# *Project Management Series: Part 1 – Defining & Planning the Project*

## *OVERVIEW*

Hello and welcome to the Project Management Series. My name is Steve Barger. I'm a partner and consultant with Greenbrier & Russel.

And I'm Rich Skibski, also a consultant with Greenbrier & Russel.

The Project Management Series consists of three courses. In Course One, we will discuss Defining and Planning the Project. In Course Two, we will look at Scheduling and Managing Resources and in Course three, we will address Managing the Project.

## *WHAT IS PROJECT MANAGEMENT?*

Steve Barger: What exactly is project management? Project management includes the techniques and skills required to manage the following areas:

> Defining project goals and scope
> Juggling the time vs. cost vs. scope tradeoff
  (a very difficult task for the project manager)
> Creating the optimum project work plan
> Building accurate project estimates

Now all of these concerns surround your project team, since they're really the most important piece of the puzzle. If these four surrounding areas are managed well and the project team that you've assembled is talented and committed, then the chances for a successful project are greatly increased.

## *AGENDA*

Let's start by going through the agenda for this course, Defining and Planning the Project. First, we'll summarize the objective of project management. Next we'll take a look at setting project goals and scope and their importance, especially early on in your project.

We'll take a look at balancing the time vs. cost vs. scope parameters and how those parameters can be used by the project manager to effectively manage the project.

Next we'll look at life cycles that you can plan your information technology project around. Finally, we'll take a look at some frequently asked questions that people commonly have about project management.

Copyright © 1998, Computer Channel, Inc. All Rights Reserved

## *PROJECT MANAGEMENT OVERVIEW*

We'll take a look at the objective of project management, so that we all have a common framework to work from. The objective of project management is to plan, control and lead a project so that it is on schedule, within budget, and delivered with the specified results.

ANNOUNCER: One of the most widely read authors and best known industrialist speakers in the world of information systems is James Martin, author of over a hundred books on computer technology in business.

He was named by Computer World as one of the 25 most influential people in shaping today's information environment.

James Martin: Why is project management so important? Because frankly, most application software development in business either fails or gives results which are not the right results for the business people.

It's absolutely essential that we have a life cycle which starts by knowing exactly what the business needs, prioritizing it, translating that into the Gantt chart, the PERT chart, the task scopes that have to be accomplished. Then have a very tight control over those tasks to make sure that nothing slips. When things slip - of course, everything is going to slip - you know about it and can take corrective action quickly.

It is absolutely disastrous to have a project that goes on for two years and nobody knows it's going wrong until the end of the two years. This is catastrophic project management.

We want to break everything down into small modules, where we can manage each module. When things go wrong - and they will - you know early and can take corrective action early.

### *The Role of the Project Manager*

Steve Barger: Every project needs a project manager. It certainly is one of the most important roles that exist within your project. The project manager must manage the expectations of the user and the teams. They must execute a vision and keep the team on track with that vision. They are responsible for acquiring resources and motivating personnel.

They make the hard decisions regarding your project goal: tradeoffs of time, cost and scope. They're definitely there to hold the line on quality. The buck stops with them as far as quality is concerned. They keep the team on track and in sync with the project goals.

A project manager is not a technologist. But they should understand and manage the technology, in order to be able to help their team further, especially in the new technology areas.

Above all, the project manager is the one who can get the job done.

James Martin: A good project manager is a pretty complex set of skills. They need to be very precise in breaking the project into the needed tasks. The project manager makes sure those tasks are well represented on a Gantt chart or a PERT chart and monitors very carefully to make sure that nothing slips.

But I think a particularly important characteristic of a project manager is to pull together the small, high-performance teams, which are going to do the critical part of the work. Team building is the art of selecting the right people, motivating them and putting them together so that, within the team, everybody helps everybody else to build up the highest level of excellence. The skill in team building is a very important project management skill that is often missing.

## *SETTING PROJECT GOALS*

Steve Barger: In this section, we'll take a look at setting the project goals and we'll give you some techniques to facilitate setting goals and scope.

