CHAPTER 4

# DELIVERY

## The Big Ideas

- By understanding the uncertainty and complexity characteristics of your projects, you can identify better ways to lead those projects.
- High complexity or uncertainty correlates to higher risk. Reduce these factors, and you reduce your level of risk. Project decomposition can reduce complexity, while incremental delivery helps lead a project through uncertainty.
- Some leaders are natural managers of complexity, while others are experts at uncertainty. Match leadership styles to project characteristics, and develop leaders' skills to broaden their capabilities.

## Case Study: The Swiss Stock Exchange

The Swiss stock exchange was preparing to enter electronic trading. There had been two prior attempts to develop the software solutions. The first attempt was an internal development effort, and it failed. In the second attempt, the project was outsourced to a prominent consulting firm; it failed as well. In both of these prior failures, everyone thought that they were doing things by the book using software development best practices.

The third attempt was getting under way and there were real concerns about the prospects for yet another failure. Pollyanna got involved on the team and eventually ended up leading the project. The challenges were significant. The team was over 120 people, and 60% of those were contractors. There were multiple customers, as every Swiss bank (approximately 50 in total) had financed the project, and each bank had different requirements and desires. The existing systems were 20 years old and

upgrades were drastically needed. Clearly, this was a very high-profile project with a lot of money at stake.

On the positive side, the project attracted the best talent and was able to specify its delivery date within a reasonable time frame. The 2-second throughput requirement, total software and hardware redundancy, and "lights out" maintenance model all seemed doable. A retired trader was heading up the functional specification team—a sort of on-site "customer." Things looked better than before.

But there were other ominous signs that history would repeat itself. The team in charge of building one of the key server-side components had developed a two-year plan detailing tasks down to the level of every 15 minutes. What were they thinking? Creation of such a schedule is over-planning even for a project with almost no uncertainty. The team leaders had fallen into the trap of the illusion of control—even though they actually had almost no control. Instead, they spent an inordinate amount of time updating the detailed plan and trying to justify how they would get back on schedule.

Pollyanna had the team members move to a system that would allow them to update and reprioritize the features and the work tasks on a weekly basis. The team leaders reviewed progress at the end of each week and then, based on this information reprioritized the next week's work. Every six weeks, the teams re-estimated and re-planned tasks for the next three-month iterative release. Change was inevitable and was managed through the reprioritization process. All team leaders made the decisions together.

Next, Pollyanna worked with the trading system team to build the workstation where the traders would make their trades and gather information. The trading system was associated with a lot more uncertainty because it was brand new and would be more intimately connected to the end users. To get it right, the developers formed a user group of traders and built prototypes to obtain useful feedback. It took four prototypes to get a system that the users liked—but what really mattered is that the users were happy with the final delivery.

Together, this team of teams successfully delivered the first fully computerized stock exchange system integrating trading, clearing, settlement, and member back-offices.

## All Projects Are Not Created Equal

Leaders are constantly hoping to find the magic project management approach that will guarantee successful delivery of their projects. But what any experienced project manager knows is that all projects are not created

equal; as a consequence, no single approach can possibly be appropriate for all projects. Certainly, many good techniques and practices are applicable to a large collection of projects, but the idea of running projects "by the book" is a mirage. While all projects are different, certain patterns can be discerned among projects with similar characteristics. For those projects with similar characteristics, certain approaches to leadership and governance work especially well.

The two primary characteristics that influence and drive project management characteristics are complexity and uncertainty. Complexity is a measure of the size of the project, the dependencies of the project, and the nature of the project team. Uncertainty is an indicator of what is known and what is not known about the project; it can include shifts in either the customer needs or the technological implementation. Relatively simple projects with low uncertainty are quite different from highly complex projects with high uncertainty and should be led and managed accordingly.

The $2 \times 2$ matrix in Figure 4.1 shows the context leadership model for using project complexity and uncertainty to help project leaders guide and govern their projects. Each quadrant is explained in more detail in the sections that follow.

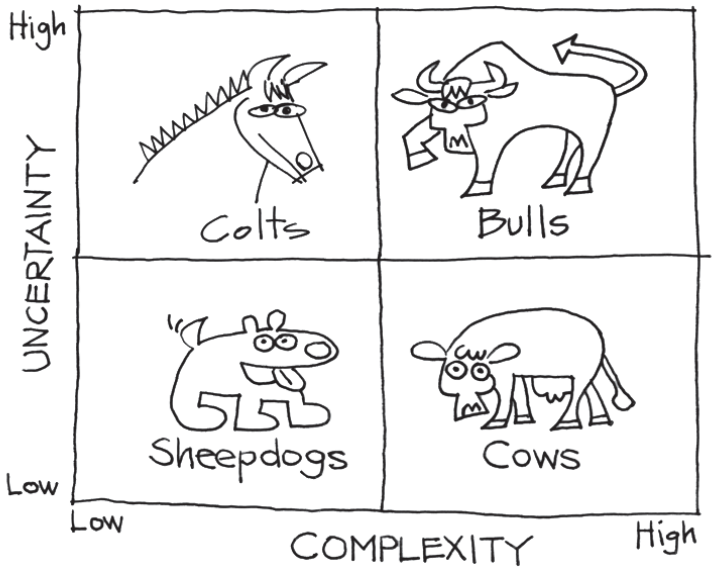


**FIGURE 4.1** Context leadership model

## Sheepdogs: Simple Projects with Low Uncertainty

The easiest and most controllable projects are the simple projects being developed by small teams. All projects have some degree of complexity and uncertainty, but projects categorized in the sheepdog quadrant have the lowest degree of complexity and uncertainty. The idea of the sheepdog is meant to be seen in a positive light: With care and feeding, the sheepdog is very productive. With these types of projects, the best thing to do is to make sure the team knows what is needed and to then stand back and let the team members do their jobs to ship the result. For those projects that have some uncertainty, it is good practice to keep their duration short or to deliver the results in incremental iterations to limit the uncertainty's impact. Prototype or skunkworks projects can also fit in this category. For projects in this quadrant, additional process ceremony and documentation are unnecessary and inefficient, so run them using only the minimal core set of practices used for all projects in all quadrants.

