Execution and Control

Execution

In traditional projects, managers are in charge of acquiring resources for a project, including hiring people and preparing them for the project. Managers also set up the work processes, quality assurance processes, data collection and reporting practices, risk monitoring and response processes, and so forth. Usually this is done by mandating certain standard processes and practices and putting people in charge of overseeing them.

In a Scrum project, new teams may be formed at the start of the project, or agile teams already formed and experienced from other projects may be brought into a new project. New teams may need to be trained in Scrum, and arranging for this falls to a manager. In an ongoing Scrum project, the overall processes and practices are already well established, although there are many details that vary among organizations. Deciding in detail how an agile method like Scrum will work is mainly left to the teams, none of whom are managers. Thus most management responsibilities for development process execution are devolved to the various players in the Scrum process.

Control

Control in Traditional Projects

In a traditional project, managers are responsible for tracking the progress of work and, if progress deviates from the plan, either changing the plan or changing what people are doing. The main way to do this is to monitor scheduled task completion: if a task is done early, then the project is ahead of schedule, and if done late, it is behind. This job is made more complex by the fact that many tasks are completed or underway simultaneously, some behind schedule and some ahead of schedule, so it is hard to get a good picture of the overall status of the project. A technique for supplying this overall status evaluation is *earned value analysis*, which uses actual and planned task completion data to compute numbers that give measures of project status. We will learn about earned value analysis later.

As noted earlier, the project management iron triangle (reproduced in Figure 1 below) imposes limits on how project plans and work can be adjusted when things have gone awry.

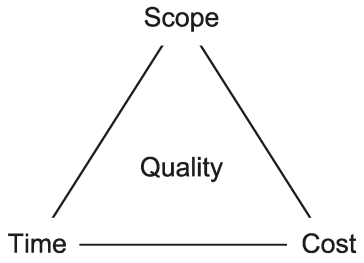

Figure 1: Project Management Iron Triangle

If a project is behind schedule, then scope may be adjusted by reducing the features and functions delivered. The delivery date and cost can then be maintained. Alternatively, the scope may be kept constant. If the delivery date is also maintained, then more people might be brought into the project, increasing its cost. If the delivery date is delayed, then because many costs depend on how long a project lasts, then costs will increase as well. If a project must reduce its costs, then scope can be reduced and either the delivery date moved closer or people can be let go, or both. If cost is reduced but scope is maintained, the people may have to be eliminated, which will delay delivery.

But there are limits beyond the iron triangle to the ways that adjustments can be made to heal an ailing project. A famous maxim called **Brooks' Law** states that adding programmers to a late project makes it later. The justification for this claim is that bringing new people into a project requires that the people already on the project spend time educating the new hires about the project and helping them become productive; this is especially true in projects that are nearly done because they have lots for the new people to learn, including requirements, designs, code, tests, and so forth. The time that project personnel spend helping the new hires is time that they cannot spend finishing the project, so the project ends up being even later than it otherwise would have been.

Another constraining factor is that there seems to be a limit on the time to develop any product, regardless of how many resources are provided to do it, as noted in the discussion of effort estimation. After a certain point, it does not matter how many people are set to work on a project: it won't get done any faster. This suggests that there is an optimal team size for every project (though we don't know what it is). Therefore, adding more people to speed up a project may just increase costs without making it finish sooner.

A final point about making adjustments to project plans or processes is that managers should do so in consultation with stakeholders. Most stakeholders become very upset when told without warning at the last minute that (a) the product they have been waiting so long for will not be available for a while, or (b) the product will not do what they expected it to do, or (c) the product will cost a lot more than they expected it to. When it becomes clear that a project is in trouble, mangers should present the situation to stakeholders, outline possible courses of action, and try to reach consensus about the best course. People who are party to a decision are much more likely to accept it (though they may not be happy about it), than are those who have it foisted upon them.

The difficulty of meeting the schedule, scope, and cost constraints of any project help explain why so many traditional projects have disappointing results or fail outright.

Control in Scrum

Progress in Scrum projects is also monitored by earned value analysis in the form of a burn chart. A **burn chart** is a graphic whose vertical axis is work (in person-effort units or story points) and whose horizontal axis is time. A **burn down chart** shows work remaining while a **burn up chart** shows work completed. Usually a burn down chart is made in every sprint so that progress during the sprint can be tracked. Adjustments can be made during the sprint if necessary, but process changes are perhaps best made in the sprint retrospective.