We'll start by showing some guiding principles for setting the project goals. The first is to examine the ultimate benefits of the project. Next, we'll take a look at relating goals to one or more organizational goals.

Third, we'll put those goals into business terms that your users understand and can be common language for the rest of the team.

Fourth, we'll break down those project goals by functional area.

If we look at examining the ultimate benefits of the project, we want to distinguish the business benefits from the deliverables. For example, we want to decrease the customer wait time in the order entry process. That's an ultimate benefit of the project, not rewriting the order entry programs. That sounds like something that's come out of your technical area and the users aren't going to understand that.

You want to relate those goals to one or more organizational goals. For example, you might have a project goal of improving your response time in the order entry system. The organizational goal that matches to that project goal is to improve the customer service.

Next you want to put it in business terms. You want to put those project goals in business terms and relate those goals to the user's concerns. Use their

Copyright © 1998, Computer Channel, Inc. All Rights Reserved

terminology, use their jargon. Again, work with them to get those goals for your project defined.

For example, you might have a project goal that states "improve the response time in the order entry system, not decrease the screen I/O time." Something like that sounds like technical jargon.

Finally, you want to break down those goals by functional area. This will help determine which functional area will sponsor a particular goal. This will become very important later on when you're trying to determine how to prioritize those goals and which of those to include in the scope.

For example, marketing's functional area might have a project goal of improving customer targeting. Sales might have a project goal of increasing the sales volume. These are two valid project goals that need to be managed as you move forward in the process.

James Martin: In setting the goals of the scope for a project, the thing which most often goes wrong is that the technical people do not really understand the business, its needs and benefits.

I would say, "Look at every business today in terms of value streams and then get the people who really understand how to make the value streams as excellent as possible to be involved in workshops for value stream-oriented requirements planning which strongly relate to GUI prototypes. You've got a really good GUI builder and you can modify the prototypes very fast and you're making absolutely sure that nothing critical to that project has been forgotten."

It's amazing how often you get the Swiss cheese effect: you get to the end of a life cycle and you've got holes in the cheese. That's because you only missed things in the initial requirements planning and so we need to make sure that we ask the business people in the early stages of requirements planning, 'what have we forgotten? Is there anything missing?' We must make absolutely sure we don't have holes in the cheese.'

And then when you've got the requirements right, then you want the requirements to trace software which follows the requirements through into the code modules which are implemented.

## *SETTING THE PROJECT SCOPE*

### *Initial Steps*

Steve Barger: Next, we need to set the project scope. You've determined what the goals are and they're usually quite large.

Copyright © 1998, Computer Channel, Inc. All Rights Reserved

Your first step here is to determine what goals you can accomplish given the time and resources that you have available.

You'll do this by assigning a high-level cost to each goal and prioritizing those goals with the users.

Next, you'll gain consensus from the IS management and users. You might want to consider a phased approach here.

This is the first indication that you might have subsequent releases. All of the scope that you're trying to chop off simply won't fit within the time constraints that you have.

James Martin: Intelligent business people, when the system is being built, are always going to think up new things that they would like the system to do. You've got to control that.

It's really important to have specifications that are frozen and then you start to implement them. But sometimes, there's a good business reason for changing those specifications. They should only be changed with the project manager fully understanding the implications of the change and talking to business people about that change.

Very often the sensible thing to do is to say, "Let's have a RAD life cycle in which we're going to implement the system fairly quickly and when it's been implemented, then we'll go into a second edition, and then the third edition."

That way, when the business people think of the changes in scope or functions, you can implement those but not in this edition. You're going to save those changes for the next edition.

### *The Work Breakdown Structure*

Steve Barger: Finally, you want to document the work to be included or excluded. One way to document the included work is a work breakdown structure (WBS). At this point, since you're simply setting the project scope, it's probably a preliminary WBS, where you group the project elements together that will organize and define your scope.

