3 chapter

The Quality Process Architecture

The change from craftsmanship to industrialization does not come with the change to a new technique. The change must come on a more fundamental level which also includes the organization of the complete development process.
—Jacobson [1992]

Putting This Chapter in Perspective

The previous chapter focused on creating the right environment, and eliciting management support for necessary quality-assurance activities and tasks. A crucial aspect of that quality environment is a quality process. In this chapter we discuss what constitutes such a process, and how it helps to enhance quality in a UML-based project. This chapter does not propose a new process, but discusses the most commonly used activities and tasks that should be present in all processes. These activities and tasks and their related roles and deliverables are described with the aim of improving the discipline in a process, resulting in the enhanced quality of UML-based deliverables and, eventually, the software product.

This chapter starts by mapping the three dimensions of a process (the what, how, and who as mentioned in Chapter 1) to the corresponding examples in UML-based projects. Unlike the UML metamodel, though, we do not yet have an officially standardized process metamodel. Therefore, to simplify our process discussion, we develop the mapping of the three dimensions of a process into an unofficial metamodel for processes.

Various process-components are then derived from this metamodel. In identifying and describing the process-components, we consider the most basic or core elements of a process. Furthermore, we consider in greater detail the methodological or "how to" aspect of the process-components (their activities and tasks). Relevant deliverables ("what") and roles ("who") are also mentioned in the process-components. This is followed by descriptions of the necessity and sufficiency aspects of quality of the process-components. Malleability is part of process enactment and therefore not discussed separately for each process-component. Thus, this chapter deals with the construction of a process. Once a quality-conscious process is created, it is ready for deployment and enactment—topics dealt with in Chapter 4.

The Prime Reader: Process Engineer, Quality Manager

3.1 The Process Backbone

In Sections 1.4.2 and 1.4.3, we considered the three dimensions of a process and discussed a simple baking example to demonstrate them. Here, we begin the discussion of a process by extending that example further, in order to put it in a process metamodel. This will help us immensely as we construct the quality software process-components later in the chapter. We will also use that understanding to facilitate the creation of a process metamodel.

3.1.1 The Three Dimensions of a Process

The first dimension of a process develops an understanding of the materials on which the actions are performed, as well as the tools that help to perform those actions. This forms the technological dimension of a process. In the example of baking a cake, the activities related to this technological dimension are the evaluation of the dish, the ingredients, and the equipment. This constitutes the "what" of a process.

The second dimension of a process is the step-by-step guide to "how" a particular process is conducted. The discipline of conducting a process comprises a sequence of well-defined activities and tasks that are organized in a specific order. This discipline, or *method* of working, constitutes the methodological dimension of a process. Some activities corresponding

to this dimension in the baking example are recipe, cookbook, and timing. This constitutes the "how" of a process.

The third dimension of a process deals with the people who are going to take responsibility for the actions, and carry them out by following the prescribed methodology. An understanding of the dimension of a process that deals with the people who carry out the tasks, and the environment or the organizational context in which those tasks are carried out, results in the sociology of a process—the sociological dimension. In the baking example, the sociological aspect is comprised of the cook, the kitchen environment, and the "soft" issue of cake presentation.

Quality assurance, although a part of the overall process, has a separate set of deliverables, activities, tasks, and roles. Therefore, these elements of the process are separately defined—focusing on the quality management, quality assurance, and quality control suite of activities. These quality-related activities continue to influence the rest of the process. They are also self-influencing in the sense that each quality activity can be used to perform quality assurance on itself. For example, an activity of workshopping in a quality-assurance part of the process can be used to verify the process of conducting workshops in a process. Understanding these three dimensions is crucial to understanding the logic behind the metamodel for a process.

3.1.2 "What" of a Process

The "what" of a process is the technological dimension of the process that answers everything related to the physical things in the process. This primarily includes the raw material with which the process is executed, as well as the deliverables. It is concerned with the technology of the process. Many factors from the technological dimension influence the overall output of a process. These factors include the quality of material that is used, the availability of the material, and the appropriateness of the tools that are used in the process. Thus, everything that deals with the "what" in a process plays a part in influencing its deliverable. Examples of some of these technological factors in a UML-based project are summarized in Table 3.1.

Table 3.1 *Influence of technology factors (what) in a UML-based project*

	Model (components, other deliverables)	Tool (TogetherSoft or ROSE, for example)
Availability	Problem statements, existing applications, MOPS, MOSS, and MOBS	Is TogetherSoft available, or is ROSE better? Consider other options
Standards	UML as a standard for modeling; documentation of project standards	Compliance of TogetherSoft to UML standards
Appropriateness	Is MOPS the right thing to create a model in problem space?	Is TogetherSoft the right tool for modeling? (for example, it can't be used for a process)

3.1.3 "How" of a Process

The "how" or the methodological aspect of the process deals with the steps you follow to produce the deliverable. It is essentially a glossary of the distilled experiences of numerous project managers. The "how" of a process is instrumental in conserving effort when the process is enacted. Taking the cooking example further, the "how" of the process is the recipe book for the baking process. A description of the activities and tasks using suitable notations and language is essential to expressing the "how" of a process. In the case of UML-based projects, examples of the methodological dimension of the process are as follows:

Table 3.2 *Influence of methodological factors (how) in a UML-based project*

	Requirements Modeling Process-component (and others, as described later in this chapter)
Notations	UML notations for models; process notations, as described later in this chapter.
Language (description)	The actual description of the methodological aspect (the how)—by using the building blocks of a process. These are the process-components described later in this chapter.
Documentation	This is the accompanying standard for process, models, the description of how to use them, and their acceptance criteria.

3.1.4 "Who" of a Process

Simply following a method, as described in a recipe book, does not produce the desired deliverable. Application of the methodology is the purview of the person who is applying it. Success depends on the skills and experience of the person, as well as the environment in which she is working. Thus skills, experience, and environment form the sociological factors, or the "who" of a project. These are also called soft factors, as they deal with the softer or human-relations issues of a process. Refer to Chapter 2 for further discussion of soft (not easily quantifiable) factors.

Skills, one of the factors influencing the final outcome of a process, require regular training—especially when new ways of doing things and new equipment become available on a daily basis. Experience comes from practicing developed skills in a real environment. In addition to the skills and experience, it is also important to consider the motivational factors and their influence on the process. These are some of the sociological factors that continue to influence the final outcome of the project. For a UML-based project, some of these sociological factors are summarized in Table 3.3.

Table 3.3 *Influence of sociological factors (who) in a UML-based project*

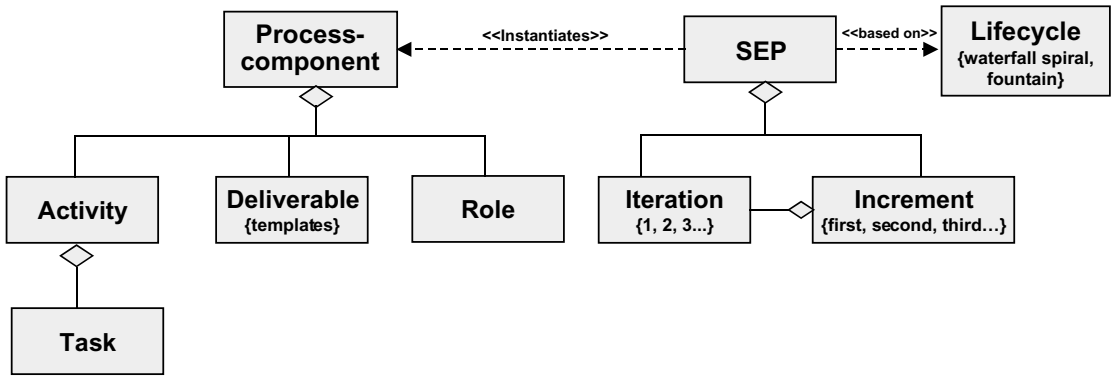
	Business Analyst (and so on with other roles)
Skills (role description)	The title and description of the role is described here. Also, the relevant skill set needed for this role is described here. A business analyst must have good business domain knowledge, familiarity with UML notations, comfort with use case and activity diagrams, and good interviewing and workshopping techniques.
Experience	A minimum of two years of business analysis experience might be considered essential for a person to be able to "get up to speed" in week one of a project. However, business analysts with less experience can also provide significant input into the project, provided they have had additional training in both UML techniques and the process.
Environment	The sociological factor or e-factor, as described in Chapter 2, should be handled here. Proper working environments (desks, phones, email facilities, decent meeting areas, and so on) are physical considerations that should be addressed in a UML-based project.

3.2 The Process Metamodel

3.2.1 Describing the Process Metamodel

In order to better understand the three dimensions of the process described above, and to further describe the various elements of a process as applicable to UML-based software development, we created the metamodel shown in Figure 3.1. It is not a formal metamodel, but one created to explain the process-components described later in this chapter. A metamodel is a succinct and helpful way of understanding things. For example, the UML has a metamodel that encapsulates the rules that govern the creation of UML diagrams. A look at the UML metamodel can tell us how a class relates to the attributes and operations inside of it.

The purpose of the process metamodel, as shown in Figure 3.1, is to provide the underlying rules for how the process elements connect to and relate with each other. Discussions on formalizing a process metamodel are underway at the OMG. In the meantime, this simplistic process



Note 1: This is an informal process metamodel created by me. It is not an OMG standard!

Note 2: The process-components provide the building block. They are put together in a SEP. Eventually, they have to be enacted (enactment is not shown here).

Figure 3.1 *A Quality Software Process Metamodel (using UML notations)*

metamodel serves our purpose of discussing and creating a Quality Software Process (QSP). The metamodel shown in Figure 3.1 uses the UML's class diagram notations, which readers should know. It can be interpreted as follows:

The *process-component* is shown as the building block for the process. This process-component is made up of *activities, deliverables,* and *roles.* Activities associate with *tasks.* The *lifecycle* provides the philosophical background for construction of a process. Examples of software lifecycles are shown as waterfall, spiral, and fountain models. A Software Engineering Process (SEP) is based on this lifecycle. In order to create a SEP, the process-components are instantiated. The SEP is made up of *iterations,* which can be 1, 2, 3, and so on. Similarly, a SEP is also made up of *increments,* which are first, second, and third (and can be many more). Increments are made up of iterations.

Each of these elements, appearing in the process metamodel, is further described with their corresponding notations.

3.2.2 Process Ingredients

Figure 3.2 shows the notations that are used to describe a process. These notations represent the elements that constitute the "what," "how," and "who" of the process. Some of these notations are UML-like, such as the role. Others, like deliverables, are needed to express the process aspect. These notations are also simple. They can be easily written on whiteboards, facilitating process-based discussion and leaving the opportunity open for other uses of these process elements, such as describing processes in a typical business exercise. These process elements can also be inserted in a process-based CASE tool that in medium to large projects will greatly ease their enactment. Each of the process elements shown in Figure 3.2 is described in the subsequent sections.