A burn up chart is usually made to track progress for the current release. The burn up chart can indicate that adjustments should to be made in the PBIs planned for the release, or in the release date. As noted before, changes to these two variables can go in either direction depending on whether the release is ahead of or behind schedule. These are changes in scope and time; the affect on cost is simple: a longer release is more expensive by the number of extra sprints times the cost of a sprint. A shorter release is cheaper by the number of fewer sprints times the cost of a sprint. We will discuss burn charts in more detail later.

Another tool used to control progress in Scrum is the **task board**. A task board allows team members to document all the tasks that need to be done to finish a sprint in one place, and then track each task as it is worked on and eventually completed during the sprint. A glance at the task board reveals how much progress has been made at any point, the tasks that are complete, those that are still not finished, and often how far along unfinished tasks are. Task boards are easy to make and maintain, and the provide more detail than burn charts. We will discuss task boards later on as well.

Earned Value Management (EVM)

Suppose that you have estimated that a job will require 40 hours of effort. After working for a while, you realize that you have put in 20 hours of effort. If you conclude from this that you are half done, then you may be wildly mistaken. For example, you may have over-estimated the effort and in fact be 80% finished after 20 hours of effort; or you may have under-estimated the effort and be only 20% finished after 20 hours of effort. Similarly, suppose that you have spent 50% of the budget for a particular project. This does not mean that you are 50% done. You may have over-estimated or under-estimated, or your spending rate may not be linear.

These problems, which are a traditional difficulty in project management, are addressed by a technique called **earned value management** (**EVM**). In EVM, *progress* is gauged by how much of the overall job is actually complete at a point in time. The *health* of the project is gauged by comparing how much you estimated you would have gotten done with how much you have actually gotten done by some point in time.

EVM measures **value** as either effort (such as person-days) or money (like dollars). Suppose that you have decomposed the work of a project into tasks and estimated the cost of each task (in effort or money). Each estimated task cost is the **planned value** (**PV**) of that task. You can also estimate duration for each task, figure out task dependencies, and use a Gantt chart of CPM to make a schedule. Then the total time for the project, called the **planned duration** (**PD**) of the project, is known as well. From the schedule it is not very difficult to figure out the accumulated PV of the project over time. Figure 2 below shows such a graph. The point on the Value axis where the PV line ends (at the PD of the project) is the **budget at completion** (**BAC**); this is, of course, the total estimated cost of the project.

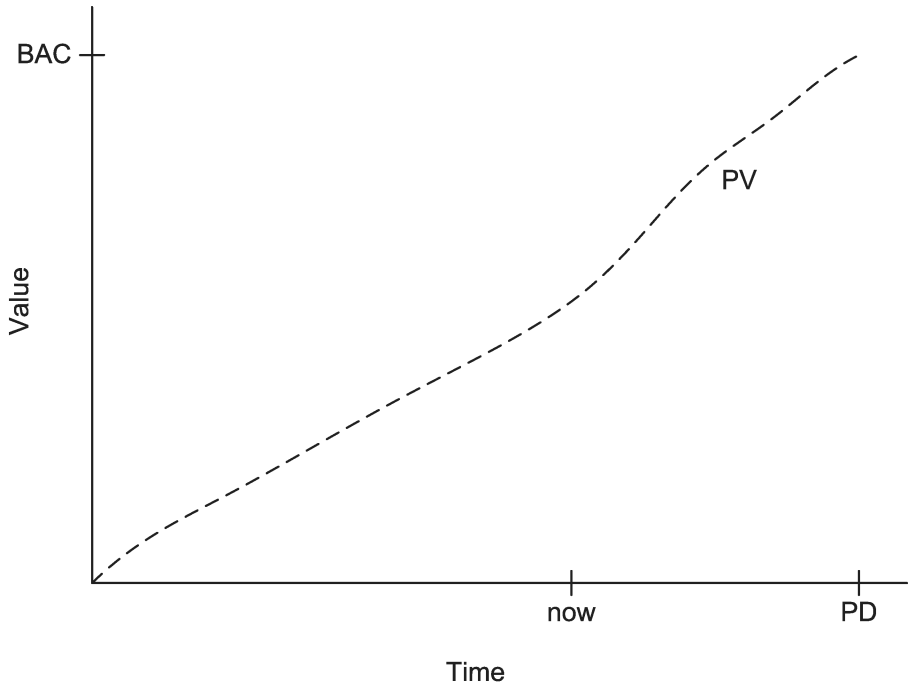

Figure 2: EVM Chart Showing PV

As a project progresses, you can collect two kinds of data: the planned value of completed tasks, called the **earned value** (**EV**), and the effort or money actually expended on completed tasks, called the **actual cost** (**AC**). Note that these will often be different. For example, if some completed task

