

Improving Processes for Commercial Off-the-Shelf-Based Systems

Dr. Barbara Tyson, Cecilia Albert, and Lisa Brownsword
Software Engineering Institute



Tuesday, 29 April 2003
Track 5: 1:50 - 2:30
Room 150 D - G

Organizations using commercial off-the-shelf (COTS) products in critical business and mission systems find that the traditional process of defining requirements, formulating an architecture, and then trying to find COTS products to meet the specified requirements within the defined architecture rarely works. Instead, new processes, skills, and roles are required. Many organizations that have derived substantial benefits through process improvement using capability maturity models want to know, "How should the Capability Maturity Model® IntegrationSM (CMMI®) be interpreted for organizations building, fielding, and supporting a COTS-based system?" This article answers that question and describes the Evolutionary Process for Integrating COTS based systemsSM, which was designed to show how COTS aspects could be addressed; it also identifies high-level guidance to facilitate the definition of appropriate work processes for developers and maintainers of COTS-based systems using the CMMI.

For many programs, commercial off-the-shelf (COTS) products offer the promise of rapid delivery to end users, shared development costs with other customers, and an opportunity to expand business and mission capabilities and performance as improvements are made in the marketplace. But the promise of COTS products is often not realized in practice. Why? An important factor is that organizations tend either to assume that COTS products can be simply thrown together, or they fall back on familiar, traditional development skills, which have been shown not to work in developing and maintaining a COTS-based system [1].

Practical experience shows that building systems using COTS products requires new skills, knowledge, and abilities; changed roles and responsibilities; and different processes [1]. Many organizations find that COTS-based systems can be complex and are often costly to build and maintain. Moreover, practitioners are finding that management and engineering processes for a COTS-based system¹ must be more (not less) disciplined.

This article characterizes the unique aspects of defining, building, fielding, and supporting a COTS-based system; describes the Evolutionary Process for Integrating COTS based systemsSM (EPICSM) (which was designed to show how COTS aspects can be addressed [2]), and identifies high-level guidance to facilitate the definition of appropriate work processes for both developers and maintainers of COTS-based systems using the Capability Maturity Model® IntegrationSM (CMMI®) [3].

Demands of COTS-Based Systems

In custom development, a system can be created to meet the demands of a particular operating environment. A COTS-based system, for the most part, is composed of products that exist off-the-shelf. COTS products introduce unique dynamics and constraints that must be accommodated by any set of work processes that build, field, and support COTS-based systems such as the following:

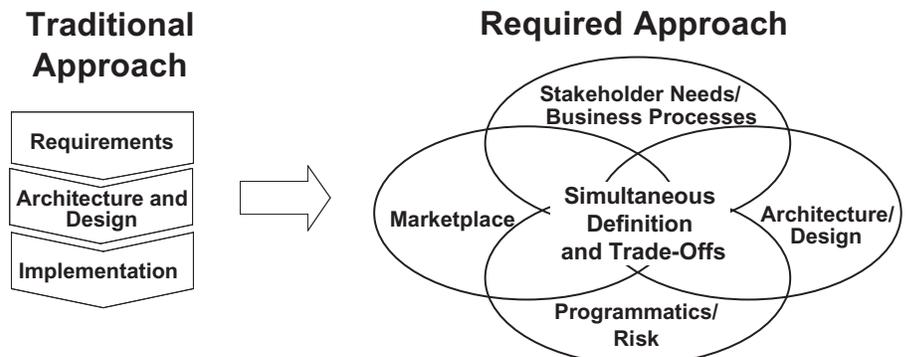
- COTS products are developed and enhanced in response to the vendor's perception of the needs of a broad set of customers – the commercial marketplace – not a particular customer's needs.
- COTS products include implicit assumptions about the way the product will be used, which seldom match the predefined operational processes of the project's end users.
- COTS products include implicit assumptions about the way the product will interact with other products and the enterprise infrastructure, including dependencies on specific versions of other

COTS products.

- The vendor often provides limited visibility into the assumptions, design, quality, and behavior of the COTS products.
- COTS products often behave in unpredictable ways when used in combination.
- The vendor controls the frequency and content of COTS product releases. In a competitive market segment, COTS products may add and/or delete functionality frequently.
- The vendor maintains the COTS product, retains data rights to the source code, and intends for the products to be used without product modification².