It's very similar to an organization chart in that you start at the top and you break it down further and further to a finer level of detail.

### *The Responsibility Matrix*

The next thing that you want to do in setting the project scope is to identify the areas of responsibility for each of the phases and activities that you've identified in your work breakdown structure.

One of the techniques that we use is the responsibility matrix. At the top of a responsibility matrix, we have the phase project initiation. Under that we have the activities that are contained within that phase: discover a project opportunity, choose the project executive, assign the project manager (that's where you come in), assess the business risk and assess the technical risk.

We have roles that are defined for each of these activities, using this typical key that we'll use for those roles are: (1) primary or actual responsibility; (2) general supervision over that activity; (3) consultation; (4) consultation, perhaps after the activity is completed; (5) notification when that activity is complete and (6) final approval.

If we take a look across one of the rows in the responsibility matrix, we'll see a row that says Assess Technical Risk and if we take a further look, we'll see some of the roles that we've defined for our responsibility matrix.

The first role that we've defined is corporate and we've assigned a (4), which means that corporate may be consulted when we're performing the technical risk assessment.

Next role is the vice president of projects. The vice president of projects has general supervision over this activity.

Next is your business analyst who also has general supervision over this activity.

The project manager must be consulted when you perform the assessment of your technical risk.

Your team - if in fact, there is a team at this point in time - may be consulted. The client project executive has no real responsibility in this particular activity and the systems architect is the one that has actual or primary responsibility. You should note that you must have, at a minimum, an actual or primary responsibility assigned to one role.

### *Competences, Direction, and Risk*

The next step in setting the project scope is to determine the technical approach. You do this by first determining your technical core competencies.

Next, you'll consider your strategic technical direction. Where is the organization going as far as their technical direction is concerned?

One of the things that I like to do when I come into a new client is look at where they are today but, also, where are they heading.

Third, we'll consider the organization's risk tolerance. This is not something that's generally written down, but, if you've been with a company for little more than a year or so, you get a pretty good idea of what the risk tolerance of that company is, whether they are risk adverse or not.

### *Technical Issues*

Next, you'll identify the technology required to perform this project.

The next step in setting the project scope is to resolve conflicts over the technical issues. These are issues that affect major technology directions and it'll affect every phase of the project.

For example, I worked with a client that had not yet set their technical architecture during the design and into the coding phases of their project. The staff was pretty confused in terms of where they were going to go once that architecture was set. Subsequently, the project was delayed because of necessary rework.

Now, the next thing that we need to do is look at a risk profile of that selected technology. One of the tools that you can use to do this is what we call a technical risk assessment.

You'll ask the question, "How far do we go and how do we get there?" That technical risk assessment will assess each of the technical areas of your selected technology and rate them on a risk scale from one to ten.

Next look at the technical infrastructure. Can the technical infrastructure that exists today handle the new selected technologies?

For example, if you're installing a new client/server system, can your WANs and LANs handle the increased traffic?

You'll also want to take another look at your staff core competencies. Do I have the people in house that can do the tasks that I'm about ready to set out and do or will I need to train them or perhaps bring in some consultants?

### *The Negotiation Process*

The last step in setting the project scope is to negotiate with the users and the IS management. You should include them all along, but it's real important now as your closing this off that you negotiate with them to get agreement on all sides.

We implement a four-step negotiation process. We first ask our people to identify the problems and goals. In this case, we're trying to set the project scope.

We next go into exploration, an understanding of each person's position. By the time you've got the goals and you've started to go through the scope, you pretty much understand where each person is coming from.

Copyright © 1998, Computer Channel, Inc. All Rights Reserved

Third, you enter into a give and take. You explore the various options. " What if we didn't do this one piece, but we added these two? Would that give you better benefit? Because remember, we certainly can't do it all."

Fourth, we close. We make an agreement that's win-win. It's something that we can handle technically and something that gives the users the biggest bang for the buck.