was estimated to require five person days (its PV) but actually needed seven person days to finish (its AC), then the EV of this task is five, but its AC is seven. Both cumulative EV and cumulative AC can be plotted on a graph along with cumulative PV up to the present time. Figure 3 shows an example of such a graph.

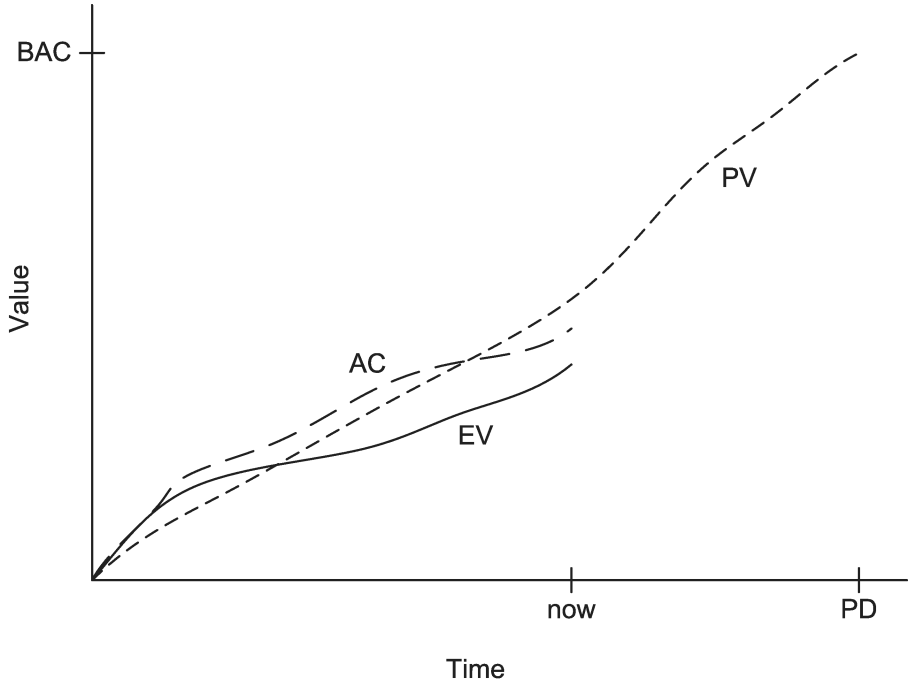

Figure 3: EVM Chart Showing PV, EV, and AC

The EV line shows that after starting out above the PV line, EV is less than PV at the present time. This means that, at the present time, it was planned that a certain value would have accrued (the PV) based on the value of the tasks that should have been done by now, but in fact the estimated value of tasks really completed at this time (the EV) is less than anticipated. So the project is behind schedule. Whenever the EV line is below the PV line, the project is behind, and whenever the EV line is above the PV line, the project is ahead of schedule.

Similarly, the AC line is above the EV line at the present time. This means that the effort or money actually expended up to now (the AC) is more than the effort or money planned to be expended for the work completed at this time (the EV). In other words, the project is currently over budget. Whenever the AC line is above the EV line, the project is over budget, and whenever the AC line is below the EV line, the project is under budget.

These conditions can be quantified using two indices.

Schedule Performance Index (**SPI**), computed as EV/PV. If this index is one, then earned value is exactly as planned and the project is right on schedule. If it is less than one, then the project is behind schedule, and if it is greater than one, the project is ahead of schedule. The degree a project is ahead of or behind schedule is indicated by how far this index is from one. For example, suppose that in the project depicted in Figure 3, the current EV is 380 while the PV at this point in time was expected to be 400. Then the SPI is 360/400 = 0.9, indicating that the project is behind schedule (as also indicated by the graph).