Despite these differences, many organizations have tried to use the more traditional approach as shown on the left in Figure 1. This approach defines the requirements, forms an architecture to meet them, and then searches the commercial marketplace for COTS products that fit into that architecture. This approach is rarely successful. COTS products do not fit. Organizations either resort to custom development or try to make COTS products fit by modifications. Either

Figure 1: Fundamental Engineering, Management, and Business Change



Note: Adapted from COTS-Based Systems for Program Managers [4]

SM EPIC and Evolutionary Process for Integrating COTS based systems are service marks of Carnegie Mellon University.

way, they incur significant cost and schedule impacts that are repeated with each product upgrade.

In contrast, the Software Engineering Institute's experience in examining more than 50 projects attempting to build COTS-based systems shows a fundamental change is required, as shown on the right in Figure 1. To effectively leverage COTS products, knowledge of how the products behave in the operational context and a projection of how that behavior is likely to change over time must influence the definition of the solution's requirements and end-user business processes, and will drive the definition and implementation of the resulting solution. Successful projects emphasize a balance among the following four competing spheres of influence throughout the project's life:

- **Stakeholder³ Needs and Business Processes.** An understanding of the relationship between project success and the organization's business drivers. How end-users will use the system with emphasis on the implications for end-user business processes. What the stakeholders want with emphasis on the minimum number of *must have* requirements.
- **Marketplace.** An awareness of marketplace drivers that are likely to affect the COTS products over the system's life. Knowledge of current and emerging COTS products, technologies, and standards relevant to the project.
- **Architecture and Design.** The essential elements of the system, any other systems or infrastructure with which it interacts, and the relationships among them (including structure, behavior, usage, etc.) so the components can work and evolve together.
- **Programmatics and Risk.** An understanding of the management aspects of the project, including the impact of

implementing any needed changes to the end users' operational processes. What the project and end-user community can tolerate in terms of cost, schedule, and risk.

While three of the spheres from the required approach in Figure 1 have analogues in traditional development processes, the marketplace is a potent addition. To accommodate the marketplace, each sphere must be defined based on knowledge of the marketplace.

For example, a stakeholder need may be stated such that any known COTS product cannot satisfy it. Similarly, a potential COTS product may not be compatible with the organization's existing infrastructure or use a licensing strategy that would be cost prohibitive.

Therefore, as information among spheres is analyzed, trade-offs among the spheres are identified that must be resolved through negotiation among the disparate stakeholders. In practice, this drives the practitioner to gather a little, synthesize and negotiate a little, and then gather a little more and synthesize and negotiate further. Due to COTS products' volatility, this cycle of gather, synthesize, and negotiate must be repeated until the system is replaced or retired. Further, the new release of an already selected COTS product may change system behavior.

An Evolutionary Integration Process

The EPIC evolved from a U.S. Air Force need to institutionalize a process that implements the necessary simultaneous definition and trade-offs of the required COTS approach. EPIC is documented [2] to provide an overview and detailed instruction. An Air Force organization and a commercial financial institution have started using EPIC across their programs.

EPIC does not *simply* evaluate and select COTS products. Rather, it leverages many of the elements of the Rational Unified Process [5] to integrate COTS lessons learned and disciplined spiral engineering practice [6] to define, develop, field, and support COTS-based solutions⁴.

The EPIC Framework

To maintain the required balance between the four spheres through the life of the solution, EPIC creates an environment that supports an evolving definition of the solution while systematically reducing the trade space within the spheres. As shown in Figure 2 and discussed in the following paragraphs, this environment consists of iteratively converging decisions and accumulating knowledge while increasing stakeholder buy-in.

Iteratively Converging Decisions

Reduce the Trade Space

While trade-offs are common in any engineering endeavor, trade-offs in EPIC are driven by an increasingly detailed knowledge of the COTS products' marketplace capabilities. Initially, as shown at the left of Figure 2, the trade space may be large, with great flexibility for negotiating trade-offs among the four spheres. However, a decision in one sphere influences, and is influenced by, decisions in the other spheres.