3.2.3 The Role Element in a Process

The *role* provides a descriptive profile of the person who is involved in executing the process. In a quality-conscious process, this role is properly defined and documented. A person will fulfill the description of the role provided here. The person playing the given role is responsible for carrying out the process. He or she can also be the recipient of the process. If use case proponents wish to use the term actor, they may—despite the fine

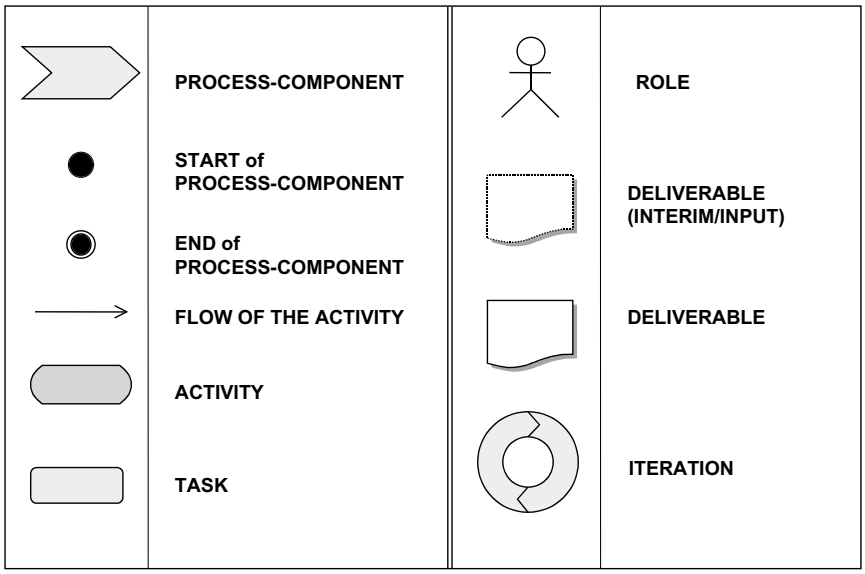


Figure 3.2 *Notations used to describe various process ingredients*

difference between actor and role. Some of the characteristics of a good quality role are:

- The role is well defined and is understood by the person who is responsible for the role.
- The person playing the role should be able to understand the activities that he is responsible for.
- The role must be assigned the suite of activities and tasks that the role player is capable of performing.
- Depending on the scope of the process, the actor element can have multiple instances. For example, a large development process may have 20 programmers but a small-scoped process may have only two.
- Examples of roles defined for a UML-based project include Project Manager, Business Analyst, System Designer and Tester, and Developer/Programmer.

3.2.4 The Activity Element in a Process

The *activity* is the description of the responsibility of the role in the process. The activity element is shown with an elliptical rectangle, and it describes in general what the role encompasses. Activities have a sequence or dependencies. Some activities can also be performed in parallel by more than one role. The activity element in a process is the controlling element for a set of tasks within the process. Therefore, the activity element on its own doesn't have the same concrete existence as the tasks. Actors playing the roles described above carry out the activities by performing a sequence of tasks within the activities. Some of the characteristics of an activity element are:

- Activity is the overall controlling element for a set of tasks.
- It is helpful in understanding the flow of events.
- Some activities may begin before others end. Thus, activities may be carried out in parallel.
- Activities are accomplished by completing the set of tasks which they encompass.
- It is not essential for all tasks within an activity to be accomplished in a single iteration.
- Example activities within UML-based projects include storyboarding, business class modeling, use case modeling, operational analysis, advanced interface design, quality resourcing, and test execution.

3.2.5 The Task Element in a Process

The *task* element in a process discipline is the atomic element in the working of a process. As shown in Figure 3.2, tasks are rectangles with rounded edges. Tasks are carried out under the umbrella of the encompassing activity. Thus, the purpose of the well-defined tasks is to complete the activity under which they fall. Some characteristics of the task element include the following:

- They are the performing elements in a process; that is, they don't need to be further subdivided before they are performed.
- A set of tasks belongs to an activity.
- In the overall process, these tasks are usually carried out in a specific sequence. The designer of the process usually specifies the sequence.
- However, since activities may sometimes be performed in parallel, so can tasks.

- The result of the execution of tasks in a sequence is the completion of an activity.
- Tasks have a concrete existence of their own. This implies that when a task is completed, it is effectively an incremental completion of the activity under which the task is performed.
- Tasks are what the project manager puts in a project plan. Thus, they can be assigned a time for completion and resources to complete them.
- Examples of tasks in a UML-based project are draw a business class diagram, conduct research, apply equality review, and execute a prototype.
- Techniques for carrying out a task may be described.

3.2.6 The Deliverable Element in a Process

A *deliverable* is the final output or result of the process. The roles (actors) are involved in performing various activities, which are carried out by executing a set of well-defined tasks. These tasks result in creating and upgrading what are called deliverables. Since deliverables are the result of work carried out by the roles, they are also called "work products." Deliverables can be concrete, as in a set of documents, or they can be abstract, as in a motivated work force (which results from work performed by a project manager). In our cooking example, the final deliverable is a baked cake. Deliverables in a UML-based project are usually produced iteratively. That means, even if a deliverable is shown as being produced as a result of activities, only some of the tasks within the activities will result in a partial completion of the deliverables. Eventually, more activities and tasks within the activities will be performed to complete the deliverable. This deliverable and its corresponding notation are shown in Figure 3.2.

3.2.7 A Process-Component

A *process-component* is a collection of a subset of the activities, tasks, roles, and deliverables in a process. Thus, a process-component indicates a logical collection of process elements that combine to accomplish a sizeable chunk of the process. The term process-component signifies that a suite of process elements is treated as a component, having a common set of roles working on a logically cohesive set of activities and tasks, resulting in a significant deliverable within that area of the process. Examples of process-components in a UML-based project are Business Evaluation, Requirements Modeling, System Architecture, and Quality Control.

3.2.8 Iterations

An *iteration* signifies an execution of a sequence of process-components, but with varying intensity. For example, some process-components related to evaluating the business proposal for a project are performed with high intensity in the first iteration, but the process-components dealing with requirements modeling are performed with high intensity in the second iteration. An iteration in a medium-sized project may last for about three months, at the end of which reviewable deliverables should be produced. Larger projects will need more time to complete iteration.

3.2.9 Putting Together a Process-Component: A Baking Process

Once the notations are described and understood by the process participants, the representation of the process by means of a diagram plays a significant part in conveying the activities, their sequence, the final deliverable being produced, and the roles responsible for producing those deliverables. Figure 3.3 shows graphically, by using the notations described in detail in the previous section, the simple baking process that we have been using as an example.

- There are three formal roles that are involved in this process-component. They are the chef, the food inspector (quality manager), and the consumer.
- The process is made up of a series of activities: environment creation, baking, quality checking, and consuming. Each of these activities has a set of tasks to go with it. For example, environment creation will have the tasks of procuring the raw materials, like sugar and butter, and preparing the kitchen. The activity of baking includes tasks like mixing the batter, preheating the oven, putting the pan in the oven, and taking the cake out of the oven at the right time. For a large cake (for example, a wedding cake) the activity of quality checking will come into play, followed by consumption. Consumption activity is shown as iterative, and it may be performed numerous times. The consumers of the cake will be multiple people filling the role.
- The final deliverable is shown as a baked caked. If necessary, the raw materials going into the process-component can also be shown (they are not shown here).

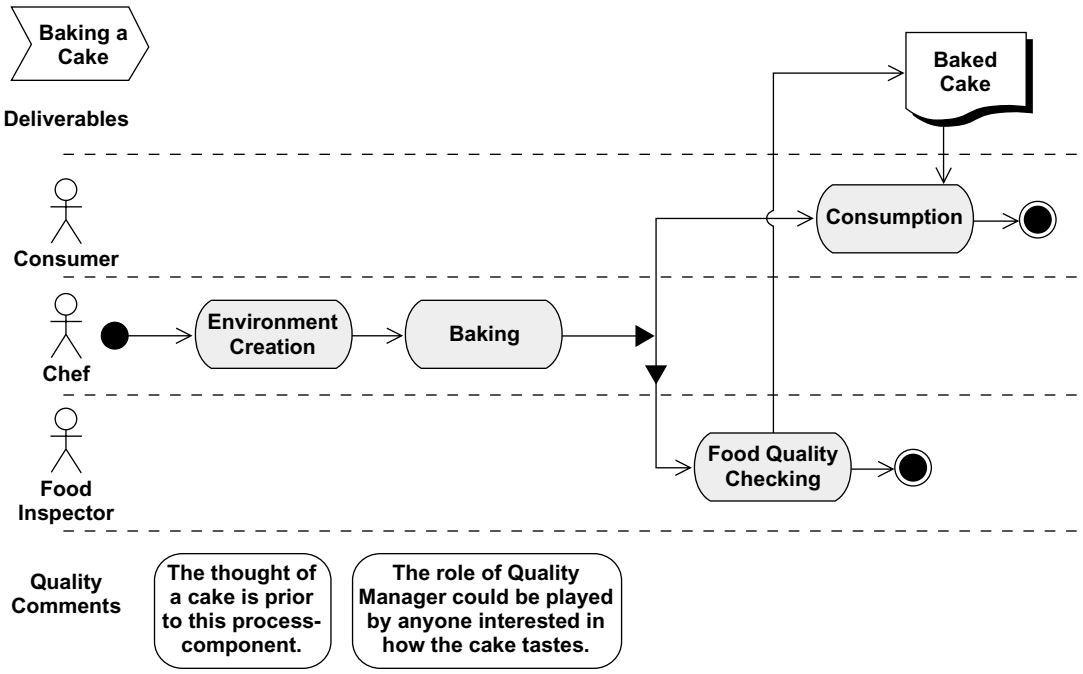


Figure 3.3 *Putting together an example baking process-component*

3.3 Quality Software Process

3.3.1 The Software Process

A software process has process-components that provide help and guidance to produce the models of problem, solution, and background space and the corresponding software. The quality process is made up of process-components that follow the actual software process-components like background shadows—providing the role descriptions, activities, tasks, and deliverables that are the end result of the activities carried out within the quality process.

Thus, an overall Quality Software Process includes process-components not only for development but also for quality assurance. A quality process is not necessarily a totally independent process but can be comprised of a set of process-components that are configured depending on the needs of the project to ensure the quality of the actual software

development process. Each process-component within the software process is subject to the criteria of necessity and sufficiency—with the third criterion of malleability being applied in the configuration of the process-component itself. A necessary checklist can be applied to the process-components, followed by the sufficiency checks, to ensure that the basic steps or tasks within a process not only have been followed but they have also been followed in the best possible way and with sufficient depth. Finally, the malleability of the process is provided by the process feedback loop and the ability of the process to change itself, not only for the next project but also during the project's execution.

3.3.2 The Quality Process

While the *software process* described above is a generic term that deals with the process of development (or integration or data warehousing, and so on), the term *quality process* describes the subprocess that accompanies the software process. This quality process is made up of the quality management, quality assurance, and quality control process-components discussed later in this chapter.

3.3.3 Rigorous Process

The extent to which a process is enforced depends on a number of factors. For example, a large project will have sufficient resources and process management skills to follow the iterative and incremental aspect of the process in great detail. In situations where the budgets are tight, or the skills are lacking, projects can still try to follow the process but the rigorousness with which they follow the process (sufficiency or depth) will be much reduced. Understanding the necessary and sufficient aspect of a process is extremely beneficial in balancing the rigor of the process.

3.3.4 Process Maturity

Judging the quality of process requires measuring its maturity. The Capability Maturity Model is the most popular and most accepted form of measurement of process maturity and subsequent quality. The CMM levels and their applicability in measuring processes, as discussed in the previous chapter, are crucial when measuring the maturity of the QSP. The process discussion in this chapter is aimed at helping organizations climb up the CMM maturity scale.

Furthermore, it is also worth considering CMMi as an integrated measure of the process. For example, in describing the six different project types for UML-based projects, we consider Internet-based development or e-development as a default. For each of the UML project types, the process-components are still described in a common format here, as process-components for development. This is because these process-components are relevant to new development as well as to most other project types. Variations to these process-components are permitted before the project starts, especially if we have a vastly different project type. Some changes to the way the process-components are executed should also be allowed. This is the malleability aspect of a process, as discussed next.

3.3.5 Malleable Process

We can judge the quality or the value of a process by its ability to configure these process-components. This is what I have called malleability of the process (see Section 1.8.3). Consider the differences of various UML-based projects in terms of their process requirements. For example, a straight development project uses certain process-components that directly provide development support. In that project we configure a set of process-components together and create a Software Engineering Process. That SEP is an instantiation of a process for new development. Once that SEP is created, we will still need to change it, particularly during the initial iteration of the process. This is malleability. It is an aspect of process not yet fully discussed within CMM.

