

---

# Introduction

---

Every software project manager knows of certain steps to take at the beginning of a project. You develop a business case, specify the product requirements, obtain funding and management sponsorship, assign a project manager, assemble the team, acquire other resources, and develop estimates and a project plan. I'm so confident that you know about these essentials from your project management training or previous experience that I'm not going to address them here.

However, several other activities are also vital to getting a software development project off to a good start. Unfortunately, project managers sometimes gloss over these steps. Perhaps they haven't had enough experience to realize how important these steps are, or maybe they don't feel they can spend the time during the frenzy of project launch. But seasoned managers know that paying attention to these critical activities can separate success from failure. These practices clarify project expectations and priorities, let stakeholders agree on what "done" means, and ensure continual improvement by learning from previous projects.

This handbook describes many actions that lay the foundation for a successful project. It doesn't attempt to cover every aspect of project initiation, instead focusing on these less-obvious activities. Both experienced and novice project managers will find these practices valuable. The focus is on software projects following any lifecycle or methodology (including agile), but the practices apply just as well to many nonsoftware projects. The handbook contains six chapters, each of which addresses a specific set of related practices for a key activity area:

1. Define Project Success Criteria
2. Define Product Vision and Project Scope
3. Define Product Release Criteria
4. Negotiate Achievable Commitments
5. Study Previous Lessons Learned
6. Conduct Project Retrospectives

Each chapter begins with a short scenario, describing an actual experience I have had. Each story illustrates a problem that can arise if a project manager neglects that chapter's principles and practices. The chapter then presents a tutorial that will let you put the chapter's topics into action. Common traps to avoid are identified with an icon of a mousetrap in the margin. True stories from real projects are flagged with a newspaper icon in the margin. Cross-references are provided to related chapters and to the relevant sections of the Project Management Institute's Body of Knowledge (PMBOK), the Software Engineering Institute's Capability Maturity Model<sup>®</sup> for Software (or just CMM<sup>®</sup>), and the CMM Integration<sup>®</sup> (CMMI<sup>®</sup>). Each chapter includes several practice activities. Worksheets are included so you can apply the practices to your own project immediately.

In my view, there is really no such thing as project management. What we call "project management" is a composite of managing many other project elements: people, communication, commitments, resources, requirements, change, risks, opportunities, expectations, technology, suppliers, and conflicts. Nearly every project includes aspects of all these activities and the successful project manager must keep an eye on them all. Use this handbook to help you start your next project on a solid foundation.

---

<sup>®</sup> CMM, CMMI and Capability Maturity Model are registered in the US Patent & Trademark Office by Carnegie Mellon University.

## Chapter 1. Define Project Success Criteria



*I've worked with some government agencies whose project funding is made available based on a state budgeting cycle. The project schedule is always presented as an imposed constraint: the project must be done by the end of the budget cycle. But people in these agencies tell me that if they don't complete the project by that time, they nearly always get more money and time to finish the project in the next budget cycle. So the schedule really isn't a constraint at all, just a desirable target date.*

“Begin with the end in mind” is Habit 2 from Stephen Covey’s *The 7 Habits of Highly Effective People* (Covey 1989). In Covey’s words, “To begin with the end in mind means to start with a clear understanding of your destination. It means to know where you’re going so that you better understand where you are now and so the steps you take are always in the right direction.” At the beginning of every software project, the stakeholders need to reach a common understanding of how they will determine whether this project is successful.

Consultant Tim Lister defines *success* as “meeting the set of all requirements and constraints held as expectations by key stakeholders.” If you don’t know early on how you’re going to measure your project’s business success, you’re headed for trouble. Defining explicit success criteria during the project’s inception keeps stakeholders focused on shared objectives and establishes targets for evaluating progress. For initiatives that involve multiple subprojects, success criteria help align all subprojects with the big picture. Ill-defined, unrealistic, or poorly communicated success criteria can lead to disappointing business outcomes. Vague objectives such as “world class” or “killer app” are meaningless unless you can measure whether you’ve achieved them.

To help you determine where your project is heading, this chapter describes the four-step process for defining project success criteria shown in Figure 1-1 (Wiegers 2002d). The deliverables from this chapter will help you make the myriad decisions that pop up during the life of a project.

