Scheduling for Success with Critical Chain

by Rob Newbold, August 2010

Abstract

The Critical Chain scheduling approach described in this paper can be used to create "good" schedules that have provided substantial benefits in speed, predictability, and efficiency across many industries.[1] The first section of this paper explains the importance of a good schedule for managing an individual project, and describes what is needed for a schedule to be "good." The second section, "Constructing the Schedule," shows how ProChain builds Critical Chain schedules with the needed attributes. The third section, "Using the Schedule," describes how a good schedule can be used to gain significant benefits.

1. A Good Project Schedule

A project schedule is only one part of a successful project management methodology, but a good schedule can provide the basic framework for excellent project management.[2] Some of the benefits that accrue from good schedules include:

- Credible schedules create consensus about priorities, enabling teams and organizations to reduce multitasking and to align on where to focus.
- Up-to-date information on where to apply attention and resources anticipates problems and keeps them from spiraling out of control.
- Better handoffs and better preparation for critical tasks create faster, more efficient flow of work.
- Portfolio decisions improve when you have realistic commitment dates and a better understanding of resource requirements.
- Improved predictability, for individual projects and across the portfolio, allows better decisions by both internal decision makers and investors.

In other words, a good schedule is essential to synchronizing project work in ways that help the organization.

Before considering how to construct and use a good project schedule, we must first ask what that means, because not just any schedule will do. A good schedule must have a few key attributes. First, it must be *credible* to all those associated with the project. Otherwise it will not be used.

Second, building that credibility will require that the schedule and the processes that produced it be *owned* by project team members and management. If they don't own it, it won't be part of their solution.

And third, the schedule must be *used*. That means, for example:

- Used to set priorities, both within and between projects. Inability to set priorities through credible schedules results in increased emphasis on fixed due dates and increased multitasking.[3]

---

[1] Numerous examples are available; see, for example, (Newbold 2008, p.5).
[2] The importance of project planning in general is gaining more and more formal recognition, as shown by its move to prominence in (PMI 2008).

1

- Used to communicate status – status of tasks, projects, and portfolios.
- Used in analyzing possible actions and making meaningful predictions. Without the ability to make valid predictions, management is flying blind.

2. Constructing the Schedule

In order to construct a good Critical Chain schedule we need a systematic and rigorous process. This section describes such a process, consisting of five steps:

- Step 1: Build the Network
- Step 2: Level the Load
- Step 3: Calculate the Critical Chain
- Step 4: Protect Against Uncertainty (The Project Buffer)
- Step 5: Synchronize the Non-Critical Chain Tasks (Gating)

2.1 Step 1: Build the Network

In order to build a credible, broadly owned project schedule, we start by building a dependency network: a model of the required tasks and resources, and how they link together. The network should be based on the collective input of the project team.

The network should be built to achieve well-defined, company-level objectives. What are you delivering to the clients? What is making you money? Pick your objectives and your project endpoints carefully; the more commitments a team makes, the more there will be confusion about priorities, resulting in conflicts and multitasking. Most projects have a very small number of objectives that we really care about.

Tasks in a schedule should be real and actionable, preferably with verbs in the names and supporting detail attached. For example, suppose you have a task called, "Drawings Rev. 1." Maybe everyone knows what Rev. 1 Drawings are, but what's the work? Are the drawings supposed to be delivered, created, adapted, or purchased? What are the drawings of? With such an ambiguous name, how could you determine how long it will take, or who has to do it?

The connections between project tasks should, whenever possible, be real and well-defined. When you're not sure, start with your best estimate of what should happen, and use the schedule to make informed tradeoffs.

When possible and sensible, tasks should have resources assigned. Depending on the complexity of the project and the time available to schedule it, great detail may not be feasible.

If we plan for the behaviors that are already in place, we will plan for "business as usual"; we shouldn't expect things to change.[4] Therefore the project schedule should embody the behaviors we want to move towards, rather than the behaviors we see. It should assume no multitasking, which means using "focused durations" or "touch times" for task duration estimates. Each task's duration should reflect the amount of work

---

[3] For a detailed description of the logic behind this, see (Newbold 2008, pp. 14-15).
[4] Or: "Do what you did, get what you got."

expected to be done on it by whoever is doing the work, with no safety time added. Safety will be added in Step 4, to protect the entire project.