3.3.6 Process Timing

Furthermore, note that a quality process or process-components that deal with quality first deal with the goals of an organization, followed by the goals of the project. Therefore, many times quality process-components in both instantiation and their enactment may start before a project starts. There are also some management-related process-components, which are included in most software processes. This management-related work usually starts before the project begins. These can be process-components that deal with the investigation, comparison, and analysis of the problem scenario to make the "Go/No Go" decision. They include numerous aspects related to cost, budget, marketing, and sales. The chance of identifying newer opportunities also comes into the picture given these process steps. Note once again these steps have to be taken before the project begins.

3.4 The Software Process

In this area we discuss the process-components that are responsible for software development. The other area containing three process-components is the quality process. (Please refer to the accompanying CD for a tabular list of the following process-components and a starting project plan based on them.)

3.4.1 Business Evaluation Process-Component

Figure 3.4 describes the process-component of business evaluation that describes the part of the process that deals with the prime reason why the project exists. This process-component also presents the very early approach

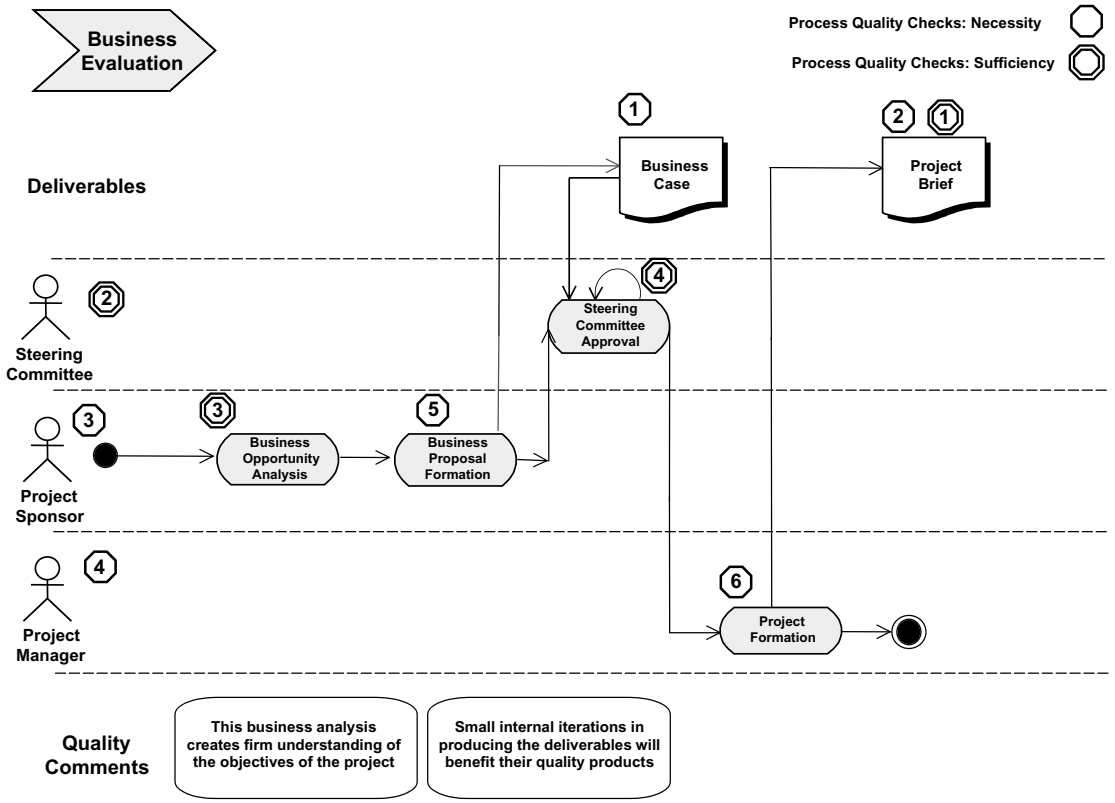


Figure 3.4 *Process-component for business evaluation*

to organizing the project. This is undertaken by the project sponsor, who starts with the activity of a business opportunity analysis. This is followed by a business proposal formulation. The steering committee performs the activity of project approval in an iterative fashion; the project manager handles the responsibilities of the project once the project is formed.

This process-component is important when undertaking a formal evaluation of the business reasons why a software project should proceed. Not only does this argument help those who are going to manage the project, but it also helps to confirm, in the minds of the project sponsor and the eventual end users, the goals and objectives of the project. Because this process-component deals with the initial understanding of the business problem, the deliverables produced here form part of the Model Of the Problem Space (MOPS). They are the *business case* and the *project brief*.

3.4.2 Roles in Business Evaluation

The roles in business evaluation are as follows:

- The steering committee deals with the highest-level management of the project. It brings together expertise from various domains including technology, business, human resources, accounting, finance, and legal, to name a few.
- The project sponsor initiates the project, benefits from it, and pays for it. The project sponsor is the chief person who must be satisfied with the quality of the end product. Therefore, the project sponsor (also known as the business sponsor) is the one who should be involved in documenting the project objectives, identifying the risks, and establishing the priorities.
- The project manager is responsible for the project once it has been evaluated and approved by the business. Therefore, in a way, the project manager's goal is more operational in nature than strategic.

3.4.3 Activities and Tasks in Business Evaluation

Figure 3.5 describes the activity-task mapping within the process-component of business evaluation. This activity-task mapping is also available on the accompanying CD for creating a project plan based on this mapping. The activities and tasks of this process-component play an important role in evaluating the business case and ensuring the project objectives within the business context.

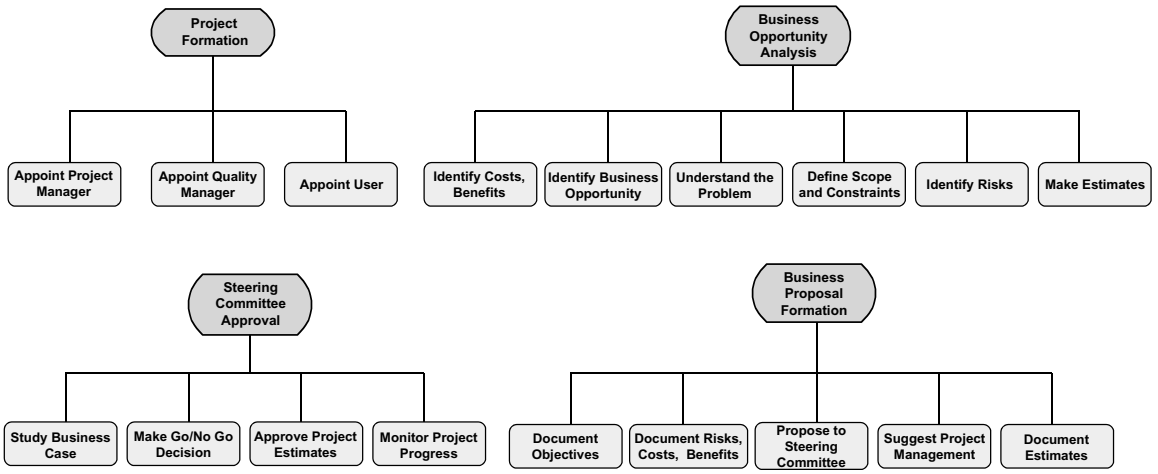


Figure 3.5 *Activities and tasks in business evaluation*

3.4.4 Deliverables in Business Evaluation

- Business case. The business case deliverable contains the arguments for why the project should go ahead. It also documents the cost-benefit analysis for the project. The scope of the project, its risks, and the related project estimates are all documented in the business case. This is the document presented to the steering committee.
- Project brief. Typically, the project brief is a two-page document describing the project; it outlines the objectives of the project, the relevant project numbers, and the project schedule. The project manager and the quality manager are named in the project brief.

3.4.5 Quality Comments on Business Evaluation

Necessity

1. The business case should be carefully prepared to justify the need for a project. This document ensures that the purpose of the project is clear to all parties involved.
2. The project brief is the second mandatory part of this process-component, ensuring that the details of the project are succinctly summarized and available for reference throughout the project.

The project brief also describes the type and size of the UML-based project.

3. The project sponsor is the main role in this process-component, ensuring that the business opportunities are properly analyzed and documented, and that the business proposal has considered all options.
4. The project formation is the starting point of the project manager's project responsibilities.
5. The activity of formulating the business proposal is crucial to the process-component of business evaluation. Results from this activity formally complete the deliverable of the business case.
6. The activity of project formation, through the tasks of the appointments of project manager and quality manager, literally *forms* the project.

Sufficiency

1. The project brief may not be produced in just one iteration. It will, perhaps, need to be updated based on the discussions during the steering committee approval. However, having the project brief deliverable provides the sufficiency, in terms of quality, in this process-component.
2. The steering committee also provides the criteria of sufficiency for this process-component. In small projects, the steering committee may play a notional role, but for medium and large projects, the committee brings in expertise from varying domains within the organization. The steering committee can also help to categorize the project into its respective type (for example, data warehousing, integration, or outsourcing). Having a steering committee for the project satisfies the sufficiency criteria of quality.
3. Business opportunity analysis provides the sufficiency in rigorously analyzing and questioning the need for the project.
4. Steering committee approval will easily iterate twice, if not more, before it provides sufficient rigor in deciding about the project. The steering committee members may ask the project sponsor to further investigate, collect, and collate the information—and only then will they approve the project.

3.4.6 Project Management Process-Component

The project management process-component deals with all activities and tasks that are carried out (primarily by the project manager) in managing the project. Project management is an extremely important process-component that includes understanding the technology, methodology, and sociology of the project. Therefore, this process-component interacts with the process-component of process configuration. The primary purpose of project management is to organize and track the project. Tracking involves risk management; therefore, project management also deals with

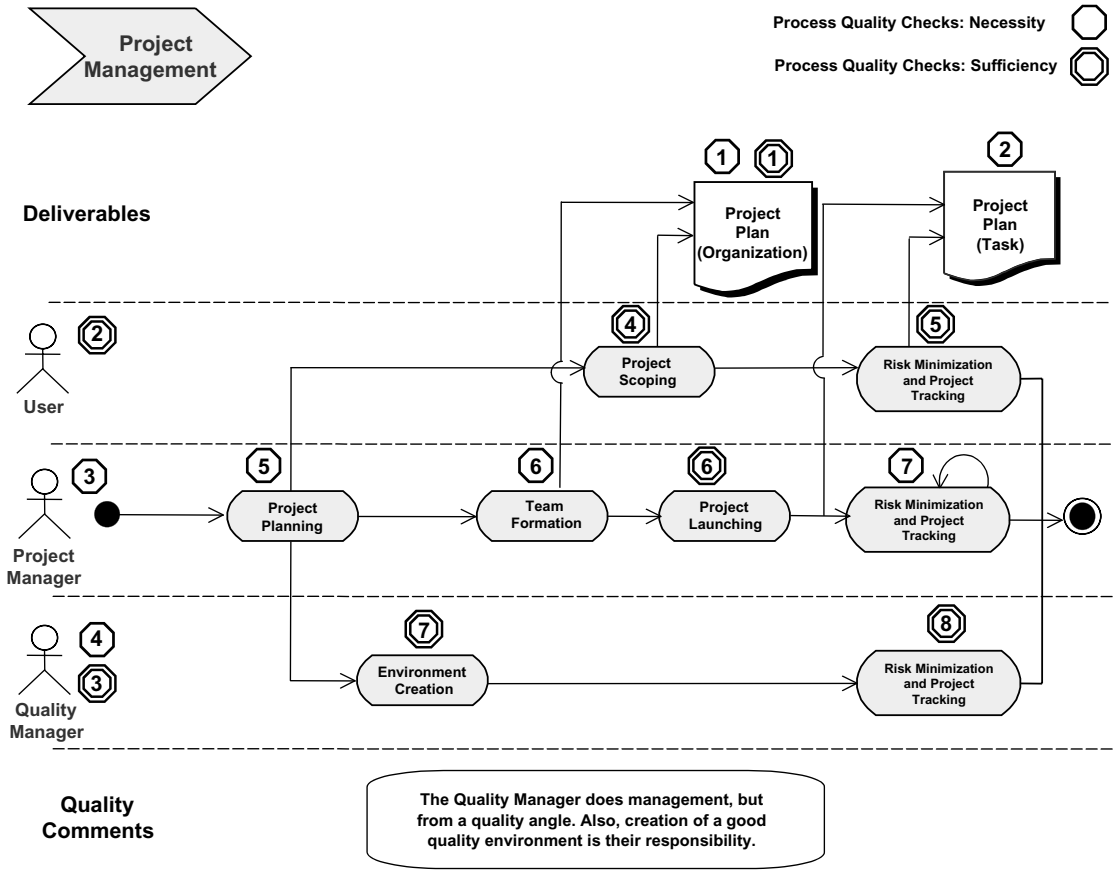


Figure 3.6 *Process-component for project management*

risks and their prioritization. Project management provides feedback on the status of the project to various stakeholders including the user, the steering committee, and the project sponsor.