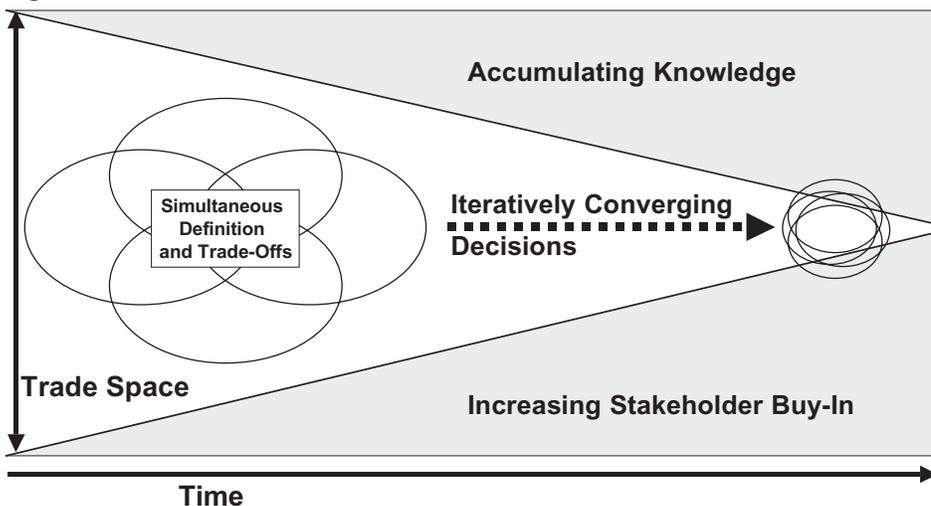
Over time, as the stakeholders' understanding of the solution evolves, decisions cause the spheres to converge. As they converge, the spheres become more interdependent, and the available trade space shrinks. Elements of the solution will continue to evolve until the solution is retired as business or operational needs change and new releases of COTS products become available.

Accumulating Knowledge Through Disciplined Risk-Based Spiral Practices

As the trade space diminishes, knowledge about the solution grows more detailed, which is reflected in the set of artifacts or work products necessary to build, field, and support the solution. Most of the artifacts are started in a rough form very early in the process and are expanded as more information is gathered and refined.

Due to marketplace volatility, keeping current knowledge about it is particularly important. In some cases, market events may invalidate already agreed-upon decisions (e.g., support for a product is dropped, a new product is introduced, or a product feature is added). While these disruptions have no easy resolution, relationships with vendors can provide warning of impending changes so appropriate actions can be taken.

Figure 2: The EPIC Environment



Increasing Buy-In Through Continuous Negotiation Among Stakeholders

An environment that includes all affected stakeholders is essential for timely resolution of mismatches between the available COTS products, the desired end-user business processes, and the stated stakeholder needs. In EPIC, *stakeholders* include the broadest set of individuals and organizations affected by the solution (or their empowered representatives). End users, one set of stakeholders, must be involved day-to-day to evaluate each COTS product's impact on the end-user business processes. In addition, the end-user needs will mature and change as their understanding of available COTS products increases. Concurrently, engineering stakeholders ensure that the COTS products considered can be effectively integrated with the organization's existing systems to meet required performance parameters and other system qualities. Business analysts ensure that viable vendors support the products. Vendors, another class of stakeholders, provide enhanced visibility into the COTS products' capabilities and gain potential insight into the organization's needs.