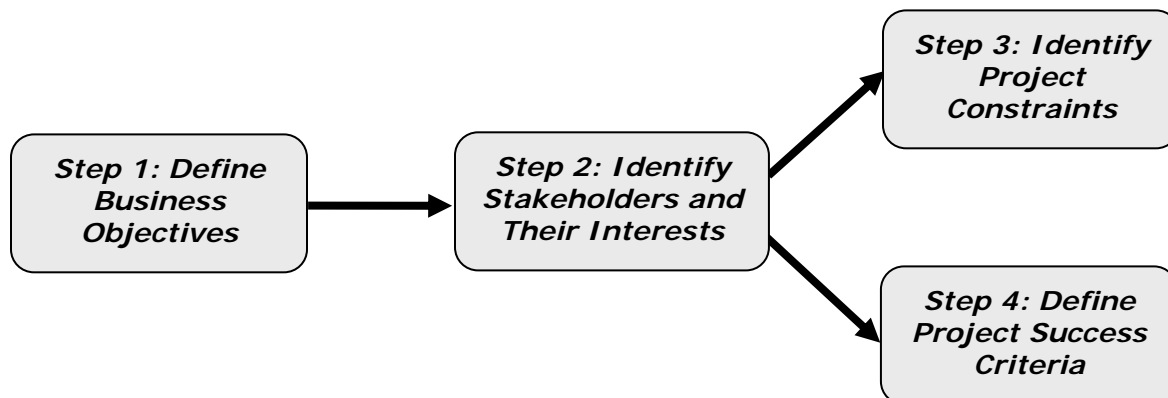


Figure 1-1: A process for defining success criteria

## Step 1. Define Business Objectives

Business *goals* answer the question “Why do we want to tackle this project?” You can transform each goal into specific, attainable, verifiable, and prioritized business *objectives* to help you assess whether you’ve achieved the goal. Express business goals and objectives in terms of measurable business, political, or perhaps even social value. For example, a goal might be to “Demonstrate competitive advantage with distinguishing features and technologies.” A corresponding business objective might state that specific features must be operational by a particular date. Another objective might require the team to demonstrate by a specific date that the planned technologies are sufficiently robust for use in your product (provided you can define what “robust” means).

A project that satisfies its requirements specification and ships on schedule and budget is good. A product that achieves its intended business objectives is much better. Well-crafted business objectives articulate the bottom-line outcomes the product will deliver for both internal and external stakeholders. They include the expected outcome, the time frame for achieving the objective, and how you will measure successful achievement (Wysocki and McGary 2003). Business objectives typically address:

- ◆ What the product<sup>1</sup> must be and do, including essential or distinguishing functions it will perform and how well it will perform them.
- ◆ Economic constraints, including development costs, cost of materials, and time to market.
- ◆ The product’s operational and temporal context, such as technologies required for compatibility in the target environment, backward compatibility, and future extensibility.

Try to write business objectives that are SMART: specific (not vague), measurable (not qualitative), attainable (not impossible), relevant (not unimportant), and time-based (not open-ended). Table 1-1 illustrates some simple business objective statements.

Record your business goals and objectives in a high-level strategic guiding document for the project. Such documents include a vision and scope document (see Chapter 2), project charter, project overview, business plan, business case, or marketing requirements document. Some software project management plan templates include a section on management objectives and priorities, which could contain your business objectives and stakeholder analysis. A sample project management plan template is provided with this handbook. This template includes slots in which to store much of the information you’ll develop if you complete the worksheets in this handbook. Other information might go into other project guiding documents, such as a quality assurance plan that contains product release criteria.

As the team gets into design and implementation, they might discover certain business objectives to be unattainable. The cost of materials might be higher than planned or cutting-edge technologies might not work as expected. Business realities also can change, along with evolving marketplace demands or reduced profit forecasts. Under such circumstances, you’ll need to modify your business objectives and reassess to see whether the project is still worth pursuing.