3.4.7 Roles in Project Management

- The project manager is the primary role in this process-component of project management. This role is responsible for dealing with the activities of planning for the project, forming teams, launching the project, and continuously monitoring the risks.
- The quality manager accompanies the project manager when performing her responsibilities but, at the same time, provides the independent crosscheck on the activities and tasks performed by the project manager.
- The user continues to help the project manager by scoping the project, prioritizing the risks, and helping to minimize them.

3.4.8 Activities and Tasks in Project Management

Figure 3.7 shows the mapping of activities to tasks in the process-component of project management, which helps to create a good quality project plan and the management of the overall project.

3.4.9 Deliverables in Project Management

- Project plan (organizational). The organizational project plan provides the detailed description of the category and type of project, its resources, and its approach to quality.
- Project task plan. The project task plan is a task list with corresponding resources assigned to it.

3.4.10 Quality Comments on Project Management

Necessity

1. The organizational project plan descriptively documents the planning process for the project. It is necessary to cross check its accuracy with the users, the project sponsors, and the steering committee.

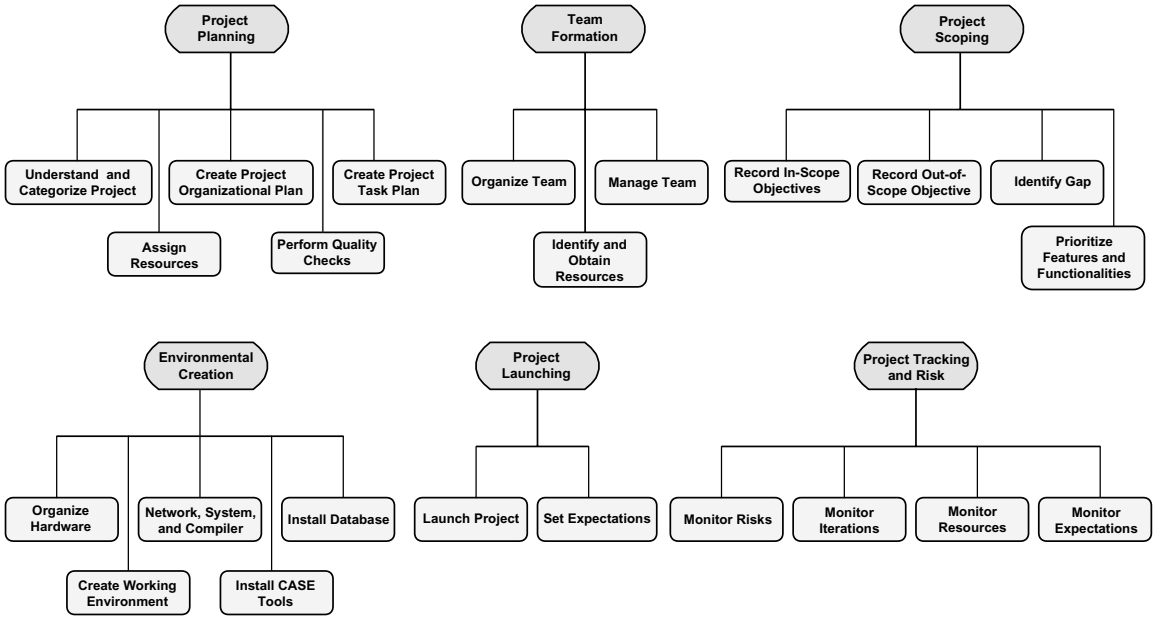


Figure 3.7 *Activities and tasks in project management*

2. The project task plan lists the tasks to be performed in the project and maps them to the corresponding resources. This is a necessary part of the process and requires an inspection to verify its quality.
3. The project manager, as the primary role in this process-component, should be checked for the correctness of the role description and the ability of the person playing this role to fulfill it.
4. The quality manager is necessary for the quality of the overall project. Creating a good quality environment is the responsibility of this role.
5. Project planning is a necessary part of the process-component, resulting in an organizational plan.
6. The activity of team formation should be checked to ensure that it provides all necessary guidelines in identifying team members and their formation into the right team. This is followed by ongoing management of the team.

7. Risk minimization and project tracking is an ongoing activity that is performed, in parallel, by all three major roles in this process-component. The project task plan is updated and fine-tuned with this activity.

Sufficiency

1. The organizational project plan should be updated with the soft issues related to team formation, to satisfy the criteria for sufficiency.
2. The user, another sufficiency criterion, is preferably onboard, as a part of the project team, to ensure the quality of project scope and risk minimization.
3. The quality manager not only performs the activities of environment creation and risk minimization as necessary, but also organizes and performs the quality tasks of inspection, walkthroughs, and reviews (described in the quality process-components), to ensure sufficient checking.
4. Project scoping is performed more than once to satisfy the sufficiency criteria.
5. Risk minimization, a responsibility of the user, will have to be performed to provide sufficient quality for the project.
6. Project launching is a formal activity in a large project, but may not be necessary in small projects.

3.4.11 Process Configuration Process-Component

Figure 3.8 shows the process-component of process configuration, which deals with understanding the process-related needs of the project, putting the process together, and later, enacting it in practice. If an organization is already process based, and the project is provided with a configured process to follow, then there is no need to undertake an extensive process survey. Initiating a process, customizing it, deploying it, and transitioning the people and the current style of working to the process style are critical for quality. Not only does this help to jump-start a situation where people working on the project do not spend or waste a lot of time deciding what to do, it also puts rules, standards, and templates in place that ensure the necessary steps within the process to produce the models in the various modeling spaces. Process configuration is also important in an iterative and

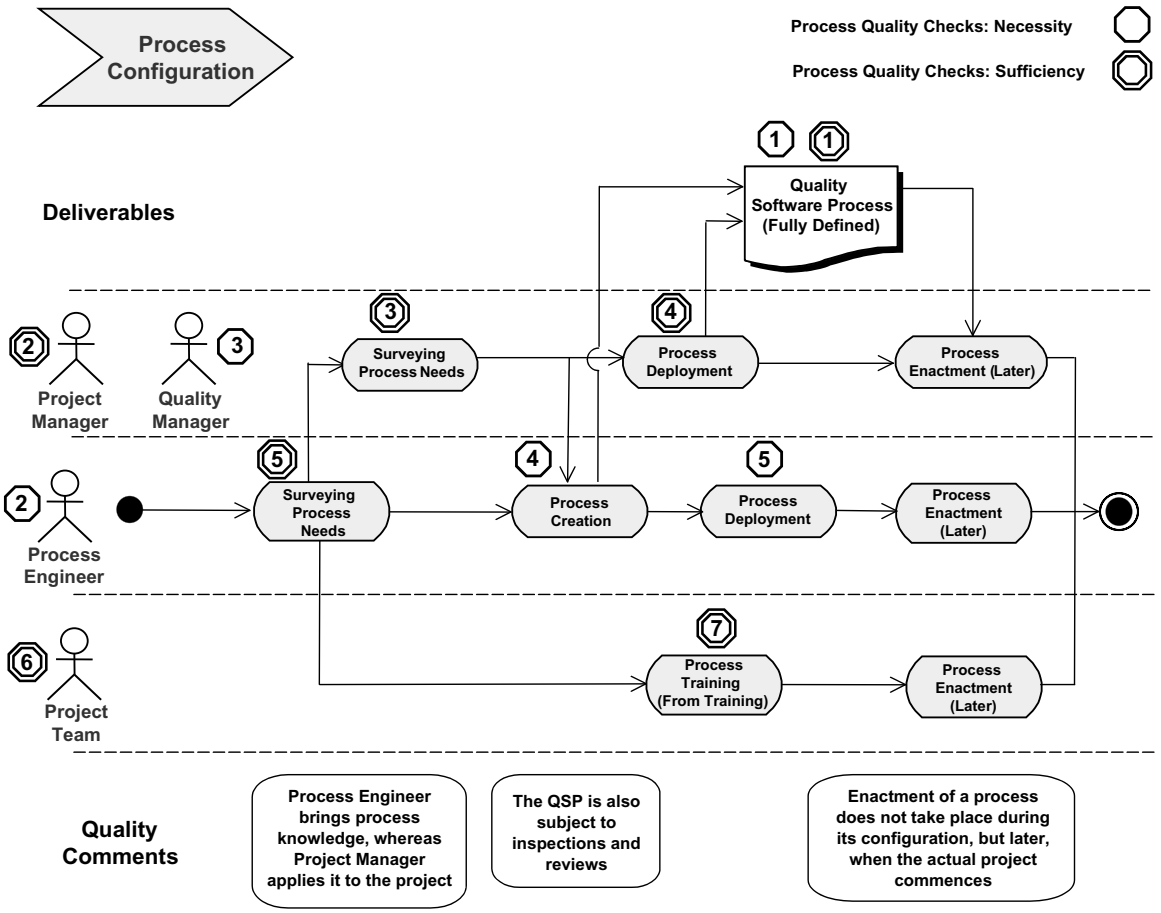


Figure 3.8 *Process-component for process configuration*

implemental process because the need to identify the extent of iterations and the things that should be included in the increments are all important to the success of the quality of the project deliverables.

3.4.12 Roles in Process Configuration

- The process engineer is the primary role in this process-component, responsible for the important activities of surveying the needs of a process within the project and creating a suitable SEP to satisfy it. It is

not necessary for this role to be a permanent role in the project, because once the SEP is created by this role, the project manager is able to enact it. However, large process-conscious projects that need to handle the malleability of the process benefit by having this role on a long-term basis.

- The project manager and the quality manager facilitate the work of the process engineer by providing the necessary needs of the process, and later deploying and enacting the process.
- The project team has to understand and follow the process. They also have to provide feedback for the process, to enable its change and fine-tuning.

3.4.13 Activities and Tasks in Process Configuration

Figure 3.9 describes the activities and tasks of the process-component of process configuration.