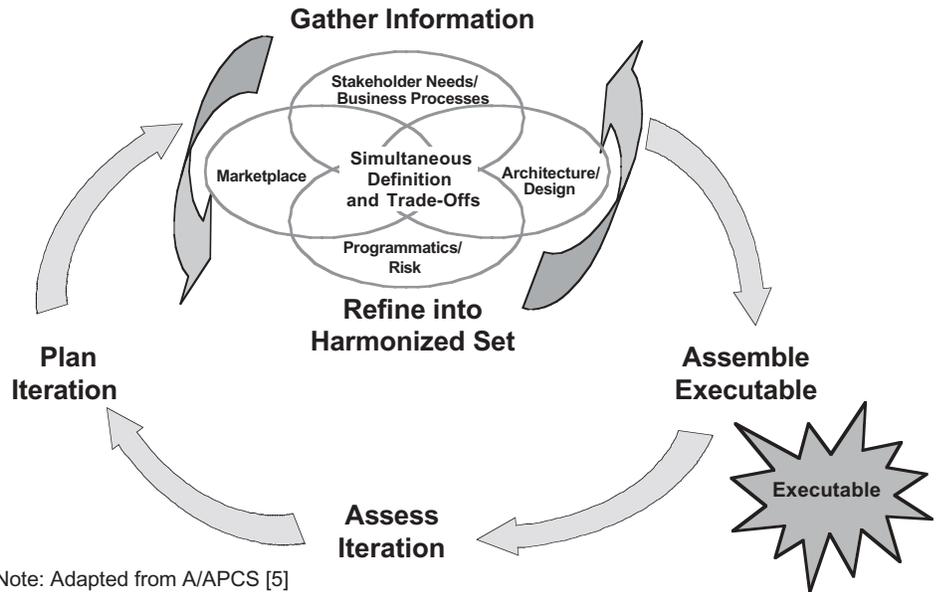
Using EPIC

To implement this environment, EPIC uses a risk-based spiral development process that keeps the requirements and architecture fluid as they are discovered and adjusts them to optimize using COTS products. The following sections summarize the COTS implications on the iteration and phase structure in which project and system activities take place.

Spheres of Influence Are Balanced in Every Iteration

Like other spiral development approaches, a series of iterations is defined across the life of the solution to mitigate specific project risks while addressing the most critical functions. In EPIC, each iteration consists of the fixed set of activities shown in Figure 3.

Each iteration begins with creating a detailed plan to meet defined iteration objectives. While many view the activities within an iteration to be a *mini-waterfall*, with COTS, information must be *simultaneously* gathered from each of the four spheres, as it is refined through analysis and stakeholder negotiation. Due to the interaction among spheres, many cycles of gathering and refining information may be required to produce a consistent set of information across all spheres to meet the iteration objectives. An executable representation of the solution is assembled to demonstrate the current understanding among the stakeholders. The iteration ends with an assessment of whether the objectives were met.



Note: Adapted from A/APCS [5]

Figure 3: An EPIC Iteration

Iterations Are Managed Through Clearly Defined Anchor Points

While the activities are the same for each iteration, the focus, depth, and breadth tend to change in character across the life of the system. As shown in Figure 4, EPIC uses the four Rational Unified Process phases (Inception, Elaboration, Construction, and Transition) and three corresponding anchor points (Life-Cycle Objectives, Life-Cycle Architecture, and Initial Operational Capability) to manage spiral development activities across the life cycle.

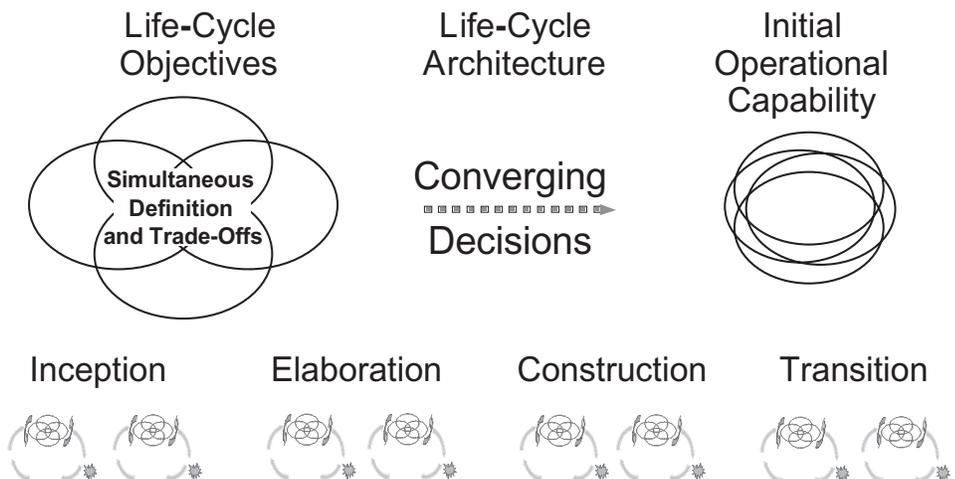
Like other spiral development processes, each phase consists of one or more iterations designed to meet explicit phase objectives and ends with an anchor point that provides an opportunity to review exit criteria, ensure continued stakeholder commitment, and decide to proceed, change project direction, or terminate the project based on progress to date. The following paragraphs summarize the goal of each phase and highlight some of the changes

made in EPIC to accommodate the unique character of COTS-based solutions.