One telecommunications project had an objective to build a replacement interface unit for an existing product at a specified maximum unit cost and with a defined reusability goal. A review of the proposed architecture revealed that the product would exceed the unit cost and couldn’t meet the

---

<sup>1</sup> I’ll use the terms “product,” “application,” and “system” to refer to whatever sort of software or software-containing deliverable your team is producing. These practices apply equally well to developing commercial software products, embedded systems, information systems, Web sites, and so on.

reusability goal. Management revisited the business case and concluded that the unit cost was still low enough to justify development. They revised the profit forecasts, dropped the reusability goal, and forged ahead. The key stakeholders redefined “success” to match business and technical realities.

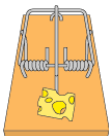
**Table 1-1: Examples of Financial and Nonfinancial Business Objectives**

<i>Financial</i>	<i>Nonfinancial</i>
<ul style="list-style-type: none"> <li>• Capture a market share of X% within Y months.</li> <li>• Increase market share in region X by Y% in Z months.</li> <li>• Reach a sales volume of X units or revenue of \$Y within Z months.</li> <li>• Achieve X% profit or return on investment within Y months.</li> <li>• Achieve positive cash flow on this product within Y months.</li> <li>• Save \$X per year by replacing a high-maintenance legacy system.</li> <li>• Keep unit materials cost below X dollars per unit in the expected Y-year lifetime of the product.</li> <li>• Reduce support costs by X% within Y months.</li> <li>• Receive no more than X service calls per unit and Y warranty calls per unit within Z months after shipping.</li> </ul>	<ul style="list-style-type: none"> <li>• Achieve a customer satisfaction measure of at least X within Y months of release.</li> <li>• Process at least X transactions per day with at least Y% accuracy.</li> <li>• Achieve X time to market that provides clear business advantages.</li> <li>• Develop a robust platform for a family of related products.</li> <li>• Develop specific core technology competencies in the organization, with competency measured in some way.</li> <li>• Be rated as the top product for reliability in published product reviews by a specified date.</li> <li>• Maintain staff turnover below X% through the end of the project.</li> <li>• Comply with a specific set of Federal and state regulations.</li> <li>• Reduce turnaround time to X hours on Y% of customer support calls.</li> </ul>

## Step 2. Identify Stakeholders and Their Interests

A project achieves success by delivering suitable value to various *stakeholders*—people or groups that are actively involved in a project, are affected by its outcome, or can influence its outcome (PMI 2000; Smith 2000). Begin your quest for success by identifying these stakeholders and what is important to them. “Value” could translate to time savings for a corporate department, market dominance for a commercial software vendor, or increased productivity for a user. Look for stakeholders both inside and outside the development organization, in the categories shown in Table 1-2. Stakeholders can be involved in a project in many different ways (McManus 2005)

Next, perform a stakeholder analysis to reveal the expectations each stakeholder group has for the project. Identify each stakeholder’s *interests*, or win conditions (Boehm and Ross 1989; Brackett 2001). Interests include financial benefits, specific time to market, required functionality, performance targets, usability, reliability, or other quality characteristics. Then evaluate how the pro-



**TRAP:** Overlooking important stakeholders, which means you’re simply lucky if they view the project as a success.

**Table 1-2: Some Internal and External Stakeholder Categories**

<i>Internal</i>	<i>External</i>
<ul style="list-style-type: none"> <li>• Project manager</li> <li>• Program manager</li> <li>• Product manager</li> <li>• Executive sponsor or funding authority</li> <li>• Project team members, including analysts, developers, testers, and technical writers</li> <li>• Quality assurance</li> <li>• Company owners or shareholders</li> <li>• Marketing</li> <li>• Manufacturing</li> <li>• Finance</li> <li>• Legal</li> <li>• Sales</li> <li>• Support</li> </ul>	<ul style="list-style-type: none"> <li>• Direct and indirect user classes</li> <li>• Procuring customers</li> <li>• Regulatory bodies</li> <li>• Auditors</li> <li>• Standards certification bodies</li> <li>• Government agencies</li> <li>• Subcontractors</li> <li>• Prime contractors</li> <li>• Venture capitalists</li> <li>• Business partners</li> <li>• Materials, information, and service suppliers</li> </ul>

ject will be affected if each interest is or is not satisfied. The project might not be able to fully satisfy everyone’s interests. This analysis will help you determine which interests are the most compelling. Figure 1-2 illustrates a simple template you can use to document essential stakeholder information.