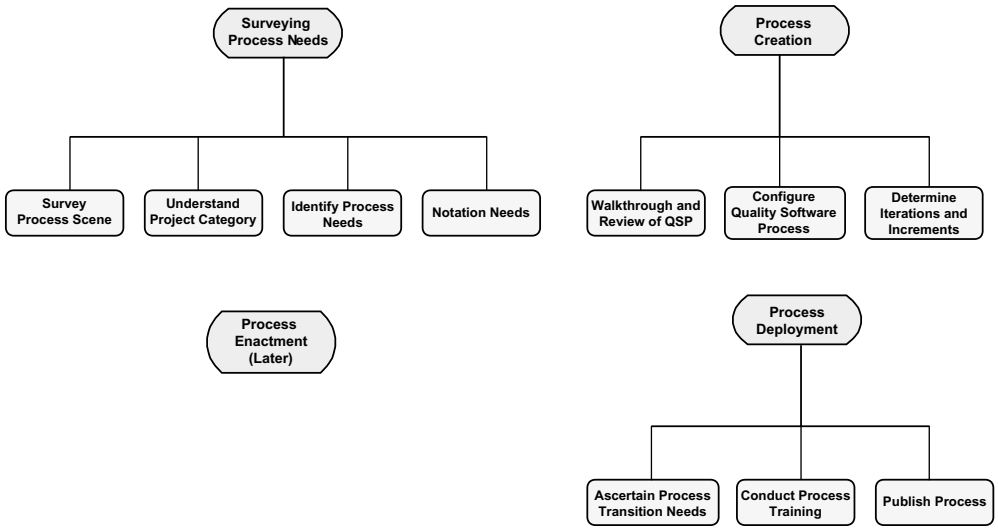


Figure 3.9 *Activities and tasks in process configuration*

3.4.14 Deliverables in Process Configuration

- The Quality Software Process is the repository of all process-components in the process. The QSP is configured to create an instance, which is the SEP to be followed in the project.

3.4.15 Quality Comments on Process Configuration

Necessity

1. The Quality Software Process with definitions of all process-components is the basic necessity of this process-component. The QSP itself is subject to quality checks, ensuring that it covers all areas of development.
2. The process engineer, also called the process consultant (due to the temporary nature of this role), is necessary for the process.
3. The quality manager focuses on the quality aspect of the process and the process aspect of quality—both of which are necessary for a good quality product.
4. Process creation is the activity of putting together a process, based on the process-components defined here.
5. Process deployment is sending the fully configured process out in the project, for use.

Sufficiency

1. The QSP is checked, rechecked, and brought to a level where it is acceptable to all stakeholders in the project, especially the project team.
2. The project manager provides the supporting role to the quality manager in organizing the process and ensuring it is the correct process for the project on hand.
3. Surveying the needs of a process is significant in a large, high-ceremony project.
4. The responsibilities of the process deployment, as handled by the project manager and quality manager, provide the sufficiency criteria to send the process out in the project.
5. Surveying the process needs by the process engineer is also part of a large, high-ceremony project. It is not necessary, in small projects, to undergo the detailed rigors of process surveys.

6. The project team must undergo sufficient training and should have the necessary buy-in, to derive the advantage of process deployment.
7. Process training is sufficient criteria, from the training process-component, to configure and deploy a process.

3.4.16 Requirements Modeling Process-Component

Figure 3.10 shows the requirement modeling process-component that deals with the actual capture, engineering, and analysis of the requirements of the business in the project. This process-component uses the primary mechanisms of use case modeling and activity diagramming to

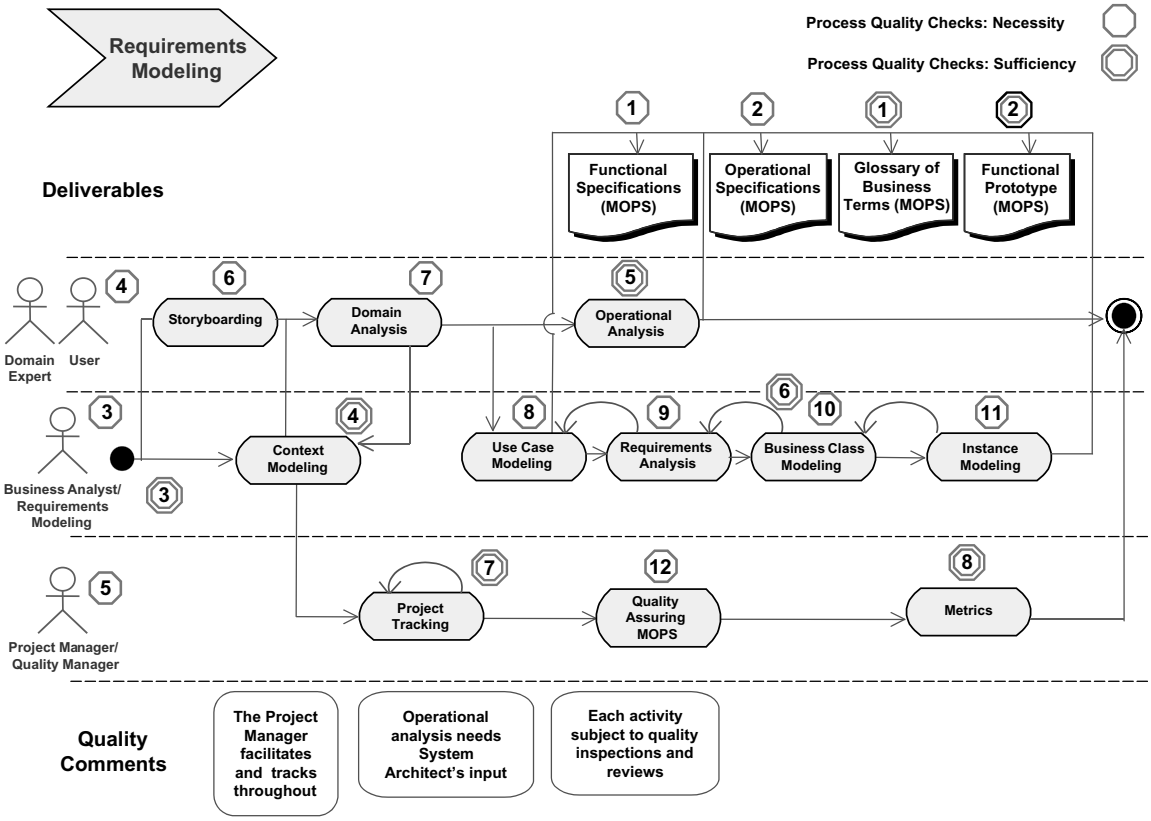


Figure 3.10 *Process-component for requirements modeling*

capture and document the problem that the business is trying to solve. Because quality is such a subjective phenomenon, it is absolutely crucial that those who will ascertain the quality (users and project sponsors) are involved as extensively as possible in this process-component.

The quality techniques of interviews and workshops are very helpful in executing this process-component. In addition to documenting the functional requirements in the problem space, this process component also encourages the user to provide the operational needs of the system. The user of the system or the businessperson who is involved in the project is ideally placed to provide the information on the expected volume, performance, and security needs of the system from an operational or nonfunctional viewpoint. This is all documented as a result of requirements modeling. Prototyping is also used in order to extract further requirements and refine the requirements already captured.

3.4.17 Roles in Requirements Modeling

- The business analyst (also called the requirements modeler for most of this process-component) is the primary role here, and is responsible for understanding and documenting the requirements.
- The domain expert and the user provide the information that the business analyst is trying to document.
- The project manager, together with the quality manager, facilitates the process of requirements modeling. They also monitor and track the progress of the requirements-modeling exercise and report to the steering committee on this crucial process-component.

3.4.18 Activities and Tasks in Requirements Modeling

Figure 3.11 describes the activities and tasks of the process-component of requirements modeling. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.4.19 Deliverables in Requirements Modeling

Functional specifications containing the use cases and activity diagrams that make up the model of the problem space are the main deliverables coming out of this process-component. Additional UML-based diagrams, namely the class diagrams, sequence diagrams, and state chart diagrams,

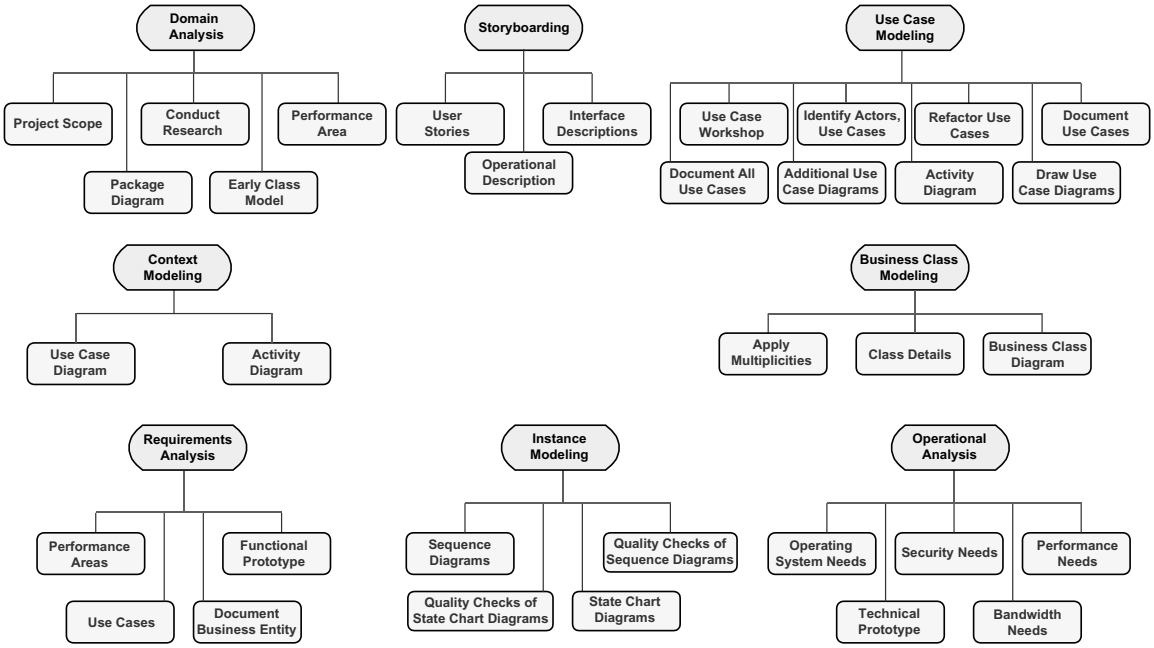


Figure 3.11 *Activities and tasks in requirements modeling*

also form part of the functional specifications. While a part of the model will usually be put in a CASE tool, some specifications (such as the use case documentation or the class responsibilities arrived at using the CRC techniques) may be outside the CASE tool.

Operational specifications document the requirements of the system when it goes out in operation. Therefore, these requirements are stored mostly in a document that describes the stress, volume, and performance requirements. Additionally, these requirements are provided using the deployment diagrams of the UML.

A glossary of business terms helps to record confusing or important terms. For example, terms like cover, policy, and insurance may mean the same thing for most people, but may have interpretation variations for the insurance domain modelers.

Functional prototype, as mentioned in the discussions on prototype, enables the extraction of complete and correct requirements.

3.4.20 Quality Comments on Requirements Modeling

Necessity

1. Functional specifications are a necessary deliverable of the process-component and should be produced iteratively with quality checks being applied to the diagrams inside them.
2. Operational specifications are an equally important and necessary part of this process-component. Without good quality operational specifications, the system may not succeed when it is deployed.
3. The business analyst is a necessary part of this process-component and must be checked for the accurateness of its role description and understanding of that description by the person performing the role. A business analyst is similar to the requirements modeler, except that the latter focuses only on creating the model (as opposed to the BA, who looks at the broader picture).
4. The domain expert and, more importantly, the user, are absolutely necessary in order to create a good requirements model. By participating in the MOPS creation process, they also firm up their own objectives and purposes for the project. This has a valuable quality connotation, as the user is eventually going to judge whether the product has quality or not.
5. The project manager, supported by the quality manager, provides the background organizational support for the process of requirements modeling.
6. Storyboarding is increasingly considered an important technique in discovering the correct and complete requirements with substantial participation from the user.
7. Domain analysis, particularly with the help of the domain expert, provides a much broader view of the requirements—not limiting them to a single project. This is very valuable in a reuse program.
8. Use case modeling, by far the most revolutionary approach to requirements modeling, is primarily performed by the business analyst—with considerable input provided by the user/domain expert. This is a necessary activity and should be performed iteratively to produce a good suite of use cases and use case diagrams, together with the activity diagrams.