Cost Performance Index (**CPI**), computed as EV/AC. If this index is one, then the amount spent on the project so far is exactly as much as it should be, so the project is right on budget. If this index is greater than one, then the project is under budget, and if it is less than one, then the project is over budget. Suppose that for the project in Figure 3, the AC is currently 378.94 and the EV is 360. Then CPI is 360/378.94 = 0.95, indicating that the project is over budget (as also indicated by the graph).

Besides providing numerical indice for the health of a project, these indices can be used to make predictions. The **Forecast Project Duration** (**FPD**) is computed as PD/SPI. This is an estimate of project duration based on how far behind or ahead of schedule the project is right now. Continuing the example from Figure 3, suppose that the PD is 30 months. Then the FPD is 30/0.9 = 33.33 months, indicating that if progress continues at the current rate, the project will finish three and one third months late.

The **Estimate At Completion** (**EAC**) is computed as BAC/CPI. This is an estimate of the total project cost based on how much has been spent for the amount of work done so far. Again using the example from Figure 3, suppose the BAC is 1140. Then the EAC is 1140/0.95 = 1200, so at the current rate of expenditure, the project will cost 60 units more than anticipated.

Managers can use EVM charts like the one in Figure 2 above to track whether a project is on schedule and on budget, and use FPD and EAC to make new estimates of project duration and cost. These indicators can help decide what actions to take to either bring a project back on schedule and budget (by changing scope, for example), or to revise estimates of project time and cost.

Burn Charts

Scrum projects traditionally use a simplified kind of earned value management in the form of graphs called burn charts. A **burn chart** shows value on the vertical axis against time on the horizontal axis. The value on the vertical axis can be the number of PBIs, story points, ideal days, or ideal hours, or it can be monetary units (which could be useful if business value was estimated in monetary units when determining PBI priorities). The time on the horizontal axis can be either days or sprints. To measure earned value, work is only plotted on a burn chart when a PBI is complete, and it is plotted as the PBI size estimate rather than the actual effort expended during the sprint (which would be actual cost).

At the start of a project, release, or sprint, the planned value over time is plotted (using the same units that will be used for the earned value). For example, if the number of story points is going to be used to measure the value, this curve illustrates the planned number of story points that will be completed during the project, release, or sprint. Thus the relationship between the earned and planned value lines indicates the health of the project, release, or sprint. We illustrate this below.

Two kinds of burn charts are used in Scrum. A **burn-up chart** shows how much value has been *completed* on the vertical axis against time on the horizontal axis. A **burn-down chart** shows the value *remaining* to be completed on the vertical axis against time on the horizontal axis. Burn-down charts tend to be used more often to chart progress in projects or releases, and burn-down charts more often to track progress in sprints.

To illustrate, consider the burn-up chart in Figure 4 below. This chart shows story points in a release against sprints. The chart shows the state of the project after the seventh sprint. At this time 13 sprints are planned to complete the release. The dashed-and-dotted horizontal line at the top shows that at first only about 350 story points were planned for the release. After the first sprint, many feature estimates were increased so that there were about 425 story points in the release. After the third sprint, some changes in the features for the release increased the story points to just under 450.

The dashed line shows the planned value for the project, that is, how many story points are expected to be completed at each sprint. Since story points are a measure of effort, and the team completes work at a more or less constant velocity, this line has a constant slope.