Also assess the relative influence each stakeholder has on the project’s decisions. Some stakeholders might wield great political power. Others could dictate essential features, impose restricting constraints, or be a potential source of substantial revenue. *Key stakeholders* are those who can strongly influence the project’s decisions—and those whom it’s most important to satisfy.

Expect to encounter conflicting stakeholder interests. Finance might want the lowest possible unit cost for the product, although the development team wishes to use expensive, cutting-edge technologies to meet stringent performance demands. Different market segments will value different combinations of time to market, feature richness, and reliability. Identifying your key stakeholders will help you to resolve such conflicts as they arise and to negotiate win-win solutions that maximize benefits for the greatest number of stakeholders.

<b>Stakeholder</b>	<b>Major Benefits</b>	<b>Attitudes</b>	<b>Win Conditions</b>	<b>Constraints</b>
Executives	increased revenue	see product as an avenue to 25% increase in market share within 1 year	richer and more novel feature set than competitors have	maximum budget = \$1.4M
Manuscript Editors	fewer errors in their work	highly receptive, but unwilling to be retrained	automatic error correction capabilities; ease of use; high reliability	must run on existing low-end PCs

**Figure 1-2: Simple stakeholder analysis example**

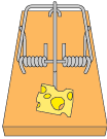


Some years ago, my small software team was building an information system for a large corporation's research division. The stakeholder who represented our primary user group naturally wanted us to address just his department's needs. However, we identified other departments as also being important stakeholders. This led us to include functionality that made the system valuable to additional user classes. The primary stakeholder wasn't thrilled about the extra time it took us to deliver "his" application. But the stakeholder analysis helped us make the right strategic choices to best leverage the company's investment in this project.

Because of the many project decisions that lie ahead, it's essential to determine your key decision makers and to define a decision-making process very early in the project. Groups that must make decisions need to select an appropriate *decision rule* (Gottesdiener 2001). Some possible decision rules include:

- ◆ Voting, majority rules
- ◆ Reaching unanimous agreement
- ◆ Achieving consensus
- ◆ Delegating the decision to a single individual
- ◆ Having the person in charge make the decision after collecting input from others

There is no single correct decision rule, but every group needs to select one before they confront their first significant decision. Participative decision making generates more buy-in to the outcome than does unilateral decision making, which often is done by people with insufficient information about the problem being addressed.



***TRAP: Failing to identify the most important stakeholders, thereby hampering effective decision making.***

### Step 3. Identify Project Constraints

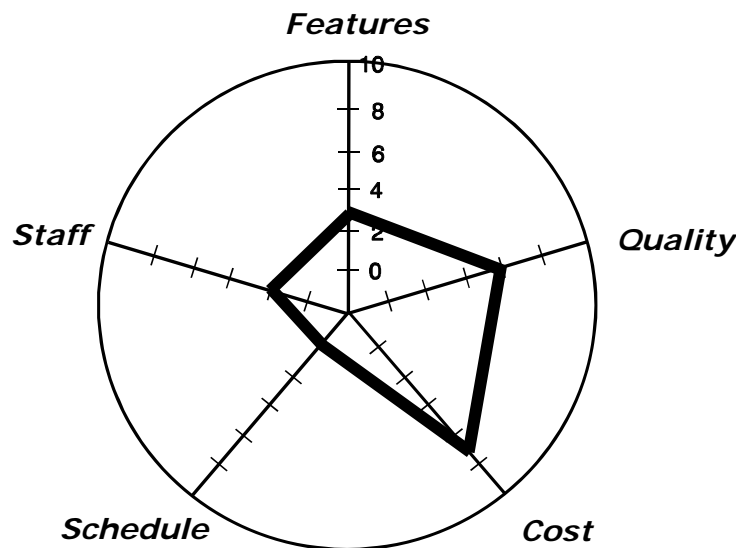
Perhaps you've seen a sign in your local automobile repair shop that asks, "What do you want: good, fast, or cheap? Pick two." People often attempt to apply this classic "iron triangle" of tradeoffs to software. But I've seen the triangle drawn in many different ways, with different assumptions made about constant project scope or constant quality. Jeffrey Voas (2001) argues that it's not realistic to expect faster, better, *and* cheaper software. It is possible to build software faster and cheaper. But it's not possible to get software faster and better: You can't accelerate creativity, and software development is a highly creative activity. High-quality deliverables take a bit longer to initial release, but they save you much time in the long run because you don't have to do extensive testing and rework to fix and maintain them. Nor is cheaper and better feasible. Software with stringent quality demands, such as safety-critical systems, is expensive because it's imperative to remove (or, even better, prevent) defects.