Other modeling elements can be added as necessary. For example, task restrictions such as "start no earlier than…" may be used to indicate information that can't easily be modeled as tasks, such as vendor deliveries. High-confidence durations may be used to show how long tasks might take as a "worst case" or "90% probability." During the scheduling process, those high-confidence durations will be used to adjust the contributions of individual tasks to buffers, as described below.

2.2 Step 2: Level the Load

Because we're looking for credibility, we must resolve any contention that tasks might have for scarce resources. Otherwise, if we assume there are no limitations on available resources, we may significantly affect the schedule's credibility and utility. This process of resolving resource contention is called "resource leveling" or "load leveling." During the leveling process we also consider limitations on the availability of resources or timing of tasks. Many software packages, including ProChain's, provide optimization functions that can help to minimize the pushing that can result from resource leveling. Note that leveling only improves credibility when focused durations are used; non-focused durations often include time to work on multiple things.

2.3 Step 3: Calculate the Critical Chain

The next step in the Critical Chain scheduling process is to identify the set of tasks that determines the project's earliest feasible end date. These tasks are called the "Critical Chain." They usually represent the most important places to focus management attention. In order to maintain credibility, the Critical Chain must take into account resource limitations. In a network with many parallel chains of tasks, there may be parallel sets of tasks that are all critical.

Figure 1 shows a simple project plan with three tasks (A, B, and C) feeding into a milestone (the black diamond). Each box represents a task; the arrows indicate links between tasks. In the figure, Task A is required to finish before Task B can be started, and Tasks B and C must both finish before the endpoint can be achieved. In addition, Tasks B and C both need the resource "Artist."
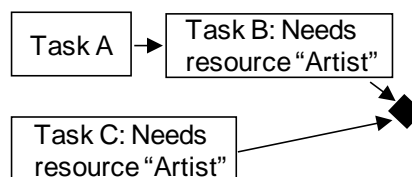

Figure 1: Basic Project Plan

Tasks A and B comprise the traditional critical path. However, if we have only one Artist resource, the critical path would not represent the most important tasks to focus on and the timing of the endpoint would not be credible. After leveling on the Artist resource, we have the picture shown in Figure 2. The Critical Chain consists of Tasks B and C (and, of course, the endpoint).
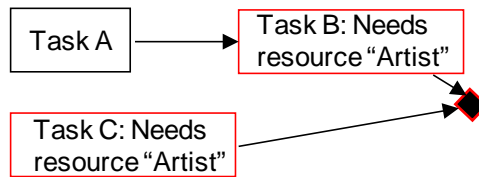
Figure 2: Leveled Project Plan and Critical Chain

2.4 Step 4: Protect Against Uncertainty (The Project Buffer)

By performing steps 1 to 3, we have created and validated a project network and identified the most important "Critical Chain" tasks. In addition, our resource-leveled schedule shows where the endpoint will fall in time; this could be called the "earliest feasible" delivery. Unfortunately, the "earliest feasible" delivery may be highly unlikely, especially if we have truly used focused durations, because in most projects there is significant variation in how long things take. Representing the delivery as single point in time would make us almost certainly be wrong. That means we haven't finished scheduling, because we haven't created a credible schedule. We need to calculate a range of times over which the project's deliveries are likely to be made. Let's start by considering components of that range. Figure 3 shows the earliest feasible finish of a small project, as determined by the Critical Chain (the red tasks). It also shows an estimated "latest finish" time.
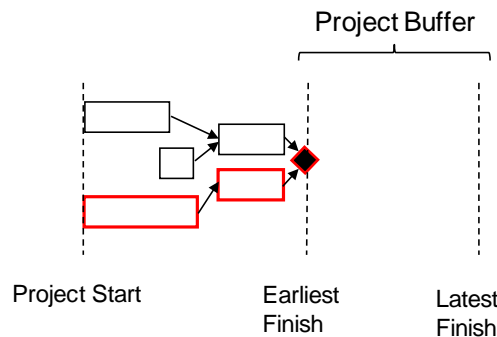

Figure 3: The Project Buffer

If we wanted to be in big trouble, we'd publicly commit to the "earliest finish."[5] If we could only announce one date publicly and wanted to have a safe commitment date, we'd probably announce the "latest finish" for the project. This range of times, from "earliest finish" to "latest finish," is also known as a "Project Buffer." The Project Buffer is a time range. It's called a buffer because, when we make it explicit, we can use it to protect project endpoints from variations in the time required to attain those endpoints.