Inception Phase: The Inception Phase establishes a common understanding among stakeholders of what the solution will do and why. It ends with the Life-Cycle Objectives anchor point when it is demonstrated that one or more, albeit high-level, candidate solutions can be integrated into the organization's broad architecture, in a reasonable period of time, at affordable cost, and for acceptable risk. In EPIC, candidate solutions may be formed around substantially different COTS products and, therefore, may address different user processes and stakeholder needs, use different architectures, and have different programmatic implications.

EPIC users have been surprised at the magnitude of the effort necessary to achieve the objectives of this phase. This is due in part to a lack of experience with spiral development processes. Just as significant, however, has been the need to reconcile a variety of

Figure 4: EPIC Phases



program mandates, which have included pre-selection of one or more COTS products by senior leadership, a predefined architectural framework, predefined target business processes, and arbitrary schedule and cost objectives. Identifying and negotiating these issues while the cost of resolving them was low has proven beneficial.

Elaboration Phase: The Life-Cycle Objectives anchor point marks a change in intensity. During the Elaboration Phase, stakeholders conduct in-depth experiments with candidate COTS products in a context that closely represents the operational environment. This phase ends with the Life-Cycle Architecture anchor point when all stakeholders agree that the defined solution provides sufficient operational value; the requirements, end-user business process, and architecture are sufficiently stable; and the solution can be assembled for acceptable cost, schedule, and risk.

While every effort is made to stabilize the solution, inevitably some unanticipated changes will occur in development and maintenance. In particular, new versions of the selected COTS products will require evaluation. Monitoring the marketplace for advance notice of these changed versions is required through the life of the solution.

Some EPIC users have applied this phase differently. One user views elaboration as an opportunity for rapid prototypes. Another uses EPIC to validate the utility of implementing a COTS-based versus custom solution. In both cases, this phase is used to converge on a stable definition of the solution to be implemented and then fielded.

Construction Phase: The Construction Phase focuses on preparing a production quality release of the solution suitable for fielding, and any necessary preparations in the target organizations to facilitate the initial fielding. In addition to developing any custom components, production rigor is applied to tailoring⁵ COTS products, developing integrating code (including wrappers, glue, etc.) and fully testing the system. The Construction Phase ends with the Initial Operational Capability anchor point when the solution is of sufficient quality for initial fielding to a subset of operational users.

No organization has yet progressed to this phase using EPIC.

Transition Phase: The Transition Phase fields and supports the solution across the user community. This requires proficiency in using the solution. As required, bugs are fixed, features are adjusted, new COTS product releases and patches are evaluated and considered for integration, and missing ele-

ments are added to the fielded solution in maintenance releases. The Transition Phase ends only when the solution is retired and/or replaced by a new solution.

Implications of COTS-Based Systems on CMMI Work Processes

Capability maturity models integrate total quality management, best practices targeted to an organization's domain, and organizational development practices to guide improvement to an organization's processes and its ability to manage the development, acquisition, and maintenance of goods⁶ or services. Many organizations have derived substantial benefits [4] from process improvement using capability maturity models.

The CMMI [3] currently contains four disciplines: Systems Engineering, Software Engineering, Integrated Product and Process Development (IPPD), and Supplier Sourcing. Each is critical to building, fielding, and supporting a COTS-based system. Systems Engineering and Software Engineering disciplines provide a basis for the necessary development and maintenance activities. IPPD provides for the timely stakeholder involvement to support negotiations among the spheres of influence. Supplier Sourcing provides practices for selecting and managing subcontractors that may be extended to COTS vendors.