The tradeoffs are real, but the triangle representation is wrong. In fact, a project manager must make tradeoffs along five dimensions: features (or scope), quality, staff, cost, and schedule (Wiegers 1996). Some people add an additional dimension, risk. Each dimension fits in one of three categories:

- ◆ *Constraints* impose boundaries and restrictions within which the team must operate. For constraints, state the immovable limit.

- ◆ *Drivers* identify key project success goals. Drivers typically afford the project manager a bit of latitude, but they define important targets toward which the team must work.
- ◆ *Degrees of freedom* are factors the project manager can adjust within certain limits. For degrees of freedom, identify the allowable range within which the project manager must operate.

It's important to classify the five project dimensions into these three categories early in your project. One way to represent this information is with a *flexibility diagram*, illustrated in Figure 1-3 (Wiegiers 1996). A flexibility diagram is a Kiviatt diagram (also known as a spider chart or radar chart), in which the five normalized axes extend radially from a center point. A point is plotted for each dimension on a scale from 0 (a constraint) to 10 (complete flexibility). Figure 1-3 illustrates a flexibility diagram for a hypothetical shrink-wrap software package. The point plotted at zero on the schedule axis indicates that schedule is a constraint. Staff and features are drivers with low amounts of flexibility. Cost and quality are degrees of freedom that offer more flexibility for the project manager. Connecting the points on all five axes creates an irregularly-shaped pentagon. Jim Highsmith describes an alternative "tradeoff matrix" to represent which project dimensions are fixed (constraints), flexible (drivers), or can be adjusted (degree of freedoms) (Highsmith 2004).



**Figure 1-3: Flexibility diagram for a hypothetical commercial software product**

The flexibility diagram is a qualitative tool intended to help the key stakeholders discuss project constraints and success drivers. Don't try to integrate the area of the pentagon or perform similar rigorous analyses. However, the smaller the pentagon, the more constrained the project is and the lower the chance of it being fully successful.

Managers often think of schedule as being a constraint when it's really a driver. To tell the difference, consider the consequences of shipping late. If it would mean paying a contractual penalty or losing a unique market window forever, then schedule truly is a constraint. One early e-commerce project delivered a set of Christmas-themed baskets bundled with a digital camera and special software. Had those baskets and the associated Web site not been available in time for Christmas, the project would have been a complete failure. Schedule really was a constraint in this case. Schedule is

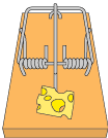


a driver when there is a strong business reason to ship by a specific date but catastrophe doesn't strike if the release is delayed by a few weeks.

With an understanding of a project's drivers and constraints, the project manager can make appropriate tradeoff decisions. For example, maybe your project has sufficient funding but a hiring freeze is in place. Perhaps you can hire contractors, outsource some work, or buy some automation tools. Every tradeoff comes with a price tag, though:

- ◆ Adding more people is costly and doesn't shorten the schedule proportionately (Brooks 1995).
- ◆ Increasing scope can degrade quality if the schedule isn't extended.
- ◆ Stringent quality demands increase the development and testing effort needed.
- ◆ Attempting to reduce cost by skimping on requirements development or product design increases long-term costs and actually delays releasing an acceptable product.

There's more bad news: Not all project dimensions can be constraints. Project managers *must* have some latitude to react to schedule slips, scope growth, staff turnover, and other eventualities. A successful manager adjusts the degrees of freedom so as to let the team achieve the project's success criteria—its drivers—within the limits imposed by the constraints.



