LECTURE 6

Scheduling a Plan

WBS identifies the work that it is necessary to carry out, but it does not show any constraints between activities, nor does it specify anything about scheduling, that is, when each activity should start and how long it should last. This is exactly what we are going to do in this section.

Scheduling the plan is composed of the following steps:

1. Identify dependencies among activities. During this step, we highlight the dependencies in our project to understand the degrees of freedom we have in scheduling our project. Some activities will have no dependencies and we will be able to schedule them more freely. Others will depend on tasks to finish (or to start) before they can be started; for these, we will clearly have less options.
2. Identify the critical path of the plan. The goal of this activity is to identify the most critical activities in the plan. These are the activities that, if delayed, will delay the plan.
3. Allocate resources to tasks and level resources. The goal of this activity is to allocate actual resources to the different activities. During this step, various additional constraints emerge, due to the availability of resources and the maximum amount of work that can be allocated to each resource. The process of dealing with such constraints is called resource leveling. The output is a plan that introduces additional constraints, called soft constraints, which ensure that the limitations related to resource availability are actually met. Note that the order in which we listed the activities above is merely for presentation purposes. In practice, there is a lot of freedom in the way in which these steps are executed. In many cases, schedules are constructed by looking at the different concerns in parallel, trying different scenarios. In the current practice, the use of a modern Gantt charting tool integrates the steps above, promoting a process in which the schedule is built by looking at all these concerns in parallel.

Identify Dependencies among Activities

No one will start building a house from the roof. Thus, the first step to scheduling our plan is to identify the order in which the activities can be executed. This is done by identifying the dependencies among activities. In general, a dependency between two activities A and B defines some kind of constraint in the executability of A and B and imposes a partial ordering on the execution of the activities. The dependencies can be

characterized according to different dimensions, as illustrated in the following paragraphs.

Type of Dependencies

Four types of dependencies can be set between two activities, according to whether the constraints involve the start or the end of the two activities. Two types of constraints are relatively common. These are:

1. Finish to start (FS). An FS constraint between A and B expresses the fact that B can start only after A is finished. This is probably the most common constraint between activities. For instance, the activity "baking a cake" can start only when all the ingredients have been poured into the baking pot.

2. Start to start (SS). An SS constraint between A and B expresses the fact that B can start only when A starts. For instance, a "monitoring" activity can start only when the activity that is being monitored starts.

Two types of constraints are used and found less often. These are

3. Finish to finish (FF). An FF constraint between A and B describes a situation in which B can finish only when A finishes.

4. Start to finish (SF). An SF constraint between A and B describes a situation in which B can finish only when A starts.

SOFT VS HARD CONSTRAINTS

Another classification distinguishes between hard and soft constraints.

A **hard constraint** between two activities A and B models a dependency that is in the nature of the work to be performed. A hard constraint cannot be broken without violating the logic of the project. An exception is fast tracking, which optimizes a plan by breaking hard constraints, at the cost of a riskier project execution

A **soft constraint** between two activities A and B can be set as a convenience to simplify project execution or to reduce risks in the project execution. A soft constraint can be broken without violating the logic of the project.

An example can be of help. In the preparation of a meal, a hard constraint exists between the preparation of the ingredients and cooking them. Cooking, in fact, cannot start if the ingredients have not been prepared. The constraint is in the logic of the activities and there is no way to break it, unless you decide to be extremely creative in your cooking. In the same scenario, we could decide to impose a soft constraint between preparing the

dessert and preparing the main course, that is, arbitrarily decide that we will start preparing the dessert and then move on to prepare the main course. No dependency between the two activities exists. In principle, we could even prepare both dishes in parallel, if we wanted to. Soft constraints can be broken, if required, by changing the hypotheses for which the links were introduced in the first place.

Lead and Lag Time

When defining dependencies between two activities, sometimes it is convenient to specify a time interval, positive or negative, that occurs between the activities. We speak of lag time if the time interval is positive. We use the term lead time if the interval is negative. Thus, for instance, if A and B are connected by an FS dependency with a lag time of 3 days, it means that B can start three days after A has finished. In the example above, if the FS dependency had a lead time of 3 days, B could start 3 days before A ends.

lag → +
lead → −

A typical usage of lag time is with SS constraints, when some progress in the first activity is necessary to start the second one. For instance, in a roadwork project, an SS constraint with a lead time of one or two weeks could be set between digging and laying pipes. The lag time, in fact, is to allow the digging to progress sufficiently to actually make the laying pipes activity doable.

Similar to the identification of constraints, it is good practice to introduce lead and lag times between activities only if strictly imposed by the logic of the plan. Introducing arbitrary constraints reduces the degrees of freedom and it could make scheduling a lot more complex than needed.