## Colts: Simple Projects with High Uncertainty

New products and initiatives will often have both market and technical uncertainty. If you keep the team small and its members close together, they can react quickly to adapt to those uncertainties. The metaphor of the young colt aptly describes these projects. Colt projects are just getting started and have a lot of energy and freedom. Software projects in these categories are excellent candidates for applying prototyping and lightweight agile development techniques such as Scrum, Extreme Programming, or Crystal. We do not imply that agile practices are applicable only to colts, but rather suggest that colts are particularly well suited to agile approaches.

Because these projects have little complexity, process ceremony and documentation should be kept to the minimal set needed by the team to be effective. The real focus of the team and its leadership needs to be on navigating and managing the uncertainties, with continual and rapid feedback being a key to project success. This type of effort usually requires a leader who has a strong connection to the source of the uncertainty. If the source of the uncertainty lies in the market or the business, then the leader will need to have the appropriate skills and background to deal with those uncertainties. If the uncertainty is primarily technical in nature, then it is a good idea to have a technical leader for the project. In many cases, both technical and market sophistication are needed; if these capabilities are not available in a single leader, the team might need two leaders who will collaborate closely.

## Cows: Complex Projects with Low Uncertainty

The mature systems and product suites that are important to the business will continue to warrant the services of large project teams and are usually the organization's cash cows. In addition to the obvious connection, the cow is a good metaphor for these projects because cows are quite large but don't move particularly fast. Cow projects have less need for agile steering; in fact, they might need disciplined change control on more rigorous requirements or specifications to reduce their impact when many projects or customers depend on them. Projects in this quadrant might aim to be agile enough to respond to some uncertainties, but they need defined and published interfaces to the projects that depend on them.

Cow projects also require more direct project and program management, including looking at issues such as cross-team communication and critical paths. It is common practice for a cow to be an integration project involving a number of smaller projects (typically sheepdogs). Successful leadership of cow projects requires strong skills in working with people and teams and the ability to coordinate the activities of those people and teams.

## Bulls: Complex Projects with High Uncertainty

Projects that are highly complex and have high uncertainty are particularly challenging. Because of the high level of uncertainty, they need to embrace change through iterative feedback to be agile enough to navigate the uncertainty. To run successfully, they require much of the agile steering required for colts, yet also require much of the same process ceremony we use with cows. Communication channels for these projects must be very efficient.

The bull metaphor is quite appropriate for such projects. Bull projects are large and can get out of control quickly if the team isn't careful. They have high visibility throughout the organization, because they often deal with emerging products or initiatives that require significant investment. Often they are next-generation products or solutions that have great hopes to supplant existing cash cows. Expectations are high, yet uncertainty and complexity are equally high.

Leading a bull project is neither easy nor a task for the faint of heart. It is critical that bull projects have the best and most seasoned leaders who understand how to work with agility while cutting through complexity, balancing the dichotomy. These leaders need to have the ability to understand the business and the technology so as to manage the uncertainty, while at the same time being experts at project mechanics and at dealing with the complexities of coordinating people, teams, and across-team activities.

Most organizations have only a few leaders with the requisite capacity to lead such projects. It's unwise for an organization to have more bull projects than project managers who are capable of running bull projects.

## Assessing Project Uncertainty and Complexity

We have introduced this model into a number of organizations. In each case, the organization has ultimately tailored the model to better meet its needs. Some organizations have taken the base model and tweaked it just a bit to fit their needs. Others have bypassed any scoring for the assessment and gotten value out of the model simply by assessing complexity and uncertainty through intuition. At the other extreme, organizations that have a history of detailed process have taken the overall concept and created what we would consider to be a rather complicated scoring model to assess complexity and uncertainty (20–30 attributes for each).

Use whatever scheme works for you. In the base scoring model, we suggest using 1 for low complexity/uncertainty, 3 for medium complexity/uncertainty, and 9 for high complexity/uncertainty. We then suggest using a simple arithmetic average or, if you are feeling more ambitious, a weighted average. Projects scoring greater than about 4 are considered "high complexity/uncertainty."

### Complexity Drivers

The project's structure determines its complexity, which affects the ease or difficulty the team has navigating through the project. Higher complexity implies the need for more structured communication and documentation. In contrast, lower-complexity projects can often thrive on informal communication channels. The parameters and scoring model for the base model are summarized in Table 4.1 and explained in more detail in the sections that follow.

#### *Team Size*

A large team generally implies a complex project. Although situations do arise in which portions of the work are decoupled, with many projects the need for cross-project communication is critical. The number of potential communications interactions goes up exponentially as more members are added to a

**Table 4.1** Complexity Attributes

| Attribute | Low Complexity (1) | Medium Complexity (3) | High Complexity (9) |
|---|---|---|---|
| Team size | 2 | 15 | 100 |
| Mission critical | Speculative | Established market | Safety critical or significant monetary exposure |
| Team location | Same room | Within same building | Multisite, worldwide |
| Team maturity | Established team of experts | Mixed team of experts and novices | New team of mostly novices |
| Domain knowledge gaps | Developers know the domain as well as expert users | Developers require some domain assistance | Developers have no idea about the domain |
| Dependencies | No dependencies | Some dependencies | Tight integration with several projects |

team. If a team must be large, it is wise to consider subdividing the team at loosely coupled interfaces to allow the subteams to act efficiently.