Process areas⁷, the primary building blocks for the CMMI, are not a process – nor are they process descriptions. It is intended that practices in the CMMI process areas be interpreted using an in-depth knowledge of the organization, the business environment, and any other relevant circumstances to define and implement work processes that meet the organization's needs.

As illustrated in EPIC, the required COTS-based systems approach demands development and maintenance environments and work processes that support the following behaviors:

- Concurrent definition and evolution of the four spheres:
 - Business process engineering becomes integral to system engineering.
 - Requirements are formed through discovery of what is available in the commercial marketplace and any other sources.
 - A flexible system architecture is defined early and maintained until the system is retired.
 - Awareness of potential marketplace changes is kept current throughout system development and maintenance.
 - Cost, schedule, and risk implications

for implementing the system *and* any required business process changes are an integral part of all trades.

- Continuous negotiation among stakeholders.
- Disciplined spiral or iterative practices with frequent executable representations of the evolving system.

In the following sections, selected CMMI process areas are identified for each of these behaviors with guidance for interpreting the process area for COTS-based systems. Not all affected process areas are discussed nor is a full treatment of the needed interpretations provided. Although all process areas are important in a COTS-based system, this article focuses primarily on the project management and engineering process areas, with references to other process areas of particular relevance.

Concurrent Definition and Evolution of the Four Spheres

Development of a COTS-based system is essentially an act of reconciling the four diverse spheres of influence. The discovery of requirements, the solution design, and the formation of project parameters must be fully integrated with stakeholders' discovery and analysis of capabilities in the marketplace. In addition, periodic disruptions to this discovery must be accommodated. This implies the following:

- Decision Analysis and Resolution. Well-established, robust decision processes are required to manage continuous negotiation among the solution's stakeholders.
- Technical Solution. Alternative solutions, including solutions using mixes of different COTS products, must be continuously developed and analyzed to reflect changes across all four spheres.

Business Process Engineering Integral to System Engineering

COTS products implement the vendor's assumptions about how end-user business processes operate. This is very different from a custom development situation where the system is created to meet the demands of predefined operational processes. For COTS-based systems it means that the end-users must be willing to modify their business processes as they learn about candidate COTS products. And, end-user business processes may need to be redetermined and renegotiated with new releases of the COTS product across the life of the system.

The CMMI does not address changes to the processes in the functional units of the enterprise. However, the concepts in Organizational Process Focus can be expanded in application to plan and implement enterprise business or operational

process improvement. In addition, a shared vision of success among stakeholders, suitable incentives, and leadership (as described in the concepts in Organizational Environment for Integration) are critical to aligning business processes with alternative solutions.

Requirements Formed Through Discovery of What Is Available

For COTS-based systems, it is unrealistic to form a detailed set of requirements at the start of a project and force the system to meet that set. Three conditions cause this to be true. First, as previously discussed, to leverage the marketplace, requirements must be informed by an understanding of available COTS products. Second, as end users interact with candidate COTS products and better understand the capabilities in the marketplace their expectations for the solution tend to change. Third, the marketplace changes product capabilities and introduces new technologies that provide unforeseen opportunities and challenges. Thus, requirements need to be fluid enough to respond appropriately to changes in the marketplace across the life of the solution. This implies the following:

- **Requirements Development.** To aid in making trades, prioritizing the requirements is an essential practice for a COTS-based system. Stakeholders must agree on a minimum set of *must-have* requirements.
- **Requirements Management.** It is particularly important to begin disciplined and controlled requirements management at project start to track identified and negotiated trades.

Early Definition and Maintenance of a Flexible System Architecture

Since the COTS products are *owned* by the vendors, the framework by which the COTS products and other components of the system are combined to provide desired functionality – the architecture – becomes an important strategic asset. And, evolvability of the system becomes a critical quality attribute.

The architecture must be based on current and predictive knowledge of both the enterprise and the underlying technologies of relevant COTS products, and carefully crafted to insulate parts of the system from changes in other parts. The structure and cohesiveness of the architecture must be maintained while allowing the system to respond efficiently to continuous COTS product upgrades, technology advances, and new operational or business needs until the system is retired. This implies the following:

- **Technical Solution.** Alternate solutions need to describe the project standards or protocols (often referred to as *glue* code

or wrappers) that will be used to link COTS products and other system components. Previous *make-or-buy* decisions may need to be revisited when an existing product changes or a new product becomes available.

