

# EPIC

---

## Summary

EPIC (Evolutionary Process for Integrating COTS-based systems) is designed to support building, fielding, and supporting software – intensive systems using available Commercial-Of-The-Shelf (COTS) solutions. This white paper provides a high level overview of EPIC, differences between RUP and EPIC and the rationale behind the differences. The paper also deals with EPIC objectives for each RUP phase, i.e. Inception, Elaboration, Construction and Transition. The paper draws heavily on information from a Technical Report by Cecilia Albert and Lisa Brownsword of SEI (Software Engineering Institute).

## Introduction

EPIC (Evolutionary Process for Integrating COTS-based systems) is a version of RUP, customized for implementing COTS (Commercially-Off-The-Shelf) systems. The essential difference in COTS implementation is that stakeholder needs have to be traded off against what is available in the marketplace and a via media has to be worked out, that balances costs and market availability against adherence to requirements.

To start with, we have to cope with significant differences between stake holder needs, solution availability, design requirements and implementation risks. The essence of the methodology is to reduce the differences as we progress along the lifecycle and achieve near total convergence by end of transition. The diagram below captures this essence.

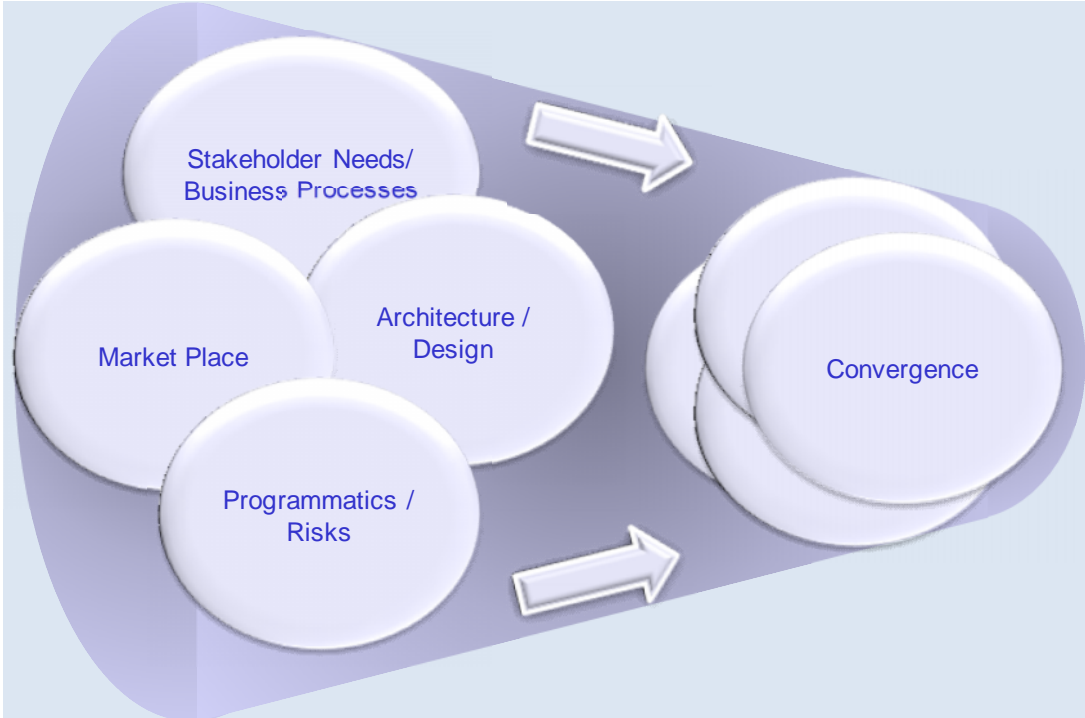


Figure 1 - EPIC - Progress to Convergence

As we move towards final selection, the market space diminishes, while knowledge about individual solutions increase, including architectural impact, business impact, and the comparative capabilities. While decisions are converging and knowledge is accumulating, stakeholders must have increasing commitment to the evolving definition of the solution and its iteratively evolving executable representation. Active participation from stakeholders is crucial at this stage.