### *Mission and Safety Criticality*

If the project puts lives or business-critical functions at risk, we must treat it differently than if the only cost of failure is the project investment. A project with a higher element of criticality will mean greater visibility or exposure for the organization. As such, it requires a more well-defined process for tracking and managing the project and the risks, which ultimately adds to the complexity of the project.

### *Team Location*

Having everyone in the same room enables high-bandwidth communication among the project team members. A widely distributed team, or one in which a significant portion of the team is located several time zones apart, can significantly increase project complexity. Team location can be a difficult attribute to assess, because use of a team that has one or a few

dispersed members may or may not drastically increase the complexity of project. We've advised teams to use their judgment on this assessment.

#### *Team Maturity*

An established team of experts who have been working together for years on product-line enhancements can almost anticipate what team members are likely to need and do. This kind of "mind reading" contrasts with the situation faced by a brand-new team of relative novices. The latter team requires far more hand holding and guidance, usually in the form of more formalized documentation of requirements and specifications.

#### *Domain Knowledge Gaps*

Many development projects have complex business processes that must be understood at a certain level by everyone on the team. At a minimum, it is critical that the product team have full-time access to the domain specialists to resolve ambiguities and produce the desired product. We've found that this process is greatly simplified when the developers are domain specialists themselves but becomes much more complex when access to domain knowledge is limited.

#### *Dependencies*

In general, the more dependencies that a project has on other projects or third parties, the more complex the project will be. It is critical to manage those dependencies and to track the activities of others so as to create alignment. Sometimes, however, an established third-party dependency does not add greatly to complexity if the team has a consistent track record of working with stable interfaces.

### Uncertainty Drivers

The needs of the customers and the choice of technology are the two major drivers of uncertainty for a project. Higher uncertainty implies the need for a means of absorbing changes and adapting to those changes, with the idea of getting to the ideal solution at the end of the project, though it may not necessarily be the solution that was envisioned at the beginning of the project. The parameters and scoring of the base model are explained in the following sections and summarized in Table 4.2.

**Table 4.2** Uncertainty Attributes

| Attribute | Low Uncertainty (1) | Medium Uncertainty (3) | High Uncertainty (9) |
|---|---|---|---|
| Market uncertainty | Known deliverable, possibly defined contractual obligation | Initial guess of market target is likely to require steering | New market that is unknown and untested |
| Technical uncertainty | Enhancements to existing architecture | We're not quite sure if we know how to build it | New technology, new architecture; some research may be required |
| Number of customers | Internal customer or one well-defined customer | Multiple internal or small number of defined customers | Shrink-wrapped software |
| Project duration | 0–3 months | 3–12 months | >12 months |
| Approach to change | Significant control over change | Moderate control over change | Embrace or create change |

### *Market Uncertainty*

If the market or customer needs are well known, the project probably won't need much steering. Conversely, if the customer needs aren't well understood and can be discovered only during the development of the solution, the ability to steer the project to the discovered goal—rather than to the initially stated objective—will be critical.

### *Technical Uncertainty*

Mature products using proven technology don't involve much technical uncertainty. Sometimes, however, a project may experience uncertainty while rolling out proven technologies that are new to the organization or to the team. By comparison, project teams building new products often want to use the latest technology, so these projects will have a high degree of technical uncertainty.

### *Number of Customers*

One primary amplifier of market uncertainty relates to the number of customers. A project in which there is one customer who is internal to the organization is quite different from a project in which there are multiple customers with multiple voices. Two issues arise with this attribute. First is the addition of multiple voices, which creates the potential for conflicting needs and gives rise to one form of uncertainty. Second is the inevitable difference between an internal customer and an external customer, particularly if the external customer is a market and not an explicitly identifiable customer. Many markets are fickle and fraught with uncertainty.

### *Project Duration*

Niels Bohr and Yogi Berra are both quoted as saying, "It is hard to make predictions, especially about the future." The further out the future is, the greater the chance for technical or market uncertainty to affect it. Iterative and incremental deliveries can play a big role in minimizing the risk of uncertainty in projects with a long duration.

### *Approach to Change*

The approach that the team takes toward managing change indicates how much flexibility the members have in managing uncertainty. For example, some components may be used by a number of other projects. This level of dependency can limit the amount of steering that the other projects can tolerate. While there may be market or technical uncertainty that would suggest a need to absorb change, continually modifying interfaces may not be acceptable when those changes affect other projects.

## Case Study: Integrating Software by Integrating People

By 1996, Landmark Graphics was already a leading provider of software applications in oil and gas exploration. It had grown to this enviable position from a start-up just 15 years earlier. During this time Landmark had expanded its operations via acquisition, resulting in a collection of corporate cultures separated by prior organization, geography, product line, and business domain. At the time, the company had six primary development centers in Houston, Austin, Denver, Tulsa, Calgary, and Aberdeen.

In most cases, the software acquired via acquisition was already the market leader. While providing strong technical applications was seen as

valuable, the real value proposition to Landmark's customers would come from providing integrated solutions that would substantially improve customer workflows. The company's means of differentiation and purpose were clear. However, each product group had been operating largely independently. There had been some integration activity in the past, but a comprehensive effort would be required to really make integration work.