In order for the Project Buffer concept to be useful, we need to decide on a credible duration for it. To do that, we break its duration into two components. First is variability due to tasks on the Critical Chain; second is a phenomenon we call "integration risk." These components are described in the following sections.

---

[5] This would put us in trouble, not just because the "earliest finish" has a low probability of coming true, but because announcing it would jeopardize the credibility of the entire schedule.

2.4.1 Critical Chain Variability

Some of buffer needed for the endpoint will come from tasks on the Critical Chain. That is, if we analyze the possible variation on Critical Chain tasks, we can calculate a safe completion date ("latest finish") for the project.[6]

This "critical chain variability" component is typically calculated as a percentage of the Critical Chain's duration. That gives a simple and predictable model that experience has shown to be effective, and avoids the kind of under-estimation that we've commonly seen with (for example) Monte Carlo simulation. While we normally use 50% of the Critical Chain to size project buffers, smaller percentages can work with projects having lower variability (e.g. some construction projects) and larger percentages can work with projects having higher variability (e.g. some software development projects).

Sometimes tasks along the Critical Chain have more or less variability than average and this needs to be reflected in the Project Buffer. For example, the time to grow a particular bacteria culture may have little variability; the programming of a new software module may have a great deal. So it can be useful to specify high-confidence or worst-case durations, in addition to focused durations, in order to indicate the unusual variability of such tasks. If a Critical Chain task has a high-confidence duration specified, the difference between high-confidence and focused durations is used in sizing the Project Buffer. If the high-confidence duration isn't specified, it's defaulted to be twice the focused duration.

As a simple example, suppose that in Figure 2, Task B and Task C both have ten-day durations. The Critical Chain length is 20 days, so when using 50% buffer sizing the Project Buffer duration would be 20 x 50% = 10 days. If instead Task C had been modeled with a high-confidence duration of 40 days, Task C's variability contribution would be (40 – 10) = 30 days, rather than 10 days. The resulting Project Buffer size would then be (30 + 10) x 50% = 20 days.

2.4.2 Integration Risk

If we looked only at the Critical Chain tasks, we would leave out important sources of variation: where the non-Critical Chain tasks integrate with the Critical Chain, and where there are multiple Critical Chain tasks that integrate together. Consider, for example, Figure 4.

---

[6] This can be a simplification; there are times when it may make sense for a high-variability task that is not on the Critical Chain to contribute to the variability portion of the buffer.
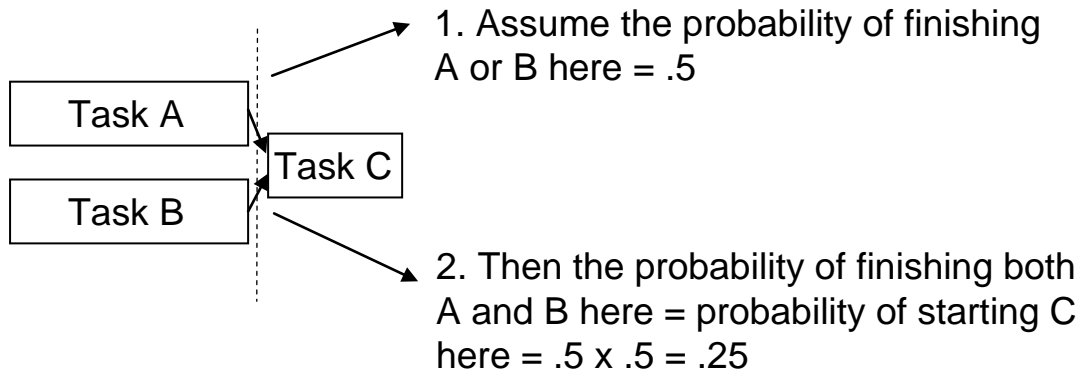
1. Assume the probability of finishing A or B here = .5