## *TIME VS. COST VS. SCOPE*

Steve Barger: Now that you've got a handle on your goals and you've determined what goals will fit within your scope, it's time to address the three parameters: time vs. cost vs. scope.

These are critical parameters that the project manager must manage throughout the entire life cycle of the project. Sometimes these parameters are represented as three sides of a triangle.

I prefer to think of them as three balance points. If you add additional scope, you must add additional cost or time in order to balance that equation all out.

### *Time*

Let's take a look at the first parameter, time. Time is also referred to as the schedule or the project's timetable. I think we're all familiar with how much time we have to do our project, or - conversely - how much time we don't have.

The schedule is the high visibility goal near the end of the project.

### *Cost*

Cost is the second parameter of the time vs. cost vs. scope equation. Cost is simply the expenditures required for the project's resources.

During the design phase and the coding phase, cost is very hard to control because you have so many people working on design and coding tasks.

Cost is generally given to you. I don't know about you, but I normally am not given an open checkbook. The costs are assigned early on and I must maneuver my time and scope around the cost that I've been given.

### *Scope*

Scope is the third parameter of the time vs. cost vs. scope equation and it has many elements. Certainly, it includes specified results, which include functionality.

Copyright © 1998, Computer Channel, Inc. All Rights Reserved

What am I delivering to my users? Usability. How usable is this product? We need to get some level of definition there, maybe some examples, maybe even some prototyping at this point in time.

Performance is a piece that a lot of people forget. What is my response time, for example, for this given system? We should include that in the scope as well.

The technology is another element of the equation of the scope equation. Technology includes what technology areas we are moving into that perhaps we have not yet addressed.

Quality is sometimes considered part of the scope parameter. I prefer to think of it as a parameter that stands on its own in the middle of those three balance points. Quality is something that you, basically, can't waiver on. You must set the quality early and watch the quality as a project manager throughout the entire life cycle.

We noted that scope is difficult to control at the beginning of the project. That's because so many peoples' ideas are trying to be incorporated in and the goals are so broad that - as we try to define those goals - more and more is asked to be added.

### *Setting Expectations*

The last thing when you consider the time vs. cost vs. scope equation is to set reasonable expectations. First, don't oversell the technology. We're performing a project in order to deliver some business benefit to the users, not to try out some new technology.

Second, build contingencies for unforeseeable circumstances. Consider what might happen if the technology that you've selected just doesn't work together.

Obtain cost and time commitments from all and these can be incorporated in the project charter. Make sure everyone agrees to what those time and cost commitments are and is clear in terms of what their commitments will be.

### *Balancing Project Variables*

James Martin: You've got three variables on a project: time, scope and the resources which are applied to it. Very often we fix the scope when we fix the resources, but let the time slip.

Now if you've got a time critical project, instead you will say that you have got some resources and you're fixing the time. The scope may have to slip in order to deliver on the deadline.

Most creative people have a deadline. If I'm writing an article for a magazine, I've got a deadline and, if I miss that deadline, I'm dead. That's why we call it a deadline.

Therefore, it makes sense on some projects to say that there's an end date which cannot be slipped, but that you might slip the functionality. If you slip the functionality, you want to make absolutely sure that the critical functions are implemented and the ones that are less useful are slipped, if something has got to slip to meet the important deadline.

Then when you've met the deadline, the business people will look at it and they'll decide whether or not to implement it. They may say, "Well, before we implement it, add this function and add this function and then take it into production."

## *LIFE CYCLES*

Richard Skibski: When structuring a project, the three main life cycle approaches are: the waterfall approach, rapid application development (RAD), and the object-oriented (OO) approach.

When selecting which life cycle is appropriate for your project, there are a number of factors to examine. One is the project's deadline. Is it sooner or later? Another is the team's experience with the technology. Another factor is the size of the project.

There are also the disparities and the importance of the goals of the project. If there is a wide variance in the goals, you may choose one life cycle over another. The management's confidence in the team is also another factor that plays a part in how confident they are in extending a long project out, rather than chopping it up into smaller pieces.