The overall integration project's complexity attributes are listed in Table 4.3 and the uncertainty attributes in Table 4.4. This effort was clearly a bull program, as it had both high complexity and high uncertainty.

**Table 4.3** Complexity Attributes

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Team size | More than 200 | 9 | |
| Mission critical | Bet the company | 9 | |
| Team location | Distributed worldwide | 9 | |
| Team maturity | Established team of experts within their own products | 3 | |
| Domain gaps | Some gaps at points of integration | 3 | |
| Dependencies | Major dependencies | 9 | |

**Table 4.4** Uncertainty Attributes

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Market uncertainty | The market was known, but integration at this level had not been done before | 3 | |
| Technical uncertainty | The technology was known, but integration at this level had not been done before | 3 | |
| Number of customers | We were aiming to change the market | 9 | |

*(continues)*

**Table 4.4** Uncertainty Attributes

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Project duration | Approximately18 months | 9 | |
| Change | Integration was the focus; change was inevitable | 3 | |

## What the Leaders Did

Several things were done in response to the challenge. First, the CEO of the company and the entire leadership team made it clear that the company's differentiation came from its integrated solutions. They made sure that this mission statement was not just lip service. They made it clear what they wanted and why it was important. Everyone in the company knew what the number one focus was and why it mattered to the company and to its customers. The leadership also reiterated that message on a regular basic. Once the company's focus was clear, a coordinating team was created with the full-time responsibility for bringing the teams together and aligned to the common objective. A senior leader had responsibility for making things happen.

The senior leader recognized that the best way to integrate the software products would be by making sure that the teams and people were better integrated. He took advantage of existing structures of informal communication, such as a weekly all-hands gathering called Friday@4, where people got together for food and drink and checked in on what was happening in the rest of the organization. He also instituted a set of quarterly face-to-face meetings for all the project and product managers and key technical staff. These face-to-face meetings were critical to the success of the integration project. In addition to creating a checkpoint for ensuring that individual projects were on track and providing a venue for sharing learning across the organization, they provided a great forum for getting teams integrated on a social level. These events were always planned to last at least two days and were arranged so that nearly everyone traveled to attend them. As a result, at least one evening was available for the team leaders to socialize on a more informal basis.

The project was an enormous success. Market share grew significantly after the release and solidified Landmark's leadership position as the

provider of integrated solutions in oil and gas exploration. In addition, the social bonds continued to flourish and paved the way for further gains in integration in the future.

## Case Study: Time Is on Our Side

In 1998, a software company was facing the Y2K issue. The software that it was producing was mission critical to its customers. The firm needed to protect its customers by ensuring that its software did not have any issues with two-digit years and would not create any work stoppages or incorrect results following the turn of century. Fortunately, the company was able to build on the infrastructure it had put in place to deliver its prior integrated release. However, the new project was a much different program because the focus was on Y2K and Y2K only.

The overall project's complexity attributes are listed in Table 4.5 and the uncertainty attributes in Table 4.6. This overall integration effort was clearly a cow program, with high complexity but low uncertainty.

**Table 4.5** Complexity Attributes

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Team size | More than 200 | 9 | |
| Mission critical | Products are mission critical | 9 | |
| Team location | Distributed worldwide | 9 | |
| Team maturity | Established team of experts within their own products | 3 | |
| Domain gaps | Some gaps at points of integration | 3 | |
| Dependencies | Major dependencies | 9 | |

**Table 4.6** Uncertainty Attributes

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Market uncertainty | Only Y2K | 1 | ■ |
| Technical uncertainty | Only Y2K | 1 | ■ |
| Number of customers | Many customers, but only one well-defined requirement | 3 | ■■■ |
| Project duration | Approximately 6 months | 3 | ■■■ |
| Change | Eliminate any change other than Y2K issues | 1 | ■ |

### What the Leaders Did

This approach to guiding the program worked very well to accomplish the objective. The approach to change was critical to keeping the teams focused on Y2K, and only on Y2K. Teams were tempted to add other functionality, but leadership made it clear that going off on these tangents was not an option. The result: All products were delivered on schedule and the overall timeline was kept short. The objective was accomplished by the end of 1998 and the customer response was very positive.

## Case Study: The Swiss Stock Exchange Revisited

At the beginning of the chapter, we related the story of the Swiss stock exchange. One of the first steps that Pollyanna took when she started leading the group was to look at what the teams were doing. The overall project's complexity attributes are listed in Table 4.7 and the uncertainty attributes in Table 4.8. The project was clearly a bull, with both high complexity and high uncertainty.

As mentioned earlier, Pollyanna took a different approach working with the server side than she did when working with the client side.

**Table 4.7** Complexity Attributes

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Team size | More than 100 | 9 | |
| Mission critical | Bet the company | 9 | |
| Team location | All in same building; most in the same room | 1 | |
| Team maturity | New team of top talent | 3 | |
| Domain gaps | Team knew what the product needed to deliver | 3 | |
| Dependencies | Moderate dependencies | 3 | |

**Table 4.8** Uncertainty Attributes

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Market uncertainty | The market was generally known but new | 3 | |
| Technical uncertainty | New technologies | 9 | |
| Number of customers | Approximately50 defined customers | 3 | |
| Project duration | Approximately 2 years | 9 | |
| Change | Some change was expected and allowed | 3 | |

This case study exemplifies how a bull project can be decomposed into component projects that can then be managed semi-independently. Each of the subprojects for the Swiss stock exchange was run in a different manner, yet coordinated within a structure that supported the overall project.