Task A

Task C

Task B

2. Then the probability of finishing both A and B here = probability of starting C here = .5 x .5 = .25

Figure 4: The Effect of Integration Risk

Suppose Tasks A and B, which have identical durations, must both complete before work can start on Task C. If the probability of completing Tasks A and B by the dotted line is .5 (50%), and Tasks A and B are independent, then the probability of starting Task C is the product of Tasks A and B's probabilities: .5 x .5 = .25. If we had three such tasks being integrated, the probability of starting Task C at the early time would be .5 x .5 x .5 = .125.[7] This integration phenomenon implies that even if the durations of Tasks A and B are on average correct (say, Task A takes a little more time than expected and Task B a little less time), Task C may still be delayed; it can't start until the later of the two predecessors completes. We call this potential delay "integration risk," because it represents a real risk of delay at an integration point, and it can be managed as a risk.[8] Integration risk can add significantly to our needed project buffer.

To calculate a precise time at which Task C would have a 50/50 chance of starting, assuming precision is possible, we'd need to find the point where each predecessor has about a 70% chance of finishing. Multiplying 70% by 70% for Tasks A and B would give a 49% (roughly 50%) chance starting of Task C at that point. This is shown in Figure 5. The time between the two dotted lines is additional buffer time we need to allow for the integration risk. More tasks being integrated would require more allowance for integration risk.

---

[7] It's easy to simulate the durations of Tasks A and B using coin tosses. Suppose each toss represents a day of work, and the task is completed when you get heads. Tasks A and B each have a 50% chance of finishing after one day or one toss. The chance of starting Task C after one day is 50% x 50% = 25%, because both Tasks A and B have to get heads in one toss.

[8] This phenomenon is also known in project management literature as "merge bias."

1. Assume the probability of finishing A or B here = .7

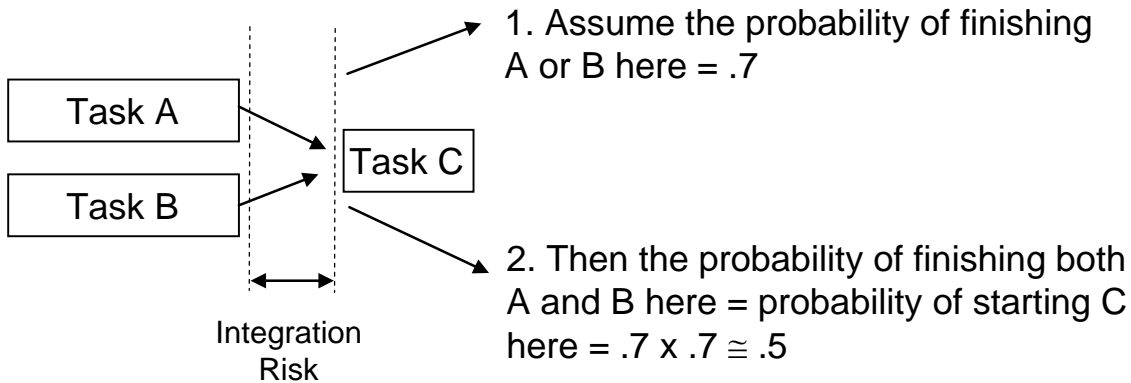2. Then the probability of finishing both A and B here = probability of starting C here = .7 x .7 $\cong$ .5

Figure 5: Calculation of Integration Risk

Note that non-critical chain tasks should be scheduled early enough that their disruptive effect on the Critical Chain is minimized. The simplest way of doing that is by scheduling them as soon as possible; the recommended alternative, "gating" them, is discussed below as part of Step 5.

