Manage Requirements Volatility to Manage Risks in IS Development Projects

Requirements volatility (RV) refers to additions, deletions and modifications of requirements during the systems development life cycle. RV creates rework in design and code that increases the system development cost and time and compromises the system quality. Ignoring requests for requirement changes can cause system failure due to user rejection, and failure to manage RV can increase the development time and cost.

A meta-analysis of several studies shows that system development projects have an average of 25 percent time overrun and 41 percent cost overrun. RV is the cause of failures in about 11 percent of system development projects. Information systems (IS) professionals consider RV a critical risk. The management of RV is essential to success in a systems development project.

This article discusses the risk factors for RV, suggests methods to manage RV to reduce project risks, and relates RV and its management to various control objectives in COBIT.

RISK FACTORS FOR RV
Strategic IS have an inherent risk for RV because such systems are new to a firm and possibly not implemented by any other company in the industry. Requirements for such systems are initially unclear and evolve over time. Since new technology takes time to evolve and mature, such technologies in systems can cause RV. Users’ inadequate knowledge of technology and developers’ inexperience with technology are additional sources of RV. Large systems with a large number of requirements have scope for many changes. Large systems without adequate user representation of all concerned departments can produce incomplete requirements that will be added later or incorrect requirements that will be corrected later. In complex systems, a change in requirement may start a chain reaction to cause more requirement changes, some of which may be missed and corrected later.

Another risk factor for RV is with unique and differentiated business processes that have unclear requirements initially and that evolve during development. Since system development is knowledge-intensive collaborative work, insufficient user knowledge of the application may lead to poor articulation of requirements that will need changes later. Similarly, if developers have insufficient knowledge of the application domain, they may not fully understand user requirements that will need changes later. Frequent turnover of developers and/or users in the development team will affect continuity and lead to changes in requirements. Projects that are forced to be completed earlier than the required time may lead to an incomplete and incorrect requirements definition and subsequent changes. Finally, users may add, modify or delete requirements due to lack of planning on their part or due to indecisiveness.

Figure 1 summarizes various risk factors for RV.

Recognizing any of these RV risk factors in a proposed systems development project can help to manage the consequent project management risks. Specifically, recognizing these risk factors would help in assessing the IT risks as in process PO9,
Assess and manage IT risks, in the Plan and Organize (PO) domain of COBIT. The next section discusses the responses to RV risks and the management of such risks.

RV RISK MANAGEMENT METHODS

RV is a risk believed to be uncontrollable and outside of a project manager’s influence. Although not all RVs can be controlled, they can be managed. Project managers can reject requirement changes that are not critical to achieving a system’s objectives, they can create a development environment that eliminates the causes for avoidable RV and they can use development methods that work with volatile requirements. Finally, where such methods are not feasible, methods to manage the RV and the consequent changes in design, development and implementation are essential. The following sections discuss methods to manage RV risks.

Reject RV

The easiest suggestion to manage RV is to freeze the requirements and reject the volatility. However, this may not be practical in many situations because users may reject the system without the changes. In addition, the expected system benefits may not be achieved without the requirement changes. A project manager may not be able to prevent the RV in systems that are controlled by project stakeholders at a higher level. Nevertheless, the manager may have freedom with the requirements of nonstrategic systems. In these systems, the manager can separate the essential requirements from useful and nice-to-have requirements and be able to freeze the nonessential requirements. However, the manager should have the authority to reject a requirement change, unless it is critical to achieving the system objectives, and to prevent the volatility of nonessential requirements.

Eliminate Avoidable Causes of Volatility

A previous section discussed factors such as unique and differentiated business processes in an information system that would have unclear requirements initially and that would evolve during development. A solution to deal with such volatility is to avoid unique and differentiated business processes that do not have major benefits in nonstrategic systems. It is better to migrate to standard and accepted business processes in such cases. Where it is not possible to avoid such unique processes, process experts should be co-opted to identify requirements correctly and completely.

Volatility created by new technology and inexperience with current technology can be reduced or avoided by co-opting technology consultants in the requirements definition process. The development team should include users and developers with knowledge in the application domain to avoid incorrect and incomplete requirements that will be changed later. Volatility caused by users’ insufficient knowledge of an application can be balanced by having process experts on the development team.

RV due to inadequate user representation of various functions can be minimized by having the right number of users to represent each function. Having a stable team of developers and users on the development team will ensure continuity and avoid requirement changes that come with new team members.

Structured definition methods are useful when documenting and managing requirements. These methods can also help to update volatile requirements correctly and completely to avoid further volatility. Verification and validation of requirements using inspections, reviews and walk-through exercises can ensure that requirements are correct, consistent and complete, and they can help to avoid changes to requirements during later stages.