Tables 4.9 and 4.10 show the complexity and uncertainty attributes of the client and server sides of this project. As you can see from the profile, the server-side project was more like a cow, whereas the client-side project was more like a colt. The overall project was still a bull, but each subteam was able to operate in the mode that was best suited for its needs.

Splitting projects into subprojects is a good practice. Many people suggest an ideal subteam size of 5–10 members, and our experience validates this suggestion. In addition to following this guideline, we find it equally important to design project teams at natural boundaries. In the software development world, the design mantra is to partition software such that it has loose coupling and strong cohesion. This mantra is just as applicable to project partitioning. Aim to partition projects in such a manner that they have only loose coupling; in other words, make sure that they are not intricately intertwined with or heavily dependent on other projects. Some dependencies will likely arise, but it is best if the teams can be as independent as possible. Likewise, it is best if there is strong cohesion within the project team. Put simply, we want the subteam working on the same basic aspect of the system.

**Table 4.9** Complexity Attributes for Server and Client

| | Server | | Client | |
|---|---|---|---|---|
| **Attribute** | **Score** | **Graph** | **Score** | **Graph** |
| Team size | 9 | ■■■■■■■■■ | 3 | ■■■ |
| Mission critical | 9 | ■■■■■■■■■ | 9 | ■■■■■■■■■ |
| Team location | 1 | ■ | 1 | ■ |
| Team maturity | 3 | ■■■ | 3 | ■■■ |
| Domain gaps | 3 | ■■■ | 3 | ■■■ |
| Dependencies | 3 | ■■■ | 1 | ■ |

**Table 4.10** Uncertainty Attributes for Server and Client

| | Server | | Client | |
|---|---|---|---|---|
| **Attribute** | **Score** | **Graph** | **Score** | **Graph** |
| Market uncertainty | 3 | | 9 | |
| Technical uncertainty | 3 | | 9 | |
| Number of customers | 3 | | 3 | |
| Project duration | 3 | | 3 | |
| Change | 3 | | 9 | |

Figure 4.2 shows the partitioning of the client from the server for the Swiss stock exchange and the overall coordination of the full solution. Although some loose coupling was needed from the client team to the server team, most of the focus of the teams' efforts was internal to the needs of the teams' deliverables.

## Using the Assessment to Reduce Risk

Decomposing larger projects into subprojects—the path taken with the Swiss stock exchange—can be a great way to help reduce complexity. In general, any degree of incremental complexity or uncertainty correlates with an incremental risk. Your teams might discover during the assessment that their projects are either more complex or uncertain than necessary to obtain their objective. Sometimes, it is possible to adjust one or more of the project attributes to reduce either complexity or uncertainty and in the process reduce the overall level of risk.

Three approaches can be taken with the risk: reduce it, mitigate it, or accept it. Use the first pass of the assessment to identify any opportunities for reducing or mitigating the risk. This decision making must be balanced

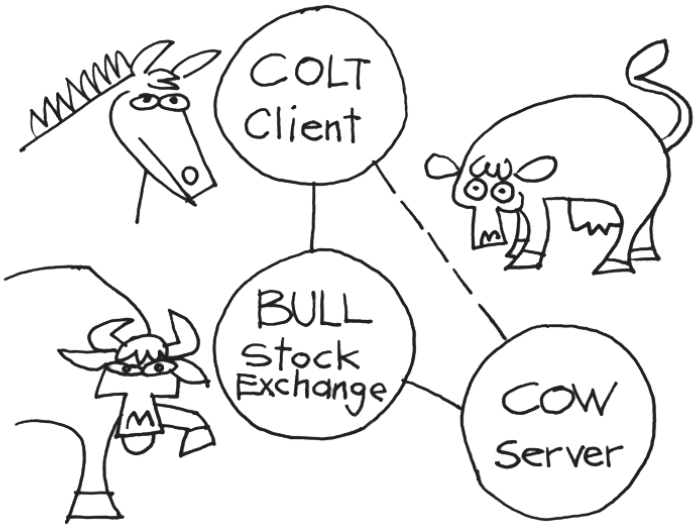


**FIGURE 4.2** Turning a bull into a cow and a colt

by the associated ramifications. Perhaps accepting the risk is preferable if it enables the team to maximize the value potential of the project. Tables 4.11 and 4.12 outline some techniques for reducing or mitigating risk from complexity and uncertainty.

**Table 4.11** Reducing or Mitigating Complexity Attributes

| Attribute | Ways to Lower the Attribute and Reduce Risk | Process Steps to Mitigate |
|---|---|---|
| Team size | Split teams into smaller cohesive groups. | Make sure teams have shared understanding of their purpose and the overall project success criteria. Bring teams together at regular intervals. Define, communicate, test, and manage project interfaces. |
| Mission critical | Not easy to reduce. | Make critical decisions and overall project status visible to all stakeholders. Ensure that stakeholders understand the consequences of key decisions. |
| Team location | Collocate the team if possible | Bring team members into face-to-face contact often. Invest in high-bandwidth communication and collaboration tools. |

**Table 4.11** Reducing or Mitigating Complexity Attributes

| Attribute | Ways to Lower the Attribute and Reduce Risk | Process Steps to Mitigate |
|---|---|---|
| Team maturity | Keep experienced teams whole, and leverage them from one release to the next. Integrate new members into the team early. | Make sure that time is allocated for mentoring of new team members, and invest in training and improvement for the entire team. |
| Domain gaps | Staff the team with members who have strong domain knowledge and use them to mentor other team members. Ensure that customer needs are constantly represented. | Educate and expose team members to the domain. Have team members sit with users and experience how they use the product. |
| Dependencies | Eliminate dependencies or work with static versions of dependencies. Build automated tests to check dependencies. | Invest in communication with teams that you are dependent on. Monitor their progress and be clear about your needs. |