There are various mechanisms in common use to calculate integration risk. Critical Chain has traditionally used a concept called the "Feeding Buffer" to decouple non-Critical Chain tasks from the Critical Chain. We have found this approach to be adequate and easy to explain for simple projects. It can also produce illogical results and be difficult to analyze in complex, real-world projects.

Some people use Monte Carlo simulation, although we've found that this approach also creates complexities in understanding and analysis. ProChain uses a mechanism that calculates results that are similar to Monte Carlo simulation, with more consistent answers and without running multiple trials. The ProChain mechanism is both intuitive and easy to analyze; it can be used to identify the key integration points and to analyze the highest-variability chains that feed them.

2.4.3 The Full Project Buffer

The project buffer is a single entity (usually represented in a schedule as a task) containing both a variability component and an integration risk component. Both components are important in protecting the project's commitment dates from disruptions.[9]

By breaking the Project Buffer's duration into these two components and analyzing them, we gain a certain amount of visibility into the reasons for its size. Ultimately – as with the rest of the project schedule – the Project Buffer durations should be evaluated for credibility by the project team and management. They need to use their experience, their schedule, and analysis tools to answer questions like, "Are the Project Buffers adequate to protect against likely problems and delays?" and "Does the overall schedule reflect the variability that we see in our projects?"

---

[9] When using Feeding Buffers to calculate integration risk, some people allow the Feeding Buffers to push out the Critical Chain. In that case, the integration risk protection is not incorporated in the Project Buffer itself, but in gaps in the Critical Chain.

2.5 Step 5: Synchronize the Non-Critical Chain Tasks (Gating)

Critical Chain tasks are scheduled as early as possible; by definition, they can't logically be scheduled any earlier. However, non-Critical Chain tasks do have slack time, and consequently we have some latitude in scheduling them. They should be synchronized with the Critical Chain in such a way that we minimize the needed Project Buffer.

The obvious solution is to schedule all tasks as soon as possible. However, non-critical tasks may not be needed right away, and there can be advantages to scheduling them later. This idea of starting tasks "when needed" (rather than as early or late as possible) is called *task gating*. Gating can help delay project expenses until they really need to be incurred. It also helps to minimize the number of active tasks, and hence temptations to multitask or micro-manage. Tasks are typically not gated if work has already started, under the assumption that work that's already "in process" should be finished with a minimal amount of multitasking. A project manager usually has the authority to start work on gated tasks earlier if he or she deems it prudent.

The integration risk calculations tell us how late we can schedule non-Critical Chain tasks while still minimizing integration risk. If a chain of non-critical tasks can be moved early enough that there is no integration risk where it feeds into the Critical Chain, we don't need to compensate for possible delays at the integration point by adding to the Project Buffer. This "early enough" time is known as the *gating time*.

Figure 6 shows the full schedule created from Figure 2, with a Project Buffer at the end represented as a task.[10] Task A is scheduled early enough to avoid integration risk (and therefore an impact on project completion), but otherwise it's as late as possible.
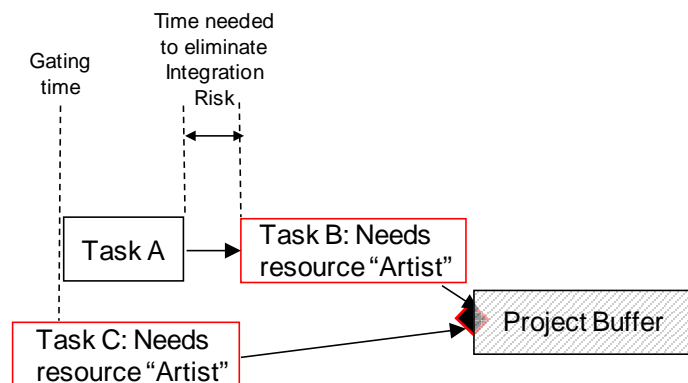

Figure 6: Full (Gated) Critical Chain Schedule

3. Using the Schedule

At this point in the process we have taken important steps towards building a credible schedule (one of our "good schedule" requirements), but we haven't discussed the other pieces: setting priorities, communicating status, and analyzing the likely impact of various actions. We'll look at schedule analysis first, because a project team typically

---

[10] In software scheduling tools, Project Buffers are typically represented as tasks. However, they aren't normal tasks in that they have no work and their scheduled times don't change when the schedule is updated. Consumption of Project Buffers, which occurs when delays cause tasks to push into the buffer they feed, is generally represented by a changing percentage complete of the "buffer task."

does significant analysis to improve schedule quality and timing before communicating priorities or status outside the team.