Also, the project manager's experience with any life cycles comes into play, as does the percentage of the project's budget that is approved and the technology and any tools that are available to the team.

### *The Waterfall Approach*

In the waterfall or traditional life cycle, you follow the project phases sequentially. You would begin with analysis and, when that is complete, get on to design and, when that is complete, coding, through testing and implementation.

Similarly to a real waterfall, it is the quickest way from Point A to Point B, but can be very rocky at the end.

For a smooth implementation using a waterfall approach, limit your waterfall projects to ones that are small. This is easier to handle from a waterfall approach. Have approved budgets and also have a low amount of user interaction. For example, it is appropriate for systems that are using characterbased user interfaces that you might find on a CICS environment or that are strictly background functionality systems.

Waterfall life cycles are also good for projects that have a team that is experienced in using the technology where there aren't going to be as many technical questions along the way.

It's also good for projects where you are accomplishing goals that are equally important, where none of the goals has to be finished before the other goal.

Also, waterfall life cycles are good for projects that have major coexistence issues between the legacy and the new system. For example, if you're trying to integrate a new system with a legacy system - while you're trying to phase the new system in and there are a lot of costs associated with that effort - you may want to opt for a waterfall life cycle.

### ***Rapid Application Development***

If you have fewer coexistence issues or if you're designing a new system altogether, a RAD life cycle approach may be appropriate.

RAD involves chunking the project into little pieces. You treat each project goal as a separate but interconnected project within the scope of the entire project.

RAD life cycles are iterative. All the phases repeat and they often use prototyping to help flush out the requirements.

RAD life cycles are good for large projects, the many systems development projects that go on for years. Using a RAD life cycle, you can cut down the deliverables from one large one over years to several small ones scattered throughout the life of the project.

This makes it easier to coordinate and manage the project.

James Martin: I don't think I've ever known a life cycle where it takes two years to build the system and where things didn't go wrong. That's much more likely to happen now than in the past, because things are changing faster. Businesses are changing very fast indeed.

It's my view that no project of this issue be longer than six months. When possible, you ought to be designing the life cycle so that it's three months.

That very often means that you're designing systems so that you implement them stage by stage with three months for each stage. Then, as new ideas come along, you are deliberately designing the system so that you quickly go on to the next edition of the system.

Copyright © 1998, Computer Channel, Inc. All Rights Reserved

Richard Skibski: RAD life cycles are also good for projects that use technology that's new to the team. Smaller deliverables make it easier for the team to become comfortable with the technology while using it in a less risky setting.

RAD life cycles are also good for projects that have high user interaction, such as with GUI-based applications.

A RAD life cycle is also good for projects that accomplish goals with a wide disparity of importance. With this technique, you can get the higher priority goals installed first, thus giving the users more value than some of the other lower priority goals that you would install at the end of the project.

### *RAD and RAC*

James Martin: The biggest advantage of a RAD life cycle is that you get the business benefits earlier. Sometimes people think in terms of productivity of programmers, but getting the business benefits earlier is much more important than programmer productivity. Now all business is changing. Especially as we look out the cybercorp, the value stream way of looking at business, we're finding very fast changes indeed. Therefore, the something that's maybe even more important than Rapid Application Development is Rapid Application Change (RAC). Rapid Application Change means you've got to select tools and techniques that enable you to build something fast, but to build something that you can change fast. That fast change is becoming increasingly critical in the important business systems.

### *Additional RAD Projects*

Richard Skibski: RAD life cycles are also good for projects that have a partial budget approved and aggressive deadline. The users need to get something up and running, no matter how basic, as quickly as possible.

Using a RAD technique, you can get the basics up and running and then add on some of the frills later.

Finally, RAD life cycles are good for projects that are managed by a project manager that's skilled using RAD techniques.