**Table 4.12** Reducing or Mitigating Uncertainty Attributes

| Attribute | Ways to Lower the Attribute and Reduce Risk | Process Steps to Mitigate |
|---|---|---|
| Market uncertainty | Target a specific market segment that is better understood. | Deliver iteratively, utilize prototypes, and elicit customer feedback on a regular basis. |
| Technical uncertainty | Accept proven technologies. Design flexibility into situations to enable decisions to be made in the future. | Delay decisions where the uncertainty will resolve itself. Conduct experiments that will provide information to help resolve the uncertainty. |
| Number of customers | Target a specific customer segment or group of customers. | Use a product champion to solicit multiple customer voices and move them in a unified direction. Use the Purpose Alignment Model as a filter. |

*(continues)*

**Table 4.12** Reducing or Mitigating Uncertainty Attributes

| Attribute | Ways to Lower the Attribute and Reduce Risk | Process Steps to Mitigate |
|---|---|---|
| Project duration | Shorten the duration or deliver functionality in incremental releases. | Deliver incrementally and maintain high quality throughout the project. |
| Change | Exert control over change where it has the biggest impact. Delay decisions so that changes can be made without major impact. | Utilize incremental delivery and feedback to enable change to be absorbed into the project. Avoid committing to too much detail early. |

## Product Life Cycle

Products and long-term initiatives tend to have a life cycle that moves through the four quadrants illustrated in Figure 4.3. In our experience, many successful products follow path A: They start with low complexity and moderate uncertainty as skunks, move to greater uncertainty and with a bit more complexity as colts, and then become successful and turn into highly uncertain and highly complex bulls. Over time, the uncertainty dies down and the product becomes a cow. Eventually the complexity is reduced and the project becomes a sheepdog.

Another group of products follow path B and, as a result, never become particularly complex. There's nothing wrong with this route, as these products often end up being profitable, right-sized sheepdogs.

While we have seen numerous attempts to start products on path C, in which the projects are launched directly in the bull quadrant, we have seen very limited success with this approach. Our most successful bull projects have first begun as colts or sheepdogs, then evolved into bulls over time. When a product begins its life cycle as a bull, the combined risk of high complexity and high uncertainty with a new product and a new team is typically just too much to overcome. Certainly there are examples of bull project teams that have done wonderful things and succeeded, but in our experience they are the exception rather than the rule.

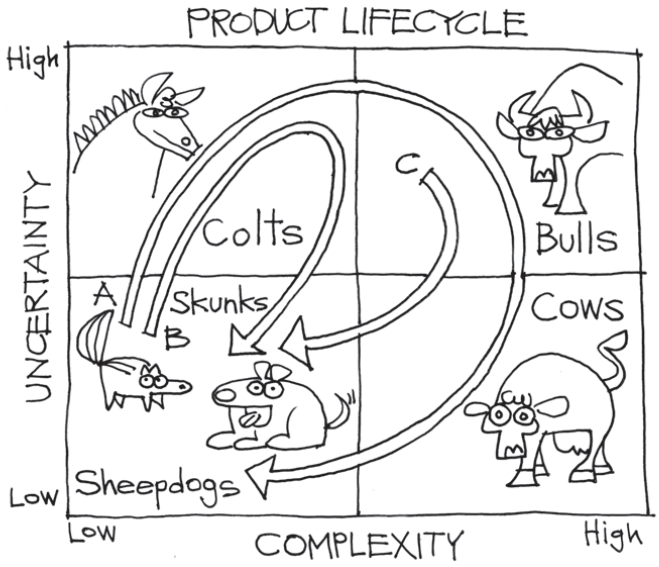


**FIGURE 4.3** Product life cycle

## Case Study: A Lot of Bull

A software company's products had been very successful, but many of its applications were 10 to 20 years old. The development teams had done their best to continually update the application suite, but the architecture was recognized to have limitations. A team of experts was created to embark on a quest to build the next-generation architecture and the initial product solutions. The original idea was to keep the team small, consisting of approximately 15 people.

Within a short time, the team realized the scope of the effort was quite large. Rather than reduce the scope to match the capacity of the smaller team, the decision was made to increase the team size to approximately 50 people.

The overall project's complexity attributes are listed in Table 4.13 and the uncertainty attributes in Table 4.14. The project is clearly in the bull category, with both high complexity and high uncertainty.

**Table 4.13** Complexity Attributes for Next Generation

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Team size | More than 50 | 9 | |
| Mission critical | New product | 3 | |
| Team location | Distributed in Houston, Austin, and Denver | 9 | |
| Team maturity | New team of experts | 3 | |
| Domain gaps | Not all developers are domain experts | 3 | |
| Dependencies | Major dependencies | 9 | |

**Table 4.14** Uncertainty Attributes for Next Generation

| Attribute | Comments | Score | Graph |
|---|---|---|---|
| Market uncertainty | Brand-new market space | 9 | |
| Technical uncertainty | Brand-new technology | 9 | |
| Number of customers | New solution for the market | 3 | |
| Project duration | Approximately 12 months | 9 | |
| Change | Embrace and/or create change | 9 | |

### What the Leaders Did

The project demonstrated some interesting dynamics. The developers had just read about Extreme Programming and wanted to try out some of the ideas; they knew there was a lot of uncertainty, so they wanted to run the project like a colt. The project managers had just seen the success of a well-run cow project, and, recognizing the complexity of this project, they wanted to run the project like a cow. Not surprisingly, the differing perspectives created some significant tension on the team. The project floundered for quite some time. It took a few iterations, but fortunately the team ultimately recognized that the project really was a bull and needed to be treated as such. Once they came to that realization, the project managers understood the need to be able to adapt to the uncertainty, and the developers realized the need to have structured communications and better documentation to deal with the complexity. In retrospect, it probably would have been wiser to stick with the original plan to start with a small team and let the project evolve over time from a colt to a bull.