3.1 Schedule Analysis

There are a number of areas we can analyze in order to understand and minimize the time required to complete the project. These analyses can provide great value in suggesting ways to speed up project completion.

3.1.1 Validate

Is the network a valid representation of what we expect to happen? Does the Critical Chain make sense? Are the links, resources, and durations reasonably correct? Fix the obvious problems before doing anything else, because in-depth analysis efforts to reduce the duration of an invalid network will usually waste time. We can use the Critical Chain to determine where our validation efforts will be most fruitful. This helps reduce the time spent scrutinizing less critical portions of the network.

3.1.2 Analyze Critical Chain

Reducing the duration of Critical Chain tasks will reduce the duration of the entire project. We conduct this analysis early, because shortening the Critical Chain will usually result in shortening the Project Buffer and moving it to the left, which means we also bring in our commitment date. Besides correcting errors, this analysis should include real challenges to the schedule assumptions and the status quo. Can we bring in additional resources? Are all the planned features essential to project delivery? Can we do work "at risk" and thereby break links? What are the tradeoffs? The results of scheduling, analyzing, and re-scheduling allow us to analyze tradeoffs and make good decisions.

3.1.3 Analyze Project Buffers

Do we need all of the Project Buffer that has been allocated? In other words, do we understand and accept where the uncertainty in our commitment date is coming from? Do we need more time?

The variability component of the Project Buffer is normally determined by the duration of the Critical Chain tasks and (if specified) their high-confidence durations. These elements can be analyzed to determine whether the variability component seems reasonable.

By looking at the integration points, we can also explore the impact of integration risk on the start and finish dates of Project Buffers. This analysis is not usually a good way to spend much of our time, because integration risk is by definition based on multiple chains. For example, where you must integrate five parts, chances are you'll have to analyze several to noticeably reduce your integration risk. However, we should still validate that the integration risk is credible and that it's not significantly over- or under-stated.

3.2 Creating and Communicating Project Status

In order to maintain ongoing credibility, a schedule and its associated buffer status must be kept up-to-date. We find that it's usually adequate to do this on a weekly basis, although in some environments it's helpful to update a schedule more often.[11]

When an update occurs, there are typically three types of changes that can affect the schedule:

- Adjustments to the structure of the work needed, such as added or deleted tasks, links, resources, or calendar time;
- Updates to remaining durations of the tasks themselves; and/or
- The current ("as-of" or "status") date.

Based on the updated schedule data, we need to compute the current picture: how much Project Buffer has been consumed, which tasks are now most critical (the current Critical Chain), and possible changes to task gating.
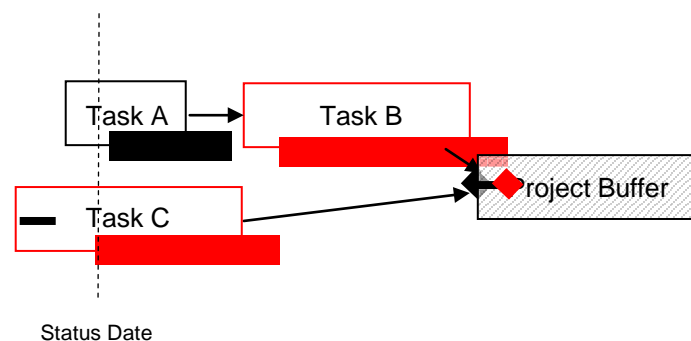

Figure 7: Updated Schedule

Some of the changes that occur after a schedule update are illustrated in Figure 7, which is based on the schedule from Figure 6.