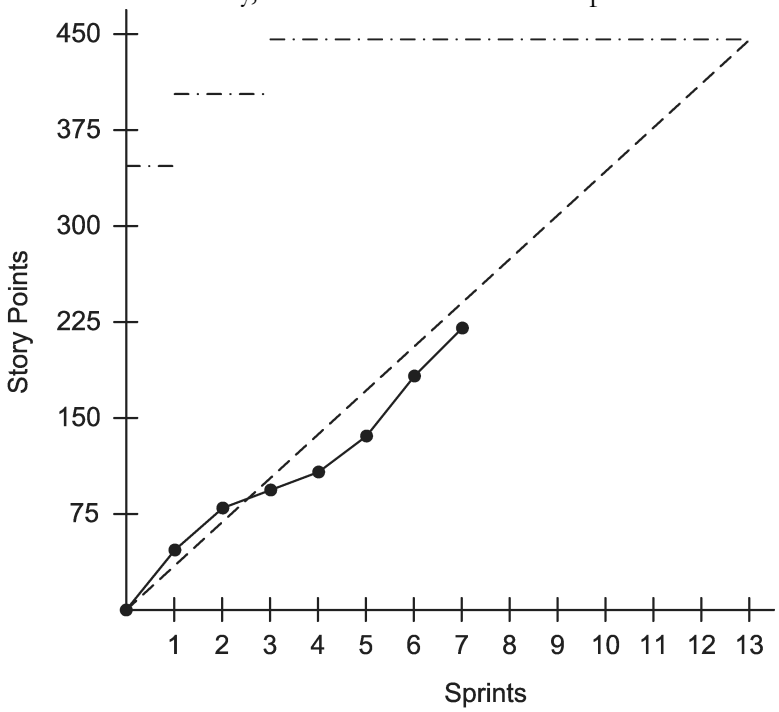

Figure 4: Release Earned Value Burn-Up Chart

The solid line with the black dots show the number of story points completed after each sprint, which is the earned value. The project started off well, with earned value above planned value for the first two sprints. But the team's velocity slowed in sprints three and four, bringing earned value below planned value (in other words, the project fell behind schedule). Velocity picked up again in the last several sprints, but not enough to make up for lost production. It may be necessary to either reduce the scope of the release by getting rid of some features (and hence story points), or adding a sprint or two to have time to finish all the features.

Figure 5 shows a sprint burn-down chart. This chart shows an entire three-week sprint. Note that there are 15 days in three work weeks, so there are 15 days on the horizontal axis. The sprint backlog started with 31 story points. By the second day of the sprint, PBIs with a total size of five story points were finished, so a point was plotted to show that 26 story points remained to complete after day two. After day three, another PBI with a size of two story points was completed, so a point was plotted showing that 24 story points remained. The other points were plotted similarly. Note that points are only plotted when PBIs are completed, and that the story points plotted are those remaining as estimated at the start of the sprint. This means that the solid line shows earned value and the dashed line shows planned value for the sprint.

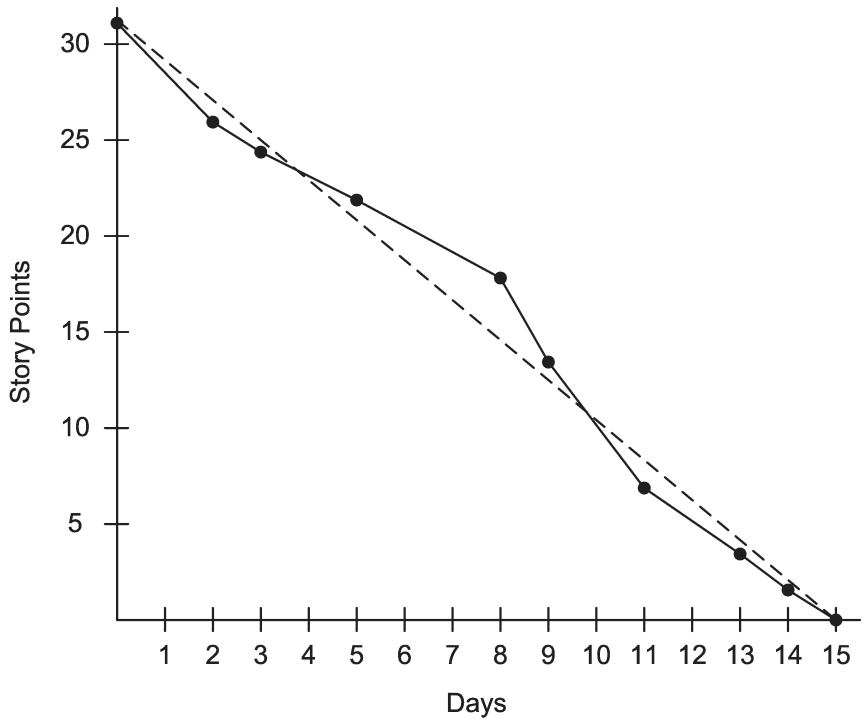

Figure 5: Earned Value Sprint Burn-Down Chart

The chart shows that earned value was ahead of planned value at the start of the sprint, then fell behind planned value in the middle of the sprint, but then earned value converged with planned value by the end of the sprint. If the solid earned value line had met the horizontal axis before the fifteenth day, then all the items in the sprint backlog would have been completed faster then expected, and the team could negotiate with the PO to augment the sprint backlog. If the earned value line had not met the horizontal axis by the end of the sprint, then one of more PBIs would not have been completed, and they would have gone back into the product backlog. In any case, the number of story points completed by the end of the sprint would be the team's velocity for the sprint. From the chart above, we see that the team's velocity for this sprint was 31 story points.

