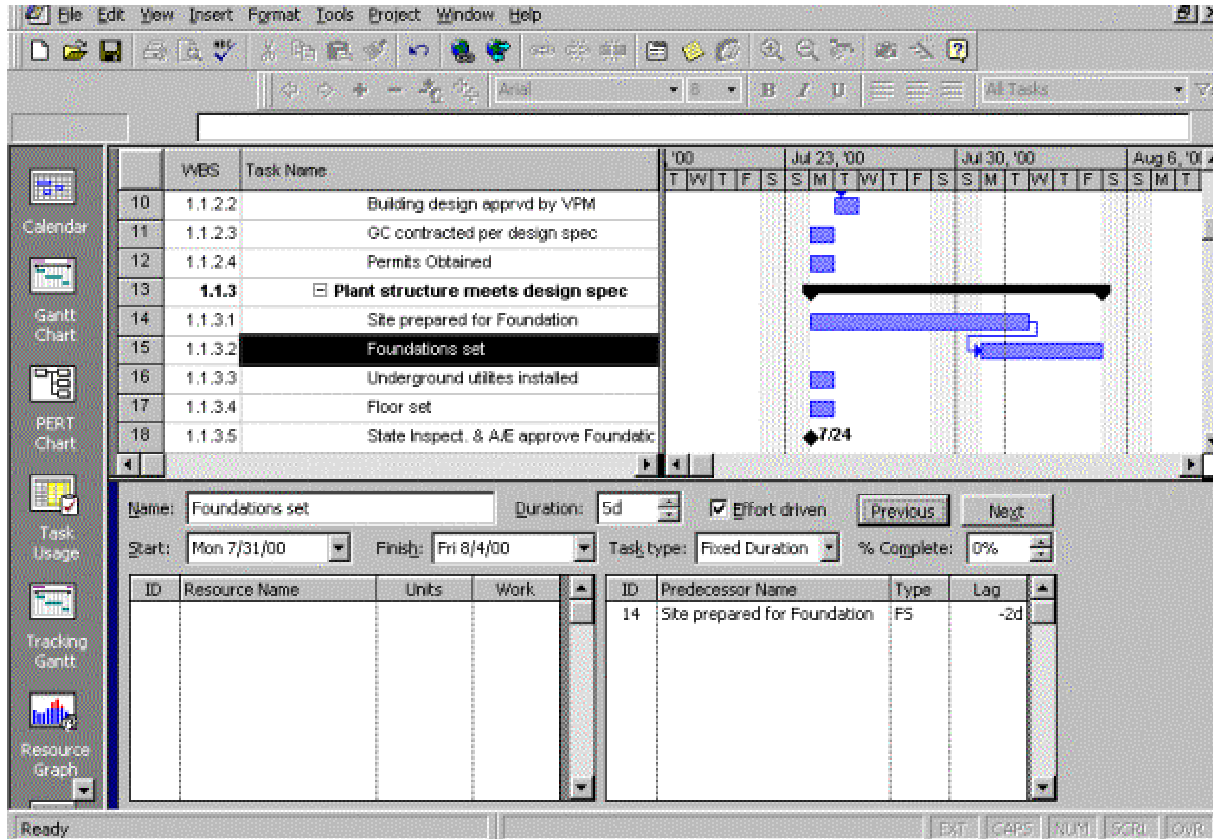
instructions for entering data into MS Project. Remember, we don't ever specify predecessors for any of our summary tasks like those in rows #1, 2, 3 and 8.

There are many ways to enter dependency relationships in the software. The method above uses The Task Entry Screen. Starting at the Gantt chart screen we get this display by clicking VIEW, then MORE VIEWS, and then select "Task Entry" from the drop down list.

Task #4 has no predecessors because it can start on the first day of the project. So we leave the predecessor field blank. Task #5 has a predecessor of Task #4. So we click on Task #5 in the upper half of the screen. Then in the lower half of the screen we click

on the column “ID” in the lower right Predecessor box and enter “4”. Or we can click in the “Predecessor” column and select Task #4’s name from the drop down menu. Then click on the “OK” button. The system assumes a finish to start relationship unless we select a different predecessor type from the “Type” column. We can directly enter FS, SS, FF or SF, or use the drop down menu to pick our choice.

Now let’s skip down the WBS and look at some of the other predecessors we need to create.



Task #15 (Foundations set) doesn't have to wait for task#14 to be entirely done before the Foundation task can start. So we will set this predecessor up as a finish to start (FS) with a lead-time. We enter -2d in the lag column to let the start of task #15 lead the finish of task #14 by two days. In other words the foundation crew can start work slightly before the site crew has finished excavating. Leads like this one can reduce the project's duration but there is always the risk that some of this early work will be wasted if there are last minute changes to the predecessor. To create a lag, we would put a positive value in the lag column and that would delay the start of task #15 rather than have it start directly after task #14 is finished.