### Iterative and Incremental Delivery

One key aspect of project management that is highlighted by this case study is the importance of iterative delivery and adaptation to cope with high uncertainty. This project was charged with delivering new products—yet no one knew exactly what the market needed. Iterative development was critical so that the team members could see working versions of the software, thereby enabling them to make adjustments and drive the product in the direction needed for the market. Likewise, it was important for team members to be able to reflect on the development process and discover what was working and what was not working. In this case, the team came to the realization that changes to their process would be necessary if they wanted to be successful.

## Leadership Development

As noted earlier in this chapter, the skills necessary to lead a bull project are quite different from the skills necessary to lead a sheepdog project.

Bull projects require leaders who can coordinate large teams and can navigate through the uncertainty minefield. Sometimes sheepdog projects can be simple enough that they require very little direct leadership. This way of looking at projects and the associated leadership styles required provides a basic tool for leadership development. Leaders exhibit excellence in four primary skill areas, as depicted in Figure 4.4:

- People: The ability to coordinate and lead people.
- Business: Connecting to and comprehending the business drivers.
- Process: Understanding the appropriate processes to get the job done.
- Technology: Understanding the technology used to develop the solutions.

As shown in Figure 4.5, the key leadership skills required to move from a sheepdog to a colt lie in the areas of business and technology. Why? Because the uncertainty is either in the marketplace or in the technology. As such, the leader of a colt project needs to have a good connection to the source of the uncertainty. Likewise, to move from a sheepdog to a cow requires skills in the areas of working with people and processes. The larger teams and the overall coordination associated with following such a path require a leader who can work well with people and utilize appropriate processes to facilitate the overall project coordination. To be capable of running a bull project, a project leader needs to have core skills in all

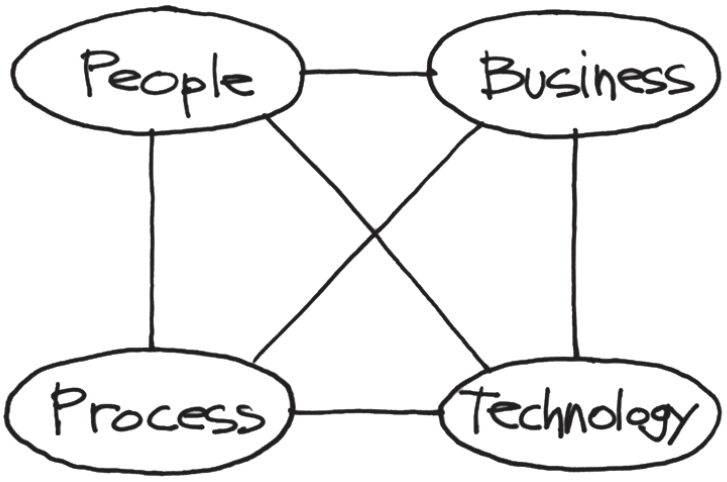


**FIGURE 4.4** Skill areas exhibited by project leaders

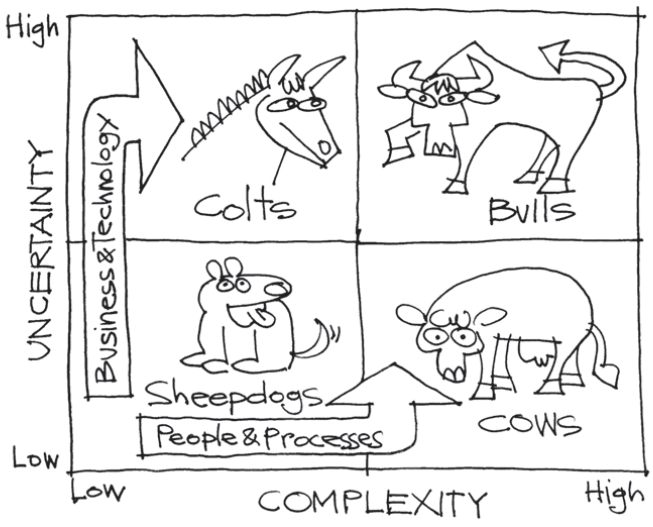


**FIGURE 4.5** Project profile

four areas. The leader doesn't need to be an expert in the technology or in the business, but does need to understand it well enough that he or she can comprehend the consequences of making and guiding decisions in these areas.

The five-stage Dreyfus model [1] is commonly used for assessing skill acquisition and competencies. Many people find it convenient to simplify this to a three-level progression from novice to practitioner to master. In the latter model, the three levels map to the actions of read, write, and delete. A novice can read rules and generally follow the procedures, but isn't proficient enough to be able to write rules. A solid practitioner is capable of writing rules for others to follow, but is generally still following standard procedures himself or herself. By comparison, someone who has reached the mastery level knows just which rules he or she needs to follow and which rules can be deleted.