Continuous Awareness of Changes in the Marketplace

Relevant market segments and products must be monitored to anticipate and track any changes that could potentially affect the solution. Knowledge of key COTS products must be sufficiently detailed to understand their potential impacts and benefits to the system. Hands-on evaluation of key COTS products and prototypes of combinations of products is essential.

The vendor is an important stakeholder for the project providing unique insights into ways that products work. Developing and maintaining relationships with the vendors who supply key COTS products may help *influence* a vendor's product direction (vendors do not often respond to *direction* from a customer). The specific nature of the relationship will depend on the importance of the COTS product to the solution and the vendor involved. Not all vendors will encourage (or entertain) a close-working relationship. This implies the following:

- **Integrated Supplier Management.** *Partnering* with key vendors is critical. Vendors, however, will seldom allow process monitoring. Therefore, to determine product suitability, it is important to conduct a hands-on evaluation of each vendor's product releases (including any patches) in the context of their use in the system. In addition, establishing and maintaining relationships to influence (not direct) future product capabilities is critical. Relationships with key vendors' other customers, while not explicitly covered in CMMI, may amplify this influence.

Cost, Schedule, and Risk Implications Integral to All Trades

Each alternative solution must evaluate team skills and expertise required to implement, field, and support it as well as the associated cost, schedule, and risks. In addition, fielding of a COTS-based system includes the cost, schedule, and risks of implementing business process changes for the functional units that are affected by the solution as part of the total cost of ownership. This implies the following:

- **Technical Solution.** Engineering trades must include the risk, cost, schedule, and other programmatic factors that are associated with each alternative solution. Estimates of work product and task

attributes should be generated for each alternative.

Continuous Negotiation Among Stakeholders

Communication and effective decision-making processes are critical to a COTS-based system. Stakeholders who reflect the full diversity of interests must be available to quickly resolve mismatches among elements in the four spheres of influence as they are discovered, and agree that the evolving definition of the system will meet their needs. Stakeholders must be actively involved in the development process, particularly stakeholders who will use the system to meet enterprise objectives.

Integrated teaming among disparate stakeholders (as described in the process areas within the IPPD discipline) throughout development and maintenance is essential. The end users must be involved to confirm the results of any and all negotiations.

Disciplined Spiral or Iterative Practices With Frequent Executables

COTS-based systems are particularly suited to spiral development work processes. In spiral development, critical attributes of the solution are concurrently discovered through an evolutionary and continuous exploration of the highest risk elements of the system. Spiral development encourages frequent and direct feedback from stakeholders while reducing the risk of misunderstandings by producing and validating executable representations (prototypes or production releases) of the evolving solution. This implies the following:

- **Project Planning.** If not already implemented, extensive effort may be needed to revamp planning and engineering processes to align with a risk-based spiral development approach.
- **Risk Management.** The risk-management strategy needs to be robust enough to allow risk to (re)direct and manage the project. Tracking the effectiveness of risk mitigation is essential.
- **Technical Solution and Product Integration.** Frequent executable representations of the evolving alternative solutions provide critical insights into how the solution will operate and how the COTS products will be integrated to achieve essential solution behaviors.

Summary

COTS-based systems introduce unique challenges that demand fully integrated work processes to accommodate the volatility of the marketplace throughout the life of the system. In particular, COTS products and end-user operations must be reconciled – and

re-reconciled – as new product releases become available. This forces linkage between business process, engineering, and system development activities. Extensive communication and strong decision-making processes are necessary to facilitate cooperation, negotiation, and continuous validation of the evolving definition of the solution across potentially disparate stakeholders.

EPIC illustrates a way to realize the promise of COTS using a risk-based spiral development process to actively manage and balance knowledge across four spheres of influence. EPIC is more than a way to select a specific product. Rather, it shows a way to define, develop, field, and support a coherent solution composed of one or more COTS products, any required custom code, and implementation of any changes required to end-user processes.

As with any CMMI application, the unique aspects of COTS-based systems must drive the development of effective work processes. Developing and maintaining a COTS-based system is more than selecting products.