Use Methods to Work With Volatility

The RV associated with strategic systems and new technology cannot be rejected or avoided. It is prudent to use appropriate requirement gathering and development methods with such systems. Iterative methods such as rapid application development (RAD) and agile development have been found to be useful in the development of systems that have RV, but these methods are not proven for large systems development. In these cases, a prototype can be developed iteratively and used to elicit clear requirements during the requirement definition stage. Prototypes and pilot projects are useful in resolving requirement uncertainties in strategic systems and in systems with new technologies. If developers do not have much experience with the technology, consultants and process experts should be co-opted to identify requirements.

The methods discussed in the previous three sections—methods to reject, eliminate or work with the risks—are consistent with control objective PO 10.9, Project risk management, in the PO domain of COBIT.
Manage Volatility and Its Effects

RV increases development cost and time, and, thus, its implementation needs to be managed carefully. An incorrect implementation of requirement changes may lead to more volatility and higher costs. A change management system is essential to identify the impact of a requirement change on cost and schedule as well as on other requirements and system objectives. Changes that have a big impact need to be approved by a committee of senior managers from the user departments and the project sponsor. Current change management methods focus largely on design and code artifacts and very little on the requirements. Structured requirement definition methods can help to trace a requirement to the system's objectives and to determine the impact of a requirement change on system objectives.

Top management commitment is the number one requirement for a project’s success. Top management support is essential in resolving conflicts that arise between users and in managing the associated volatility. Both the project and the project manager should have the support and commitment of top management in dealing with requirement changes and in using the RV management methods. These methods to manage the implementation of requirement changes map well to COBIT process AI6, Manage changes, in the Acquire and Implement (AI) domain.

**Figure 2** summarizes methods that can be used to manage RV risks.

Before executing a systems development project, the project manager should identify various RV risks discussed in the previous section and summarized in **Figure 1**. Depending upon the RV risk factor for the project, appropriate risk management methods as indicated in **Figure 3** should be adopted. Please note that management methods—such as getting top management commitment, arming the project manager with sufficient authority, verifying and validating requirements, and staffing the project with users and developers who are knowledgeable in the application area—are good practices that are essential in all types of systems development.

Project managers and IS developers are generally aware of risks due to RV, but other project stakeholders may not be aware of various RV risks and the implications of these risks. The project manager has the responsibility of communicating the RV risks to project stakeholders and in educating them in the management of these risks. The project manager and the head of IS development are jointly responsible for identifying and assessing the RV risks in consultation with the owner of the business process, architects of the system, IT administrators, and the information system audit and control group. They are responsible and accountable for identifying

**Figure 2—Methods to Manage RV Risks**

<table>
<thead>
<tr>
<th>General Methods</th>
<th>#</th>
<th>RV Risk Management Methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reject requirement volatility.</td>
<td>1</td>
<td>Freeze requirements and reject volatility.</td>
</tr>
<tr>
<td>Eliminate avoidable causes.</td>
<td>2</td>
<td>Avoid unique processes in nonstrategic systems.</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Use technology consultants.</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Use process experts.</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Ensure that all functions have adequate representation.</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>Use structured requirement definition methods.</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>Have a stable development team with low turnover.</td>
</tr>
<tr>
<td>Use methods to work with volatility.</td>
<td>8</td>
<td>Use pilot projects and prototypes in requirements analysis.</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>Use iterative development methods.</td>
</tr>
<tr>
<td>Manage volatility and its effects.</td>
<td>10</td>
<td>Use a change management system.</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>Use conflict resolution methods.</td>
</tr>
</tbody>
</table>

Elements that are essential to the success of all systems development projects:

- Top management commitment
- Project manager with sufficient authority
- Verification and validation of requirements
- Users/developers with knowledge of the application domain
and implementing methods to manage these risks. As the project progresses, the project manager is also responsible for keeping the chief information officer (CIO) and chief financial officer (CFO) informed of the management of RV risks and the impact of the risks on the project outcome.

CONCLUSION
The guidelines discussed in this article help to manage not only the RV risks in in-sourced system development projects, but also help to manage several other risks in these projects. For example, new and unique business processes should be avoided in nonstrategic systems because they not only create RV, they also create a risk of higher development costs without the concomitant benefits.

Outsourced projects would have additional risk factors for volatility of requirements. For example, outsourced projects with time and material contracts have a risk of higher volatility compared to fixed-price contracts. This may be due to vendors expanding the scope of a project to increase their revenue. Globally outsourced projects would have more RV risks due to separation of users and developers in distance and time and due to communication barriers created by language and culture. These types of projects warrant additional RV risk management methods.

ENDNOTES
2 The Standish Group, “Extreme CHAOS,” USA, 2001
4 Thakurta, R.; F. Ahlemann; “Understanding Requirements Volatility in Software Projects—An Empirical Investigation of Volatility Awareness, Management Approaches and Their Applicability,” 43rd Hawaii International Conference on System Sciences (HICSS 2010), USA, 2010
5 IT Governance Institute, COBIT 4.1, 2007, www.isaca.org/cobit
7 Op cit, Thakurta