### *The Object-Oriented Approach*

The object-oriented (OO) life cycle - similarly to RAD - uses an iterative approach. This technology facilitates reusability and is based on a component architecture. An object-oriented life cycle is more than just a life cycle. It's a technological direction for the entire project.

The phases in an object-oriented life cycle include conceptualization, where you determine which classes and methods you're going to relate to actual real world entities and processes.

Object-oriented analysis is used to further flush out those classes and methods. Design involves what the classes will look like and how the methods will work through implementation, testing and integration.

Since object oriented is such an iterative process, you are constantly adding and adding to the classes and methods that you built originally. Thus, you may be taking two methods away from a class and adding four and you end up retesting the whole thing.

Finally, deployment is the last phase in object oriented life cycle.

Object-oriented project management differentiates itself from RAD or waterfall approaches in that it is not fully mature. It has not been used as nearly as widely as the other two approaches.

The reusability that comes with object-oriented technology adds complexity to the running of the project and, thus, the project should be managed by project managers skilled in object-oriented techniques.

James Martin: When you're really going to try an approach, you've got to really build things quickly. What are the most effective methods in dealing with it? Use the most powerful tools. Use a code generator. Use object oriented. Develop a way - you've got a preexisting set of objects or components or templates - and then tightly control it so that you build things that you can create with those tools. If there is any function which will be outside that tool set, then slip it into the next edition. So it's saying, "let's get the first edition built and then the second edition is going to come later." Manage that very tightly indeed and understand that everything is likely to go wrong. Therefore, you break it up into small modules. When as soon as something goes wrong, you know quickly and you can take corrective action quickly.

Richard Skibski: Choosing a life cycle approach for your project whether it be object oriented, RAD or waterfall is very important to the success of your project. Choose it wisely.

## *FREQUENTLY ASKED QUESTIONS*

### *RAD Pitfalls*

Steve Barger: In this section of the Project Management Series, we're going to take a few minutes to look at some of the frequently asked questions that people commonly have about project management. Rich, what are some of the biggest pitfalls that you've encountered when running a RAD life cycle project?

Richard Skibski: Well, the first one, the main one is that people are unclear of what RAD really means. They hear Rapid Application Development and they think because it says "rapid," it just means that everybody works faster.

But the other thing that you run into - and this is from a project manager's perspective - is that you've got all these extra choices and decisions that you need to make.

You've got JADs. You've got prototyping. You've got case tools. All these decisions that you have to make now, which, in some of the older situations with waterfall approaches, you didn't have to make.

Finally, when the project is rolling, you've got all this effort that you have to manage. You've got people on one phase of the effort. There are other people on another phase and they are at different points in their development. You're trying to manage all of this and coordinate it.

And one of the key things, of course, is to make sure that you've sequenced your phases properly. You can't have a phase that is reading a database when you haven't implemented the phase that actually populates the database.

## *The Fixed Time Parameter*

Steve Barger: Have you ever had to back into dates? A situation where you had to fix time parameter? And if so, what happened?

Richard Skibski: You get that all the time. In today's environment, you're always trying to meet a deadline or meet a date. Usually, the project manager is managing the three parameters - time, cost and scope - and the time is usually fixed. But the project manager usually has a flexibility with the cost and the scope in order to successfully lead the project.

Now the problem comes in when not only the time but the cost and scope are set. In that situation you want to be clear that the estimates are good and that you feel comfortable with them before embarking on the actual project.

One of the projects that I've worked on that was in a situation like this with fixed time, cost and scope. The project manager tried to get around this problem by just merely having us all work harder. We all worked overtime.

Now as the project progressed, more and more people quit the team and ultimately, the project ended up being rescheduled.

## *An Overly Large Project Scope*

Steve Barger: Have you ever seen a project where the scope was too large, especially given the organization's technical core competencies?

Richard Skibski: I've seen it several times. A company has a strategic direction. They want to go to a new technology and they bring in the new technology. The first thing they want to do is use it and start building applications from it.