Table 4.15 shows the base levels of competencies required by project quadrant. A leader of a sheepdog project must have core capabilities in all areas but need not excel in any. As previously mentioned, a leader of a colt project must be strong in the areas of business and technology. To lead a cow project, the leader must be strong in the areas of people and process. A leader of a bull project needs to be strong across all four areas and must have particularly strong skills in working with people. These leaders need

**Table 4.15** Competencies Required by Project Quadrant

| | People | Process | Technology | Business |
|---|---|---|---|---|
| Sheepdog | Novice | Novice | Novice | Novice |
| Colt | Novice | Novice | Practitioner | Practitioner |
| Cow | Practitioner | Practitioner | Novice | Novice |
| Bull | Master | Practitioner | Practitioner | Practitioner |

to have mastery in working with people and be able to break down the traditional rules on occasion to get things done.

It is important to match the appropriate leadership style and capability to the project needs. The leadership skills required to deal with uncertainty are not the same as those required to manage complexity. Some leaders are naturally drawn to managing uncertainty, whereas others are naturally drawn to managing complexity. For bull projects, a comprehensive set of skills is required to be able to manage both complexity and uncertainty. Some leaders who are naturally drawn to one dimension may find it difficult to cross over to the other dimension. Those who wish to develop their leadership potential will look to develop their skills by taking on more diverse challenges.

Some people can make the jump directly from leading a colt or cow project to leading a bull project. For others, it is advantageous to develop experience with a cow project before becoming a leader of a colt project, and vice versa. Leaders of sheepdogs can advance their leadership experience by taking on colts or cows or by taking on progressively more challenging sheepdogs.

## Portfolio Assessment

An assessment of certainty and complexity can also provide useful information when looking at an overall portfolio of projects. The most critical consideration is that the organization should not try to take on more projects in any particular quadrant than it has capable leaders for those projects. This constraint is particularly applicable for bull projects. An organization is unlikely to have a large number of project leaders who are capable of delivering on bull projects.

A typical portfolio will consist of projects that are distributed throughout the four quadrants of the uncertainty/complexity matrix (as shown in Figure 4.6). In the example shown in Figure 4.6, a high percentage of projects is in the sheepdog category. Obviously, a younger or very established organization might have a project distribution that skews in one direction or another. The important thing is to make sure that, whatever the distribution is, the organization understands the reasons for that distribution and then builds the organizational capacity to be able to make those projects happen.

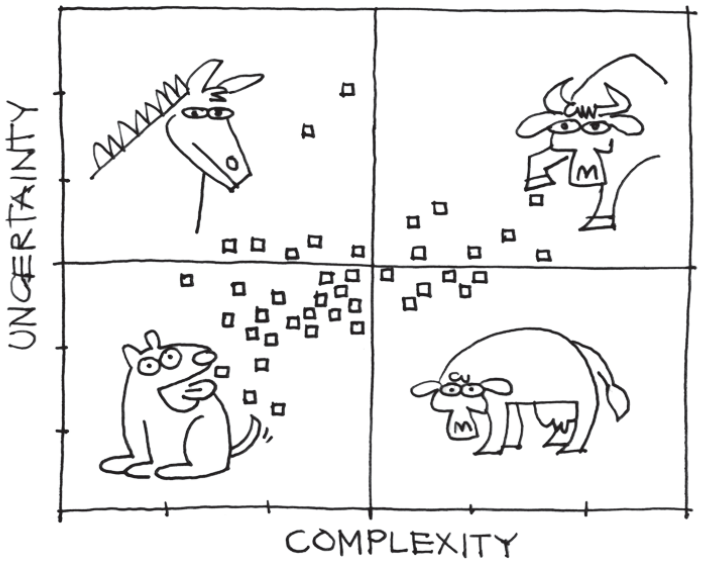


**FIGURE 4.6** Sample portfolio of project uncertainty versus project complexity

## Summary

Figure 4.7 provides an overall summary of the context leadership model. While all projects are not created equal, an examination of the complexity and uncertainty characteristics of projects is likely to reveal groupings of projects that behave similarly. Using this model can provide guidance for leaders to help with running projects and overall project portfolios. The complexity and uncertainty assessment can be used to better understand some portions of a project's risk profile and to look for opportunities to reduce some of that risk. The model also can play a role in leadership

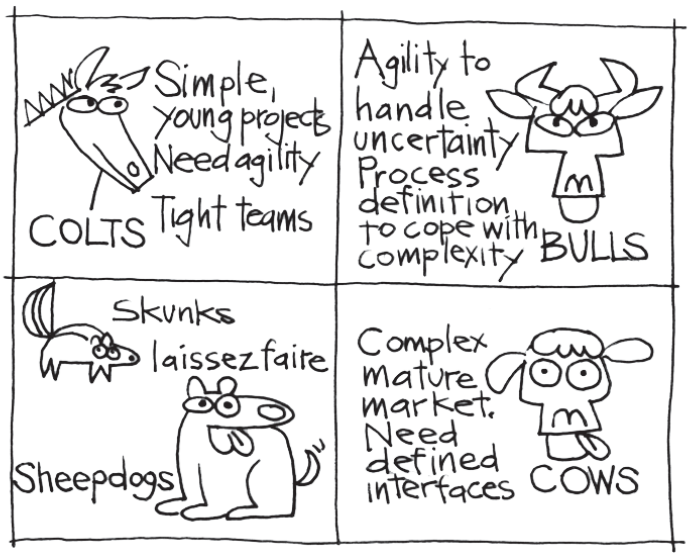


**FIGURE 4.7** Context leadership model: summary

development. The skills needed to manage uncertainty are different from those necessary to manage complexity. Aligning and developing leadership skills to be compatible with the project portfolio can significantly affect project success.

## References

[1] Dreyfus, Stuart, and Dreyfus, Hubert. A Five-Stage Model of the Mental Activities Involved in Directed Skill Acquisition. Berkeley, CA: University of California Operation Research Center, February 1980.