***TRAP: Overconstrained projects that don't give the project manager enough flexibility to adjust to changing conditions.***

## Step 4. Derive Project Success Criteria

You can't judge whether a product satisfies all of its business objectives until some time after delivery. However, each business objective should imply technical success criteria you can monitor prior to release. The development team can't directly meet a business objective of "Capture a 40 percent market share within 12 months." However, they can decompose that objective into specific project actions and product features aligned with achieving the market share target. Disconnects between success criteria and business objectives give stakeholders no way to assess whether the project is likely to meet those objectives.

Success criteria shape many aspects of your project, beginning with the functional and non-functional requirements specifications. If the stakeholders understand the project's principal business objectives and success criteria, it's easier to make decisions about which proposed product features and characteristics are in scope and which are not.

Table 1-3 shows some simple examples of project success criteria. Well-written success criteria are feasible, quantitative, and verifiable.<sup>2</sup> For example, performance goals are often written against either internal benchmarks or external industry reference data. A goal might compare a new search engine's performance to that of the prior version and also to the performance of a competing product under some set of standard conditions. Such success criteria let the project team design tests that measure whether performance, reliability, or throughput goals are being met. Trends in these

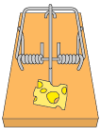
---

<sup>2</sup> Too often, success criteria are worded imprecisely and cannot be assessed objectively so it's difficult to judge if they've been satisfied. Tom Gilb has developed a flexible keyword notation called "Planguage" that permits precise statements of requirements and project goals (Gilb 2005). See Chapter 3 for a brief overview of Planguage and an example of how to use it to specify a product's release criteria.

Table 1-3: Examples of Project Success Criteria

- Total project cost does not exceed X% of the initial budget.
- The actual project duration is within X percent of the committed duration.
- All high-priority functionality defined in the requirements specification is delivered in the first release.
- The estimated number of residual defects does not exceed X per function point<sup>3</sup>.
- Load testing confirms successful scale-up to X concurrent users, with Web page download times no longer than Y seconds over a 56K dial-up connection.
- All unauthorized network intrusions are detected, intercepted, blocked, logged, and reported.
- The mean time to failure under specified load conditions is at least X hours.
- The average installation time of the product is less than X minutes, with Y% of installations being error-free and requiring no support call.
- Prerelease development rework does not exceed X percent of total development effort.

measures may provide early warning that you might miss a success target, so the team can either take corrective action or redefine the success criteria.



***TRAP: Unattainable, qualitative, or unverifiable success criteria. You'll never know if you're there.***

Not all of these success criteria can be top priority. You'll have to make thoughtful tradeoff decisions to ensure that you satisfy your most important business priorities. Without clear priorities, team members can work at cross-purposes, which leads to rework, frustration, and stress. Weight your success criteria based on how strongly they align with achieving the critical business objectives, so team members will know which ones are essential and which are merely desirable. In one weighting scheme, stakeholders allocate 100 points to the success criteria associated with each business objective, with the more important items receiving more points.

Consider summarizing your success information in the form illustrated in Figure 1-4. List each business objective, the stakeholder who provided it, corresponding project success criteria, and each criterion's method of measurement and priority weight. Use the stakeholder list to verify that you haven't overlooked any important business objectives. You don't need to associate all stakeholders with every business objective, but every objective should be important to some stakeholder.

If you clearly define what success means at the beginning of your project, you greatly increase the chances of achieving it at the end. Understanding your stakeholders' interests, writing clear business objectives, and defining corresponding success criteria lay the foundation for a happy outcome.

---

<sup>3</sup> Function points are an implementation-independent measure of software size (IFPUG 2002).

<i>Business Objective</i>	<i>Stakeholder</i>	<i>Project Success Criteria</i>	<i>Measurement</i>	<i>Weight</i>
Achieve a customer satisfaction measure of at least 4 out of 5 within four months after release.	Marketing	Human factors approves user interface design through usability review.	All major severity UI design defects are corrected.	20
		UI prototype evaluation with focus group results in at least a 4.2/5.0 rating and will enable 90% of high-priority use cases to be performed.	Survey of focus group members; count of defined high-priority use cases.	30
		Product passes acceptance criteria defined by 80% of beta site evaluators.	Pass/fail rating by beta sites.	50

Figure 1-4: Sample Success Criteria Table

## Cross-References

**Define Product Vision and Project Scope (Chapter 2):** The vision describes a solution that should satisfy the project business objectives. The scope defines the subset of that ultimate vision that the current project will address within the bounds imposed by project constraints.