The authors solicit feedback from organizations implementing EPIC or similar processes. ♦

References

1. United States Air Force Science Advisory Board. Report on Ensuring Successful Implementation of Commercial Items in Air Force Systems.

SAB-TR-99-03. Washington, D.C.: SAB, 2000.

2. Albert, C., and L. Brownsword. Evolutionary Process for Integrating COTS-Based Systems (EPIC): An Overview. CMU/SEI-20030TR-009. Pittsburgh, PA: Software Engineering Institute, 2002 <www.sei.cmu.edu/publications/documents/02.reports/02tr009.html>.
3. Software Engineering Institute. CMMI Product Team: Capability Maturity Model® Integration, Version 1.1. CMU/SEI-2002-TR-11. Pittsburgh, PA: Software Engineering Institute, 2002.
4. Goldenson, D., and J. Herbsleb. After the Appraisal: A Systematic Survey of Process Improvement, Its Benefits, and Factors That Influence Success. CMU/SEI-95-TR-009. Pittsburgh, PA: Software Engineering Institute, 1995.
5. Kruchten, P. The Rational Unified Process: An Introduction. 2nd ed. Addison Wesley Longman, Inc., 2000.
6. Boehm, B. "A Spiral Model of Software Development and Enhancement." IEEE Computer May 1998: 61-72.

Notes

1. A COTS-based system can be one substantial COTS product tailored to provide needed functionality or multiple components from a variety of sources, including custom development, integrated to collectively provide functionality.
2. We use the term *modification* to mean

changes to the internals of a hardware device or the software code. This does not include vendor-provided mechanisms for tailoring the product to specific operational environments.

3. A stakeholder is any person or organization with a vested interest in the outcome of a project, including customers, developers, engineers, managers, manufacturers, end-users, etc.
4. In EPIC, a *solution* is the integrated assembly of one or more COTS products or other reuse components, any required custom code (including wrappers and *glue*), appropriate linkage to the organization's broad architecture, and any necessary end-user business process changes.
5. *Tailored* means non-source code adjustment necessary to integrate the COTS products into an operational system, e.g., scripts.
6. Goods are any tangible output intended for delivery to a customer or end user (CMMI uses *product*).
7. To distinguish them from generic process names, CMMI process area names are underlined.

Cecilia Albert and **Lisa Brownsword** will also be speaking at STC 2003 on "Evolutionary Process for Integrating COTS-based systems (EPIC)" on Tuesday, 29 April, Track 8, Room 251 D-F, from 3:00-3:40 p.m.

About the Authors



Barbara Tyson, Ph.D., is a senior member of the technical staff in the Software Engineering and Process Management Group at the Software Engineering Institute. She develops and promulgates software engineering processes and organizational change. Tyson provided software development and systems engineering technical and management support in industry, federal, and academic environments. She has taught graduate courses in marketing, management, organizational change, and information systems at Johns Hopkins and Marymount Universities.

Software Engineering Institute
4301 Wilson Blvd. Suite 902
Arlington, VA 22203
E-mail: btyson@sei.cmu.edu



Cecilia Albert is a senior member of the technical staff in the Commercial Off-the-Shelf-based systems Initiative at the Software Engineering Institute. Albert served in the U.S. Air Force where she developed major software programs for simulation, command and control, and mission processing of national satellite systems. She taught at the Industrial College of the Armed Forces, and managed the archive and dissemination programs at the National Imagery and Mapping Agency.

Software Engineering Institute
4301 Wilson Blvd. Suite 902
Arlington, VA 22203
E-mail: cca@sei.cmu.edu



Lisa Brownsword is a senior member of the technical staff in the Commercial Off-the-Shelf-based systems Initiative at the Software Engineering Institute. Brownsword was on staff at the Computer Sciences Corporation in support of NASA/Goddard's Software Engineering Lab and has worked for Rational Software Corporation where she provided consulting to managers and technical practitioners in the use of transitioning to software engineering practices, including architecture-centered development, product lines, object technology, and Ada.

Software Engineering Institute
4301 Wilson Blvd. Suite 902
Arlington, VA 22203
E-mail: llb@sei.cmu.edu