COTS packages typically have process and domain knowledge embedded within it and these embedded processes are considered best-of-breed, abstracted from a host of implementations. Hence, companies have to find a via media between using the package processes as is (i.e. modifying current processes) and incurring additional expenditure in building customizations. More often than not, companies adopt the package processes for the most part and change their current operating modes to fit in these processes.

### EPIC vs. RUP

---

This process of accumulating knowledge, morphing of stakeholder needs in the march towards convergence, combined with the normal lifecycle processes of requirements, design, build and testing throws up some interesting differences in COTS implementation with respect to typical RUP processes. Some of these differences in the typical lifecycle activities are discussed below:

- **Requirement** – In EPIC, the solution is not derived from the requirements – available solutions are matched against stakeholder needs and a solution is selected based on trade-offs. This constitutes one of the major differences between normal solution development and COTS implementation. Requirements must stay fluid until the solution availability implications are understood. Business modeling is mandatory and not optional.
- **Solution Selection** - is a major pre-construction activity and this has to find a place in the RUP iterative life cycle.
- **Configuration** – Configuration is a critical work stream in COTS implementation. Typically COTS packages would have a global configuration, which is born out of the extensive experience of the COTS vendor. The global configuration could also constitute part of the best practices or part of an industry content pack provided by the vendor to accelerate implementation. Typically the global configuration is the starting point for the configuration work stream and in some cases could serve as the basis for requirements consolidation. Configuration activities will find place in elaboration, construction and transition phases.
- **POC** – POC may be done in two parts.
  - As part of the selection process, we may conduct multiple POC's - one or more POC's for each candidate solution.
  - Once the selection is done, we may conduct a further POC to validate some of our architectural assumptions

- **Architecture** - In normal solution development, we create the application architecture, while ensuring it is in line with enterprise guidelines. In COTS, the application architecture is largely provided by the vendor and bulk of the effort is to understand the integration effort to ensure realization of significant use cases and ensure compatibility with the enterprise architecture.
- **Analysis & Design** – Analysis must start sooner, i.e. in parallel with requirement activities. Design will be done to meet customization / integration requirements, as against the normal solution development, where bulk of the design effort is towards meeting functionality requirements.
- **Build** – A large part of the build effort will be towards configuration and customization activities.
- **Transition** – The transition effort could potentially start much earlier, as compared to normal solution development, as the solution (with global configuration and without customization) is already available. We could for instance, establish very early in the lifecycle, a sandbox environment that will enable users gain familiarity with the solution. We could also kick off performance / capacity planning activities much before go live, using a few user licenses.

The diagram below captures the phasing of the various differences in COTS implementation:

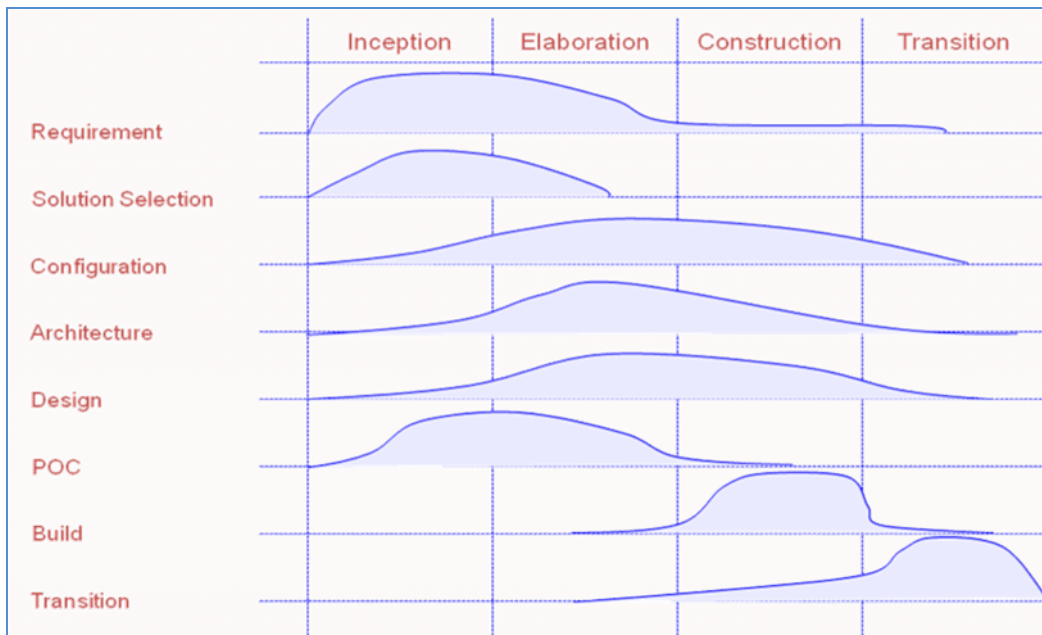


Figure 2 - COTS Differences by RUP phases

Besides the above differences, which maps into typical lifecycle activities, there are other differences, which could potentially impact planning and costs:

- **Project Management** – PM activities include management of vendor / supplier relationships.
- **Monitoring marketplace** – This is an activity that should be factored in while planning. The monitoring should (at a minimum) continue till end of the construction phase.
- **Other Activities** – Business, contracting and organizational change activities are integrated across the lifecycle.

## Phase Objectives

---

The objectives for each of the four phases have been discussed below for an EPIC implementation:

### Inception

The prime objective of the inception phase is to gain a clear picture of the problem space. In the current case, we have to understand the four disparate spheres, which require to be converged:

- Market Place
  - We have to gain understanding of what is available, and gain insight into each vendor's viability.
- Stakeholder needs
  - A high level understanding of stakeholder needs, expectation and constraints is essential. As we define solutions, these needs and expectations will be challenged and will morph into realizable representations.
- Architecture
  - We have to gain quick insights into impacts on the enterprise architecture and possible integration challenges.
- Risks
  - Besides the normal risks of cost overruns and delays, the risks involved in inducting an alien technology into an existing eco-system will always pose some risks. These risks have to be recognized and mitigation strategies should be planned.

The inception phase ends with the LCO (Life Cycle Objective) anchor point. In EPIC, LCO means one or more candidate solutions are identified, that meet the high level objectives and that have key stakeholder concurrence.

### Elaboration Phase

The basic activities of this phase are the same as those in the inception phase, with higher level of detail and greater level of resource commitment. The focus of this phase is on in-depth hands-on experiments with candidate solutions. Stakeholder needs and business processes will be further refined. There will be some prototyping to validate strategies for integration, component tailoring (customization) and migration.

The elaboration phase ends with LCA (Life Cycle Architecture) anchor point, when all stakeholders agree that the selected solution provides sufficient operational value to stakeholders and can be assembled within acceptable cost, schedule and risk.

### Construction Phase

The focus of the construction phase is preparation of a production-quality release of the selected solution. Custom components are developed, integration codes are written and migration strategies are

finalized. The construction phase includes preparation of necessary support materials, such as installation instructions, version descriptions, user and operator manuals etc.

The construction phase also envisions preparation of the business environment for initial fielding of the solution. The preparation includes development of required policies and procedures, organizational restructuring, implementing changes to business processes, and institution of mechanisms to promote adoption, such as establishment of initiatives, user groups etc.

Incorporation of any change in the final implementation can be easily incorporated before this phase is complete and this constitutes an opportunity. Hence, continued monitoring of the market place to keep abreast of development of new versions, new components etc. would be important during this phase.

The construction phase ends with IOC (Initial Operational Capability) anchor point, which allows stakeholders to verify that a production quality release of the solution is ready for fielding to at least a subset of the operational users.

### Transition Phase

The transition phase is focused on moving the solution to the user community. Users must attain required proficiency in the solution and end-user business processes must gain capability for self-support. The transition phase begins with an initial fielding to a limited cross-section of users and the user base will be progressively enhanced as we progress in this phase.

The transition phase envisages continued support for the solution and the phase ends only when the solution is retired and is replaced by a new solution. It is incumbent on the implementation organization to document and transfer knowledge that has been gained in the previous phases to the support organization.

### ABOUT THE AUTHORS

Object Edge is a high-end business & technology services company with proven track record in enterprise/application architecture, IT strategy and integration services. The core competence of Object Edge is business/ technology architecture. The company effectively combines business and technology expertise to provide innovative solutions for complex business problems. Object Edge practices include enterprise SDLC using RUP/UML and IT governance.