9. Requirements analysis is necessary to understand whatever has been documented in the use cases, and to extract correct business entities from that documentation in order to produce business class diagrams.
10. Business class modeling, also occasionally known as business domain modeling or business object modeling, is the creation of class diagrams at the business level, and should be checked as a necessary step in this process-component.
11. Instance modeling is necessary to correctly identify the way in which instances like objects on sequence diagrams or state charts behave. Documenting this behavioral aspect of this process-component is vital for quality MOPS.
12. Quality assurance of the entire MOPS, by conducting extensive inspections and reviews, is a necessary quality check, and should be performed following the quality techniques described in Chapter 6.

Sufficiency

1. The glossary of business terms is very helpful, if produced in a project where the team members are new to the domain.
2. A functional prototype, although not necessary for every project, is sufficient for a good quality requirements modeling exercise.
3. The roles of business analyst and requirements modeler should also undergo process sufficiency checks to ensure that they are properly defined, are staffed in the right numbers, and have sufficient understanding of the uncertainties around this process-component.
4. Context modeling, through the use of case diagrams at an abstract level with a boundary and activity diagrams at an abstract level, can provide the context of the system. Performing this activity satisfies the sufficiency criteria of quality.
5. Operational analysis should be performed with sufficient depth—otherwise, the operational specifications will end up getting produced through the other activities of domain analysis and use case modeling—not an ideal situation.
6. The business class modeling should be iterated sufficiently before a good class model at business level emerges.

7. Project tracking, an ongoing activity, satisfies the sufficiency criteria of a quality process.
8. Measurements and metrics provide the additional benefits of process maturity to a project, if desired.

3.4.21 Interface Modeling and Design Process-Component

The process-component of interface modeling provides guidance for modeling and designing the important interface aspect in a system—the GUI, printing interface, and interfaces to other devices. The GUI and

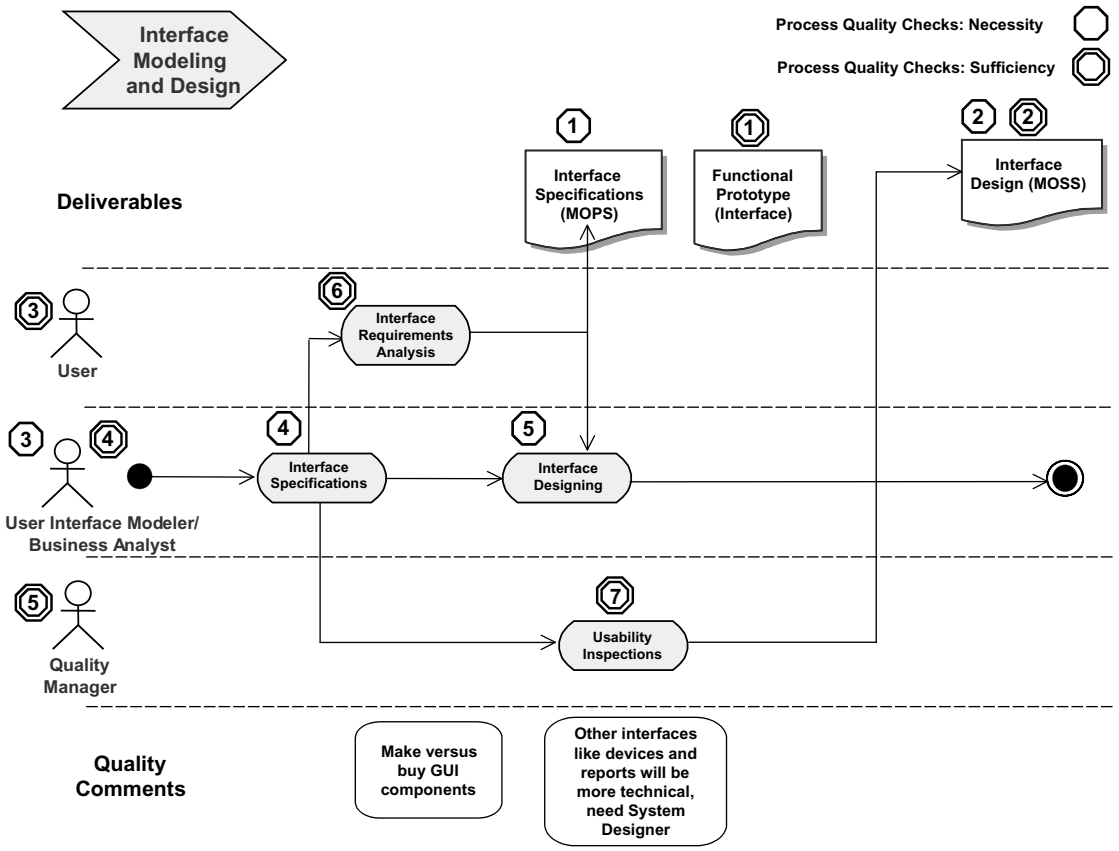


Figure 3.12 *Process-component for interface modeling and design*

printer interfaces are essential, not only for the performance of the system, but also for the way it is perceived by the users. In all e-business applications, this particular process-component takes an even more crucial role. It is not enough to provide just a good-looking interface; it should be designed as a system on its own—with sufficient discussion and modeling on the navigational aspects of the interface. Availability of the interfaces across geographical and time boundaries is vital. Issues of language and notation (as in Chinese, Hindi, or French and corresponding cultural notations) relating to widely dispersed user groups are important in Web interfaces. Furthermore, providing feedback to the user through legible messages, sound, and related mechanisms is vital for a good user interface. Thus, the interface modeling process-component should encompass prototyping, navigation diagrams, site maps, sketching, play acting, and other appropriate techniques to create a quality-conscious interface.

3.4.22 Roles in Interface Modeling

- The user interface (UI) modeler and the business analyst play the primary roles in this process-component; the BA specifies the requirements of the interface and the UI modeler designs it.
- The involvement of the user in interface modeling and design is of interest both to the user and the UI modelers. Quality perception is positively affected by the involvement of the user in this process-component.
- The role of quality manager facilitates the activities and tasks within the process-component.

3.4.23 Activities and Tasks in Interface Modeling

Figure 3.13 describes the activities and tasks of the process-component of interface modeling. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.4.24 Deliverables in Interface Modeling

- Interface specifications document the requirements of the interface and are mainly driven by the needs and desires of the user.

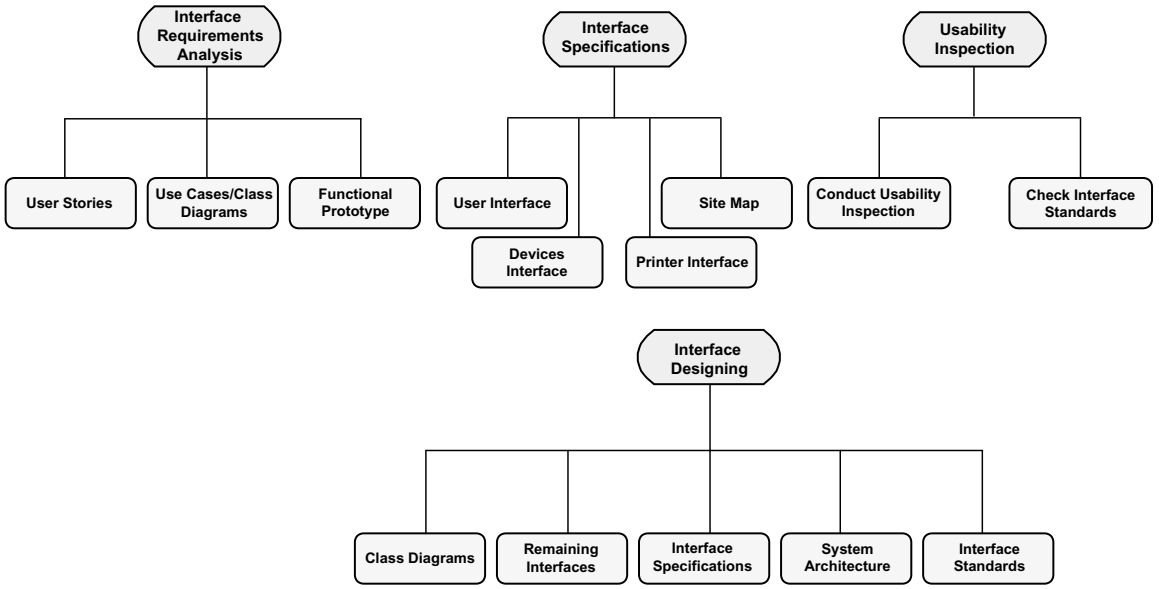


Figure 3.13 *Activities and tasks in interface modeling and design*

- Interface design is a deliverable containing details of the design that implement the GUI. It contains solution-level class diagrams that also include reusable user interface libraries and so forth.
- Functional prototype (interface) optionally adds value to the interface specifications and designs in this process-component.

3.4.25 Quality Comments on Interface Modeling

Necessity

1. Interface specifications are a necessary part of this process-component. The quality of the final interface depends on specifying, in detail, what users want and how they want it. Therefore, this specification document relates to the use case specifications, associated use case diagrams, and activity diagrams.
2. Interface design is a deliverable produced in the solution space. It is the design of the interface and a system on its own. Therefore, it contains all the necessary checks for a good design, including

checking the class diagrams that mainly have their stereotypes as `interface` or `boundary`.

3. The user interface modeler and the business analyst are necessary parts of this process-component and should be checked for their understanding of the requirements from the user's viewpoint. Therefore, a good BA and a good UI modeler work iteratively, by showing the interface progressively to the user and getting their feedback before proceeding with more designs.
4. Interface specifications are necessary before a good design can be produced. They are also checked for their ability to satisfy the requirements described in the use cases.
5. Interface designing is the activity that ensures that the results of the interface specification activity are taken down close to low-level design, wherein they can be easily implemented.

Sufficiency

1. The functional prototype provides additional criteria for quality, as it enables better documentation of the interface.
2. The interface design is also sufficiency criteria. It is possible to jump directly from interface specifications straight to implementation, but it's not advisable. Therefore, the interface design is provided as sufficiency criteria, as well.
3. The user should be sufficiently involved in the interface specifications by analyzing the interface requirements.
4. The business analyst, in particular, provides the quality criteria for sufficiency by providing the link between the modeling done by the UI modeler and relating it to the user.
5. The quality manager is involved in this process-component by providing sufficient support and coordination—especially when usability inspections are formally conducted.
6. Interface requirements analysis provides the background to perform the user stories, the use cases, and the functional prototypes, in order to understand the interface requirements fully before designing them.
7. Usability inspections are along the lines of Constantine's Collaborative Usability Inspections (CUIs) and provide a formal review of interface quality, in the presence of the user and the developers.

3.4.26 System Design Process-Component

The process-component for system design produces the system design deliverable. The system design deliverable has the low-level designs that contain classes and class definitions that deal with the language of implementation and the databases. Solution-level design includes solution-level class modeling and instance modeling.