In addition to plotting earned value compared to planned value, it is also helpful to track the estimated effort still remaining as work progresses. Sometimes earned value management can be misleading because it does not reflect new knowledge about the extent of the work, and because it does not count work until a PBI is completely finished. Continuously estimating effort remaining provides a means of projecting where a team will end up at the end of a sprint, which may influence the way people arrange their work. For example, a team that thinks it will not finish the work in its sprint backlog might forego some background research or other tasks to concentrate on building the product, or the team might decide to concentrate on a few higher-priority PBIs so that it maximizes the value it generates from the sprint. Consequently Scrum teams often forego earned value management when tracking progress in sprints, and instead use burn-down charts to plot estimated effort remaining.

Figure 6 shows such a burn-down chart (in which the number of ideal hours per day is constant). This chart displays the state of the sprint after 11 days. Note that the vertical axis is estimated ideal hours of effort remaining. The team estimates the effort remaining and plots a point every day. This is not as onerous as it seems, because in sprint planning the team has already generated and estimated all the tasks for each PBI in the sprint backlog. The team only needs to adjust the estimated time remaining for the tasks that it has worked on that day, and perhaps modify estimates

for tasks that it now understands to be different. In this case, the team appears to have realized on the third day of the sprint that it had considerably under-estimated some tasks, and so the estimated time remaining actually increased quite a bit that day. Despite progress, the team is clearly not on a trajectory to finish all the work in the sprint by day 15. Presumably the team would make adjustments in anticipation of not completing all PBIs in the sprint backlog.

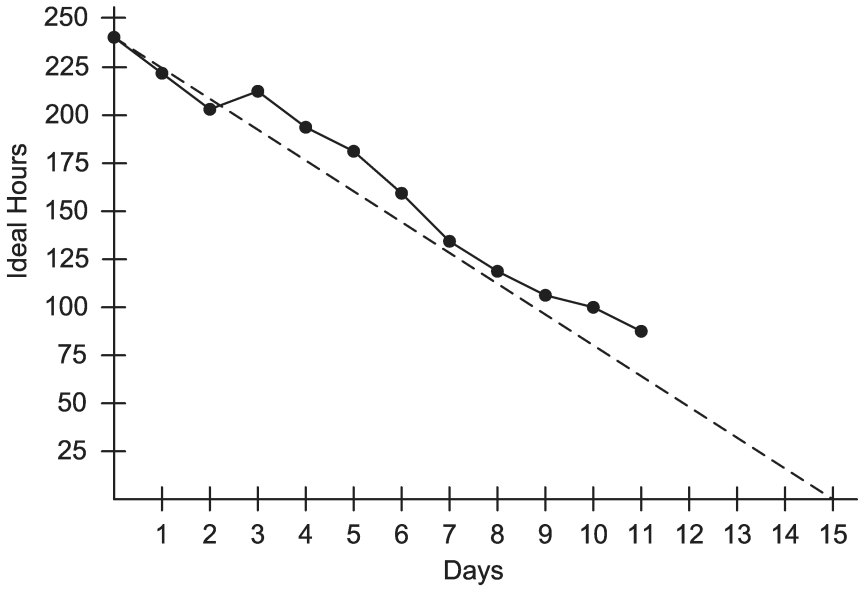

Figure 6: Sprint Effort Remaining Burn-Down Chart

In general, earned value burn-up charts are used more for project and release tracking, while estimated effort remaining burn-down charts are used more for sprint tracking. However, either kind of burn chart can be used for either kind of tracking.

Task Boards

A **task board** is simply a board showing the current status of all tasks to be done in a sprint. In its simplest version, a task board is a portion of a wall in a public location divided into four columns labelled "PBI," "To Do," "In Progress," and "Done." The wall is also divided into rows, one for each PBI in the sprint. At the start of the sprint, all the PBIs and the tasks to be done to complete each PBI are written on cards or sticky notes. The PBI cards define rows, and the task cards are placed in the PBI rows in the To Do column. When a team member is free to work on a task, he or she takes a task card from the To Do column and moves it to the In Progress column (in the same row), perhaps writing his or her name on the card. When the team member finishes a task, he or she moves it from the In Progress column to the Done column. The figure below shows an example task board.