**Define Product Release Criteria (Chapter 3):** Align your product's release criteria with the project's success criteria.

**PMBOK 2.2:** This section of the Project Management Body of Knowledge addresses project stakeholders (PMI 2000).

**PMBOK 6.4.1.6:** Two types of time constraints are discussed here: imposed dates, and key events or major milestones (PMI 2000).

## Practice Activities

1. Complete Worksheet 1-1.
2. Complete Worksheet 1-2.
3. Select an appropriate decision rule and decision-making process that your stakeholders can follow to make strategic project decisions.
4. Complete Worksheet 1-3.
5. Complete Worksheet 1-4.
6. Identify stakeholders having conflicting success criteria. Based on your stakeholder analysis and your decision-making process, resolve these conflicts.
7. Based on the results of the above practice activities, think about what actions you might take if your project environment undergoes some significant change, such as those on Worksheet 1-5.

## Worksheet 1-1: Your Project's Business Objectives

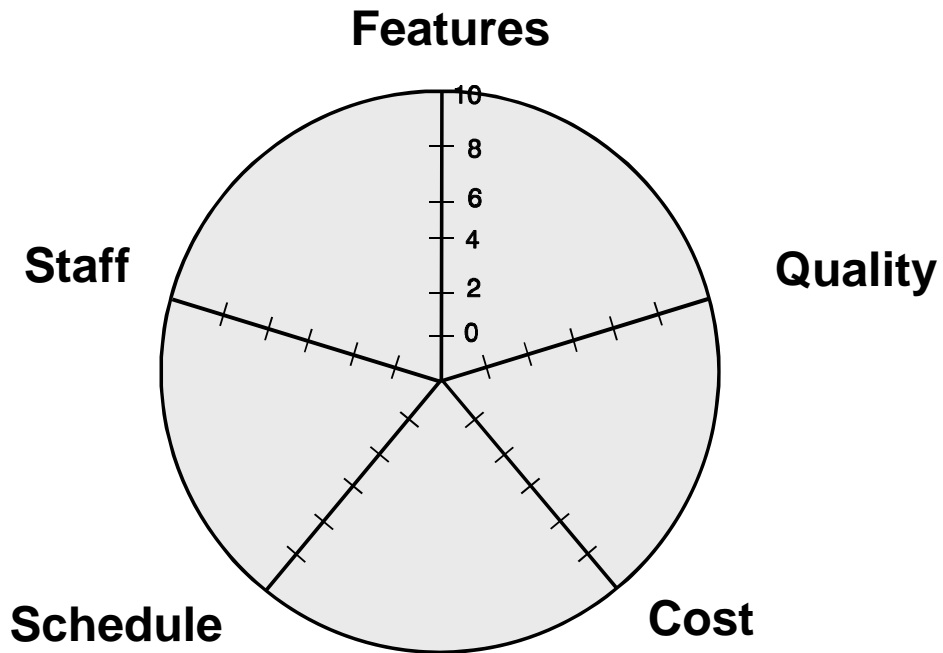
State several of your project's business objectives. Remember to make them SMART (specific, measurable, attainable, relevant, time-based).

ID	Business Objective
OB-1	
OB-2	
OB-3	
OB-4	
OB-5	
OB-6	
OB-7	
OB-8	



## Worksheet 1-3: Your Project's Five Dimensions

Define your project's constraints, drivers, and degrees of freedom. Draw a flexibility diagram like that in Figure 1-3.



Dimension	Constraint (state limits)	Driver (state goals)	Degree of Freedom (state range)
Features			
Quality			
Cost			
Schedule			
Staff			



## Worksheet 1-5: Responding to Change

Based on your business objectives, success criteria, constraints, and drivers, what actions might you take if the following events occur? Alternatively, identify incorrect assumptions or modified constraints that would require you to change the project's direction. How might you respond?

**A competitor comes out with features you want to match?**

**Marketing moves the delivery date up 1 month?**

**Half the team quits to form a start-up?**

**New government regulations force changes in your requirements?**

**Your customer wants to add many new features?**

**Other changes specific to your project or application domain (describe)?**