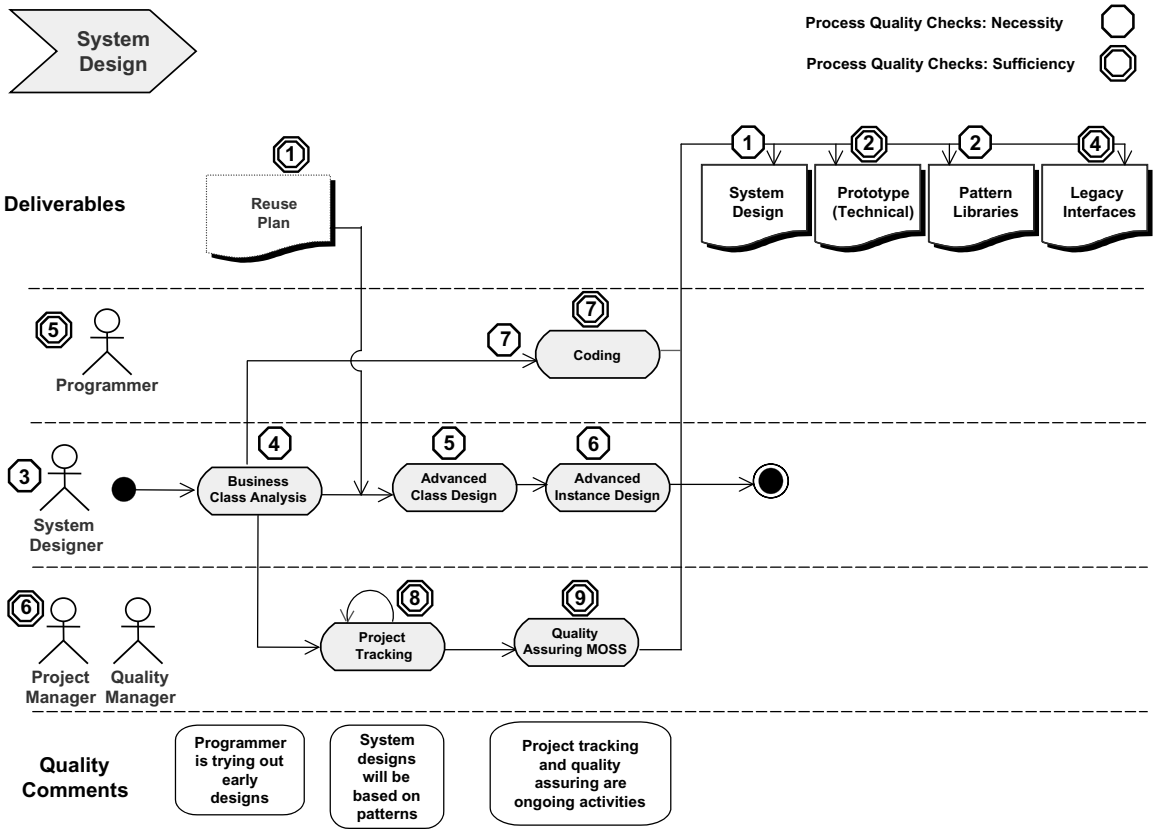


Figure 3.14 *Process-component for system design*

3.4.27 Roles in System Design

- The system designer is the main role in this process-component. The system designer must have enough information about the implementation language and the environment for implementation to perform this role successfully. Quality in this process-component comes not only from the experience of the designer but also from his/her technological knowledge.
- The programmer continues to assist the system designer, checking on the feasibility of the designs by trying them out in code.
- Project manager/quality manager

3.4.28 Activities and Tasks in System Design

Figure 3.15 describes the activities and tasks of the process-component of system design. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

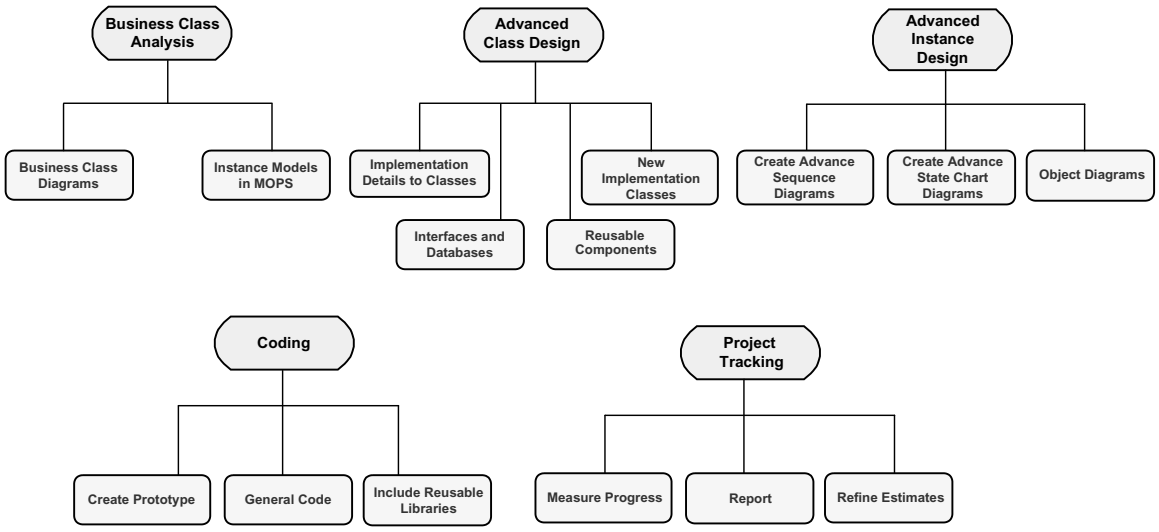


Figure 3.15 *Activities and tasks in system design*

3.4.29 Deliverables in System Design

- The system design contains the details of the technical design that form part of MOSS.
- The prototype (technical) is a code-level prototype that helps when trying out a few code examples.
- The pattern libraries may be directly inserted in the system designs or may be newly created for local project or organizational-level patterns.
- The legacy interfaces (especially in integration projects) are considered at the design level. We think of all the necessary issues of implementation, including legacy interfaces, before a system can be fully implemented.
- The reuse plan (input), if produced, provides the background support for reuse in the designs.

3.4.30 Quality Comments on System Design

Necessity

1. System design is a necessary part of the system design process-component and should undergo the quality checks using the techniques of interviews and reviews in workshop format.
2. Using the pattern library, or at least giving patterns serious consideration, is necessary for good quality.
3. The system designer is the primary owner of this process-component. The role should be well-defined and understood by the person performing it.
4. Business class analysis requires understanding the class diagrams drawn by the BA in the requirements modeling process-component. Therefore, this is a necessary starting point for any work that takes place in the design.
5. Advanced class design, as shown in Chapter 4, is necessary for this process-component. It deals with creating classes that are very close to implementation.
6. Advanced instance design is a necessary step in system design because the sequence and state chart diagrams ensure that the

classes drawn in the class diagrams are complete and correct. This is done by cross checking the sequence and state chart diagrams with the classes in the class diagrams.

7. The coding activity provides the necessary support for creating the prototype and verifying the designs. It may be undertaken quickly by the system designer herself.

Sufficiency

1. Check the availability of a reuse plan and incorporate the suggestions and standards from the plan into the design.
2. Check to determine if the technical prototype is necessary—it should be created to test out the validity of the designs and will satisfy the sufficiency criteria.
3. Pattern libraries should also be sufficiently considered in the system designs.
4. Legacy interfaces, for a legacy integration project, are created to satisfy the sufficiency criteria.
5. The programmer, if available, is sufficient for the quality system designs. If not, the cursory programming work is carried out by a system designer.
6. The project manager/quality manager provides the background support and coordination activities.
7. Coding may be attempted a few times, in creating prototypes, reusing libraries, or simply creating earlier cuts of the code.
8. Project tracking is an ongoing activity, which may be undertaken at the completion of major activities in this process-component.
9. Quality assurance of MOSS will have to be performed a few times, iteratively, to satisfy the sufficiency requirements of the project.

3.4.31 Persistence Design Process-Component

Persistence design, another term to describe database design, is treated as a separate process-component because of its importance—not only in storing standard relational data, but also in storing a variety of content in today's Web applications. The need to interface with existing legacy systems, store and massage audio and video contents, convert data, and pro-

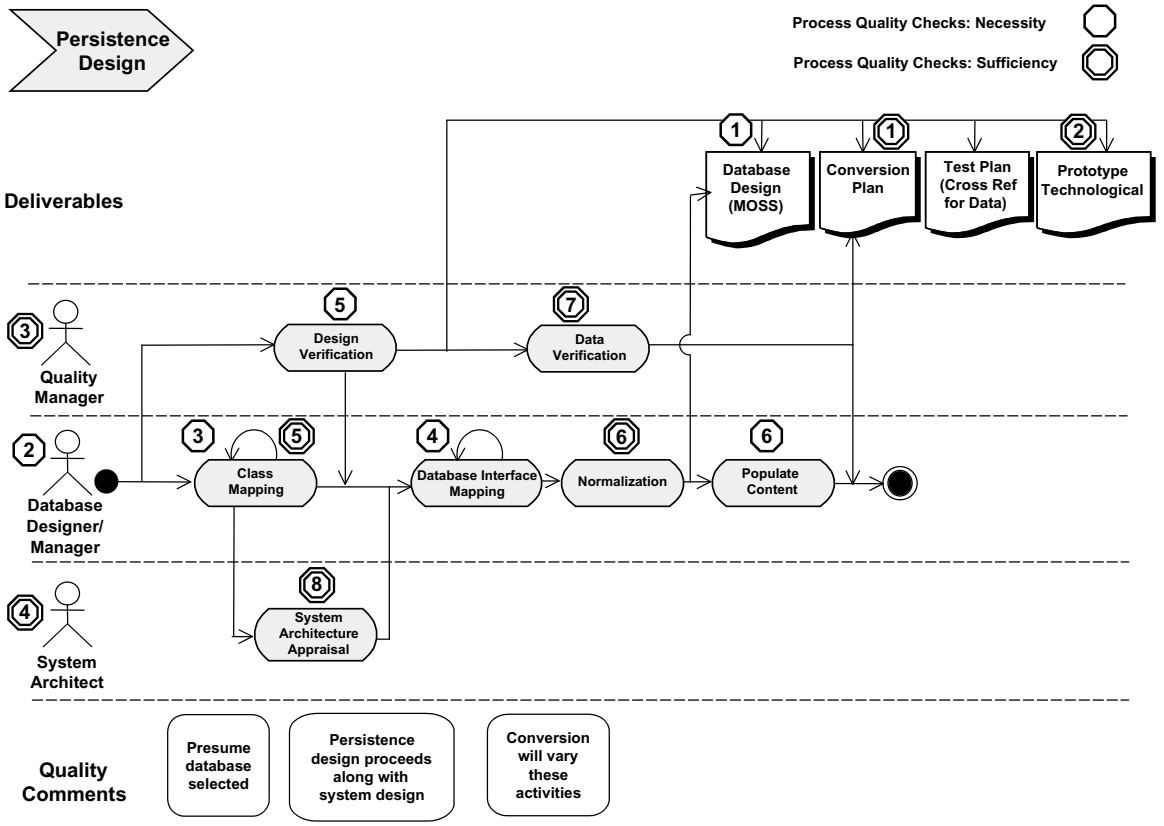


Figure 3.16 *Process-component for persistence design*

vide reuse and multidimensional drilling into the data are some of the reasons database design is important and is treated as a separate process-component.

While using the capabilities of the UML to represent the databases in class diagrams, it is also important to use sequence diagramming techniques in order to document the access to the databases, the security to the databases, and the consistency requirements. Prototyping from an operational viewpoint can also provide valuable information during the database-design phase.

3.4.32 Roles in Persistence Design

- The database designer/manager creates database schemas, strategies for conversions, and population of data.
- The quality manager facilitates the environment for the creation of the databases and ensures the quality of the schemas created. Thoughts are also given to the population of the databases, especially if data is to be converted from an existing system.
- The system architect provides operational input.