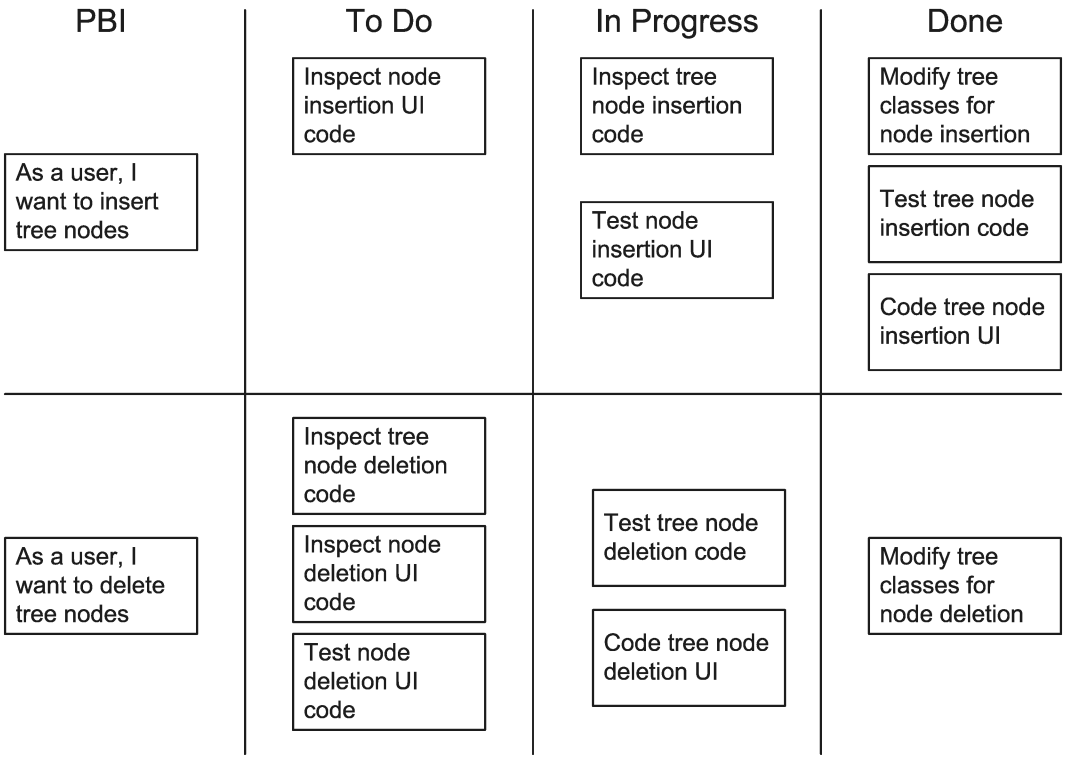

Figure 7: A Task Board

As this description should make clear, task boards are very easy to set up and maintain. They show the status of every task in the sprint, and a quick look at the board reveals where the bulk of the tasks are, and hence how far the work of the sprint has progressed. They are thus a valuable tool for tracking progress. They can also reveal problems. For example, if the bulk of the tasks are in the To Do column late in the sprint, then there may be a problem finishing many PBIs on time. Or if a task is stuck in the In Progress column for a while, the developer working on this task may need help.

Task boards also help solve problems. For example, a team may be able to see from the task board that it will not complete several of the PBIs in a sprint, but that it can finish some of them if it concentrates its efforts on the tasks for certain PBIs. The team may then decide to concentrate on certain PBIs to maximize the value produced from the sprint. In the task board above, the team is closer to finishing the first PBI than the second. If the end of the sprint were approaching the team could concentrate on moving the tasks in the first row across the board, thus completing that PBI by the end of the sprint.

Tasks board may be elaborated in several ways. The In Progress column may be further divided to indicate steps in task completion. For example, it is common to add a column labelled "To Verify" or "In Testing" between In Progress and Done. The PBI cards might also show estimated size, giving even more detail that might help in tracking progress or changing course.