- Move the current or "status" date out to its new time.
- Apply any needed updates to the schedule. In Figure 8, the horizontal black line in Task C indicates that some work has been done.
- Re-calculate task times based on current information. This includes resolving resource contention based on the current picture and re-calculating needed gating times. The solid red and black bars in Figure 8 show how the task schedules have changed from the originals.
- Re-calculate the Critical Chain. In Figure 7, the solid red bars represent the Critical Chain; in this example, it still consists of Tasks B and C.
- Based on the re-calculated times, calculate the amount of incursion into the buffers by the endpoints that feed them. This is converted to buffer % consumed.

---

[11] A simple rule of thumb would be to update schedules at least every 2% of the overall project duration, but no less than weekly. A one-year project might be updated weekly, but it would be reasonable to update a two-month project daily. When even a long a project is coming down to the wire, daily standup meetings can be useful.

Figure 7 shows some Project Buffer consumption due to the delay of Tasks C and B.

A popular means of communicating project status is the Fever Chart, shown in Figure 8.[12] The horizontal dimension is the percentage of the Critical Chain that has been completed, with the left- and right-hand lines representing 0% and 100% complete, respectively. The vertical dimension is the percentage of Project Buffer that has been consumed, with the top line representing no buffer left (100% consumed). If the project's status point falls in or above the red zone, the project is likely in trouble; the yellow zone indicates potential trouble; and the green zone indicates that things are probably on track. The boundaries between the red, yellow, and green regions are slanted. This reflects the fact that late in a project, less buffer is needed to maintain confidence in the safe commitment date. For each schedule update, a new point is added to the picture. In that way the Fever Chart provides a good sense of project progress to this point, and helps you to extrapolate the potential for the project to be late. Figure 8 has four data points, representing the initial schedule and three updates.
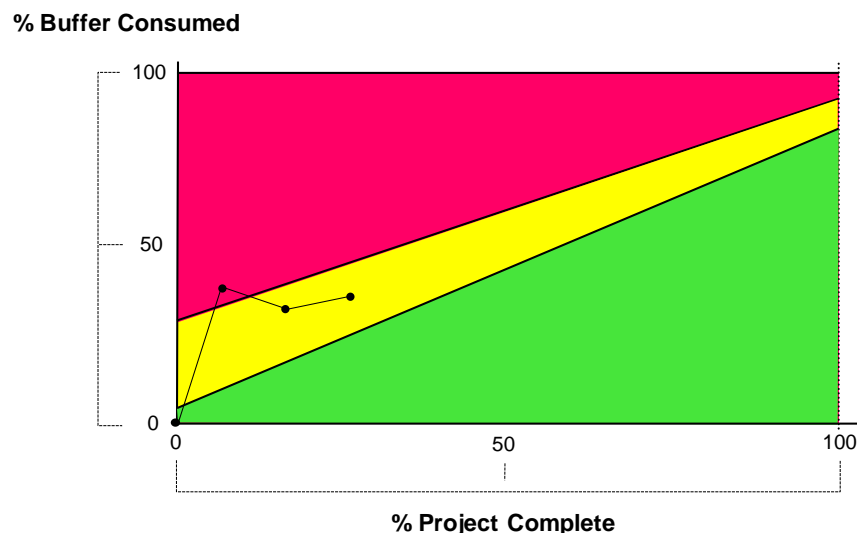

Figure 8: Fever Chart

3.3 Setting Task Priorities

Task prioritization is an essential output of any good scheduling process. Clear, stable priorities help people to focus attention where it's needed and reduce multitasking. When prioritizing tasks, the first place to look is at the current schedules, including current task status and task gating. Since we gate tasks so that they don't start until they're needed, tasks scheduled to start tomorrow will almost always be lower priority than tasks scheduled to start today.

Now suppose you have a number of tasks that could start today. Which do you work on next? There are many factors that can be considered in prioritizing tasks, including how the tasks impact project endpoints, the relative priority of the tasks' projects, and the current status of those projects. The variety of factors and the breadth of

---

[12] For more information on this, see (Newbold 2008), chapter 10.

possible considerations suggest that we can't possibly find a single numeric measure that can be used to prioritize tasks "optimally."