But in some cases I've seen, they're using new technology for some mission critical work for the first system. It's really biting off more than they can chew.

In a situation like this - where they've brought in the new technology - a company is best served by bringing in experts on the technology. The company should not just relying on their staff to do all the design and coding on the new technology, because typically these people have only had a weeks work of training.

### *Coexistence Issues*

Steve Barger: Have you ever seen any cases where there is some major coexistence issues between the new proposed system and a legacy system? And, if so, would you prefer a RAD or a waterfall approach?

Richard Skibski: Typically, when you've got major coexistence issues, you want to use a waterfall approach because it's difficult to phase in as you would with a RAD approach.

One caveat: if you've got a system that is so large, doing it as a waterfall could create a big bang. Now I've worked on a system where we were implementing over 1,000 new modules over one weekend. This didn't go very well and it went in as a waterfall. This probably should have gone in as a RAD.

### *Selecting the Project Manager*

Richard Skibski: Who usually selects the project manager? How is that done?

Steve Barger: Ideally, the project manager is selected based on the skills that he or she has and how that best fits the project. For example, you might want to bring in a project manager that's facilitative, especially if you're working with a project that has maybe a difficult project team.

Realistically, the project manager is selected because he or she is available.
Time Boxing

Richard Skibski: I noticed in some organizations that they use a technique called time boxing. Could you tell me a little bit more about this technique and how it fits in the time vs. cost vs. scope balance?

Steve Barger: Time boxing is setting very short periods of time that allow project managers and the team to select what will occur within that time and subsequent periods of time. You may time box, for example, a deliverable to exist within two months, another deliverable to exist within three months and so forth. Since your time is fixed, it allows them to determine what exactly you're going to accomplish in that period of time.

### *Phases: RAD vs. OO*

Richard Skibski: In the object-oriented life cycle approach, phases tend to overlap, whereas, in a RAD approach, phases will run in parallel. Can you explain the difference?

Steve Barger: Object oriented is more than just a life cycle approach. It's a whole technology investment that you're making. You're having all your people - users, designers, coders - thinking in terms of classes and methods.

Objects themselves tend to live beyond the actual application that you're working on currently. Often times, you go back into an object, pull the covers off of that object, maybe add new methods to that object and so an object lives on.

### *Managing the Pre-Set Project*

Richard Skibski: Now if you were assigned to a project as a project manager where that the time and the cost and the scope has already been set, how do you approach that?

Steve Barger: This happens quite a bit. Sometimes the project manager isn't around when those things are set, quite frankly. What I like to do when I walk into a situation like that is assess whether or not the amount of scope that you've bit off - if you will - can be done within the time and cost constraints.

If you feel pretty comfortable with that - and the user feels comfortable with that - then it's probably okay to continue on. But if you have some uncertainty there, you might want to raise the red flag right away.

## *COURSE REVIEW*

In this course, we have examined ways to define the project in such a way as to get it off the ground successfully.

We first looked at the goal or objective of project management: to effectively plan, control and lead a project so that it is completed successfully, on time, under budget, and with all the functionality requested.

We then looked at the determination of scope and goals. One of the key points is that you'll want to set your project goals so that they relate to one or more of the organizational goals.

We then examined one of the most important responsibilities of a project manager: making tradeoffs of time, cost and scope.

Finally, we looked at the appropriate life cycles that a project manager must choose from, given the factors of the project.

Copyright © 1998, Computer Channel, Inc. All Rights Reserved

Thanks for your time.

## *SERIES OVERVIEW*

In this first course, we took a look at some very critical steps in the project initiation phase: defining and planning the project.

In our next course, we'll take a look at scheduling and managing the resources, because you really want to get your project off to the right start.

If you have any questions, feel free to check out our Web site at www.gr.com or feel free to e-mail us at sbarger@gr.com or rskibski@gr.com

And enjoy the rest of the series.

Computer Channel, Inc.
www.compchannel.com

Copyright © 1998, Computer Channel, Inc. All Rights Reserved