For individual projects, we've found "Percent Task Impact" (PTI) to be adequate. It is defined as the impact a task would have on the Project Buffer if the task were to be delayed. It is a combination of available slack and existing buffer consumption. If, for example, a project buffer is 20% consumed, a critical chain task feeding it would have a PTI of 20%; non-critical chain tasks would have lower PTI values.

4. Summary

In order to construct a good Critical Chain schedule, you must build a credible project network, level the load on resources, identify the chain of tasks that determines project completion, schedule the non-critical tasks earlier to protect the Critical Chain itself against merge bias or "integration risk," and protect the customer (delivery) against disruptions along the Critical Chain by creating a Project Buffer. If non-Critical Chain tasks can't be moved sufficiently early, time must be included in the Project Buffer to protect against integration risk.

During both network creation and updating, the schedule should be analyzed for opportunities to complete the project more quickly.

The Critical Chain updating process keeps the picture up-to-date by calculating updated task times, buffer status, and relative task criticality. This process, coupled with ongoing buffer analysis and timely management actions, helps to ensure that Critical Chain projects meet their schedule, scope, and cost objectives.

This scheduling and updating process produces what we defined at the beginning of this paper as a "good" project schedule:

- The Critical Chain scheduling process is designed around *ownership* and *credibility*, from the team-based network building to project buffering to regular project updates.
- *Task priorities*, within and between projects, are set using information about tasks' impacts on the Project Buffers.
- *Project status* is communicated through Project Buffer consumption and the Fever Charts; *task status* is communicated through measures of how close a task is to being on the Critical Chain.
- With the credible schedule and up-to-date project status, we can *analyze* possible management actions and predict their impact, thus focusing our time and resources in areas where they'll do the most good.

All these factors help companies that adopt the Critical Chain scheduling process to gain significant improvements in speed, predictability and efficiency.[13]

---

[13] There are many documented results of the effectiveness of Critical Chain scheduling. See, for example, (Newbold 2008, p. 5).

Key Terms

Buffer consumption
Buffer consumption is a percentage calculated during a schedule update. It is the amount of time that an endpoint's optimistic finish goes past the start of the Project Buffer it feeds, divided by the buffer duration, times 100.

Credible dependency network
This forms the basis of a Critical Chain schedule: an activity-based model of the required work with validated links, resources, and durations.

Critical Chain
The Critical Chain is the set of tasks that determines the timing of a project's endpoint(s). If you're going to focus anywhere, it should be on the Critical Chain tasks.

Fever Chart
The Fever Chart is a picture of how project status has changed over time. The dimensions are Project Buffer % consumed versus % Project Complete – essentially, 100 – (current Critical Chain length) / (original Critical Chain length). Each point on the Fever Chart represents a schedule update.

Focused durations
Today, people in your environment may not be able to focus on their work effectively. A "focused duration" for a task is the time expected to be spent on that task, if it were worked start-to-finish with an absolute minimum of interruptions. If you plan for people to spend focused time by specifying focused durations in your schedules, you have a much better chance that people will respect the need for focused work.

High-confidence (low-risk) durations
These are the "worst case" or "90% probability" durations for tasks. The difference between the low-risk and focused durations tells us the contribution of individual tasks to the buffers they feed.

Integration risk
Work at an integration point may only start when all the components being integrated are present. Integration risk (also known as "merge bias") is the risk that one or more components feeding an integration point will be late, thereby delaying the start of work at the integration point.

Project Buffer
The Project Buffer is the range of time over which you expect a project delivery to be completed. It protects your customer from variations in the time needed to perform the work. The project buffer end dates are usually published as project commitment dates.

Task gating

"Gating tasks" are tasks that have not been started and that have no predecessors. These tasks will be scheduled to start later than "as soon as possible," if delaying them is unlikely to affect the project completion date.

References

Newbold, Robert C. 2008. *The Billion Dollar Solution: Secrets of ProChain Project Management.* Lake Ridge, VA: ProChain Press.

Project Management Institute (PMI). 2008. *A Guide to the Project Management Body of Knowledge, Fourth Edition.* Newtown Square, PA: Project Management Institute.