3.4.33 Activities and Tasks in Persistence Design

Figure 3.17 describes the activities and tasks of the process-component of persistence design. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.4.34 Deliverables in Persistence Design

- Database design
- Conversion plan
- Prototype (technical)

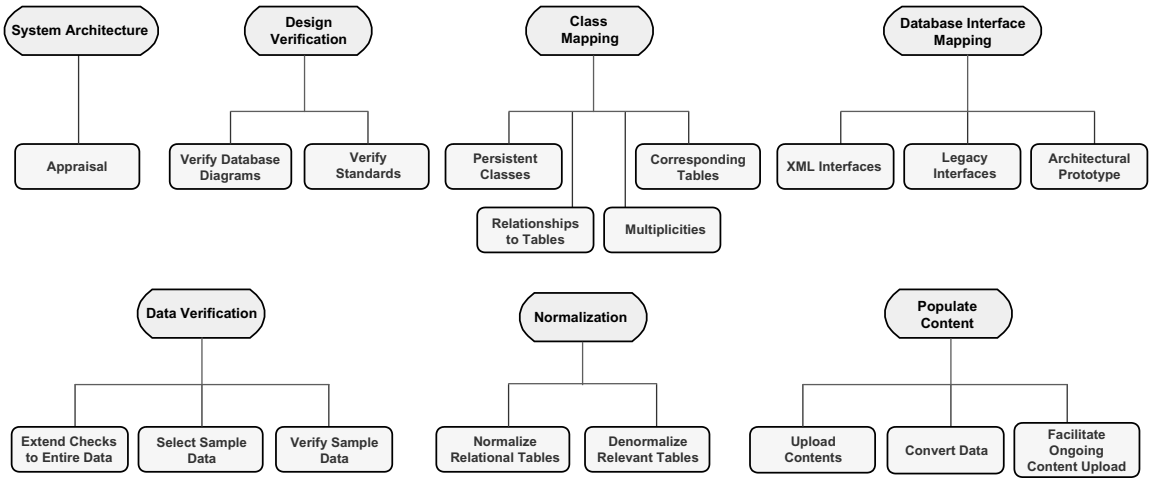


Figure 3.17 *Activities and tasks in persistence design*

3.4.35 Quality Comments on Persistence Design

Necessity

1. Create the database design based on the persistent classes from the MOSS class diagrams.
2. Ensure that the person playing the role of the database designer understands both the UML modeling techniques and the capabilities of the database in implementation. Conversion requires additional skills in understanding the structure of the existing database.
3. Class mappings require mapping the class diagrams in the solution space to the database tables. Relationships between classes in the class diagrams are translated to relationships between database tables using the primary and foreign keys of relational database designs. Multiplicities also provide additional and valuable information.
4. Database interfaces include interfaces to front-end Web interfaces (using, say, the XML), or back-end legacy interfaces. Prototypes are created to investigate these interfaces and produce designs.
5. Design verification is a formal activity, following the review techniques, to ensure the consistency of the new database schema.

Sufficiency

1. A conversion plan is required only when data has to be converted into the new system. Alternatively, for a Web application, contents might be required.
2. Prototypes provide sufficient details in creating a good database design—not only from the structural viewpoint, but also from the population and performance viewpoints.
3. The quality manager provides the organizational support and applies the quality techniques of inspections, reviews, and workshops to the designs.
4. The system architect ensures that the database design is in accordance with the overall system architecture.
5. Class mapping has to be double-checked for sufficient compatibility of the classes and the corresponding relational tables.

6. Normalization may be attempted in practice and will be influenced by the multiplicities in the class diagrams.
7. Data verification follows some of the testing techniques of equivalence partitioning and boundary value (as discussed in Chapter 6).
8. System architecture appraisal ensures that the database design does not transgress the architecture of the system (for example, the bandwidth limitations of the architecture will influence the database design).

3.4.36 Implementation Process-Component

The process-component for implementation deals with coding. While all other process-components deal with understanding the problem in managing the project, this one deals extensively with implementation of the models using the available technology. Thus, the designs created during the system design, database design, and interface modeling are implemented during enactment of this process-component. Implementation deals with understanding both the requirement models and the designs.

Before creating the actual code it is necessary to incorporate the reusable libraries (already done in the system design process-component). This is followed by creating the implementation classes and compiling, linking, building, and testing them. Testing includes the creation of test harnesses within the code itself, as well as conducting unit tests and stepping through the created code. Occasionally, if processes like eXtreme Programming are followed in the project, the results from the implementation effort can be immediately showed to the users with a very short turnaround time.

3.4.37 Roles in Implementation

- The programmer implements the designs iteratively. Many people play this role in a project; they interact with each other in the roles of programmer and, when the need arises, with other roles such as system designer and business analyst.
- The system designer supports the programmer by explaining the designs created in the MOSS.

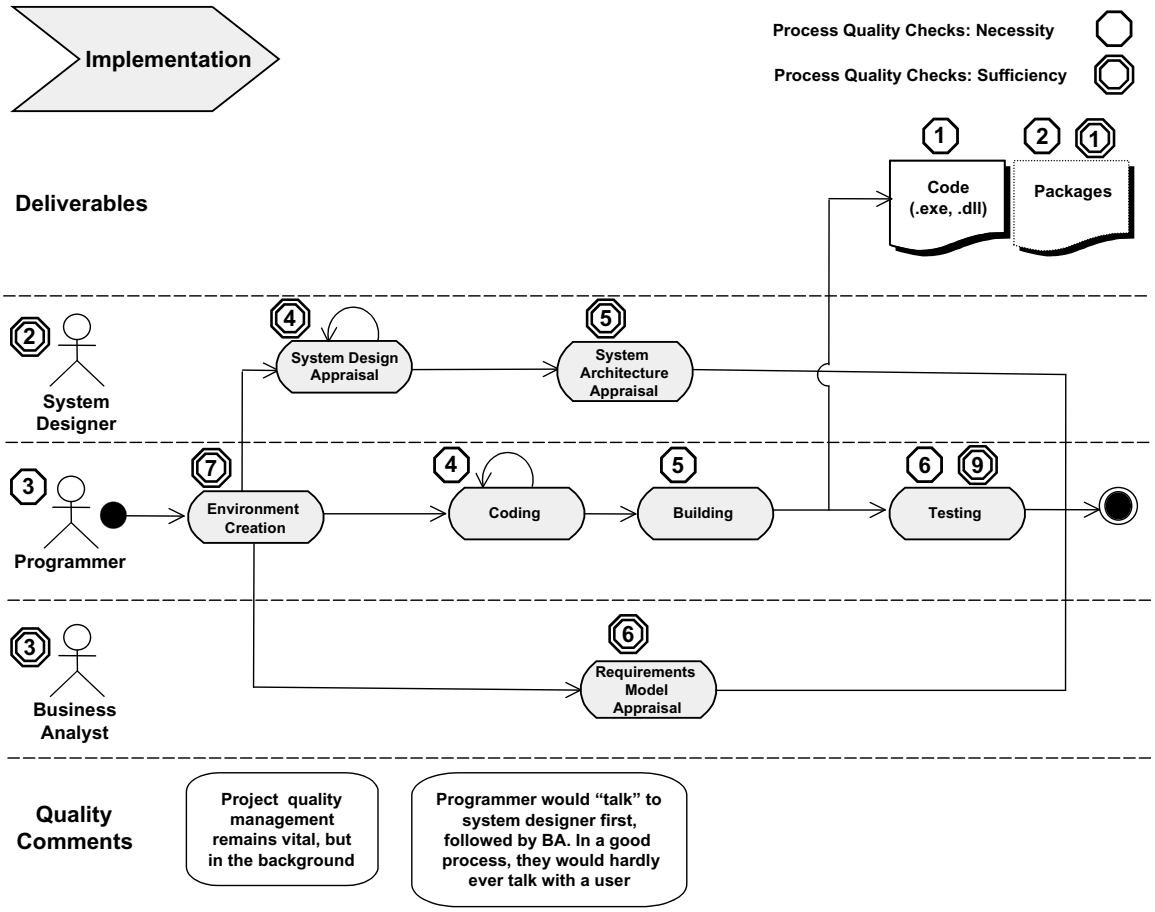


Figure 3.18 *Process-component for implementation*

- The business analyst supports the programmer by ensuring that the requirements are properly understood and met.

3.4.38 Activities and Tasks in Implementation

Figure 3.19 describes the activities and tasks of the process-component of implementation. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

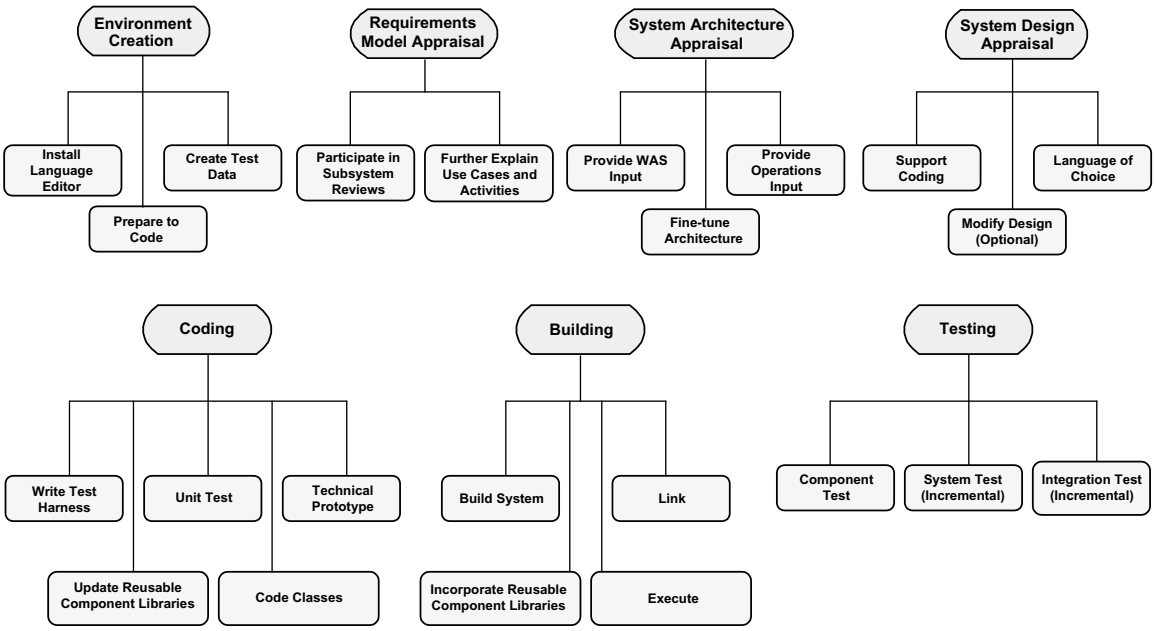


Figure 3.19 *Activities and tasks in implementation*

3.4.39 Deliverables in Implementation

- The code (executable) is the final software deliverable resulting from the modeling effort. This executable may not be a single file; it may be comprised of a number of executables spread over the system architecture.
- Reusable libraries (components and packages). In addition to the final code produced, there should be a provision, in a good process, to produce and store reusable components for future use. This reusable library deliverable enables the creation and storage of reusable code, typically made up of components and packages.

3.4.40 Quality Comments on Implementation

Necessity

1. Executable code, including dynamic libraries, is the primary product of this process-component. While this is produced iteratively, eventually it is the product deployed. In outsourced

projects, this deliverable is produced by an external (outsourcing) organization.

2. The reusable library deliverable is almost an integral part of code production because it can be termed an interim deliverable before the final product is produced. This is because most object-oriented/component-based systems are not built as systems but, rather, as reusable components, which are then assembled to create the product.
3. The programmer is at the core of this process-component. Her profile, job description, skills, and experience should be well coordinated and a match with the project technology. Furthermore, it is vital that the programmer is able to converse with the system designer to determine that the designs are properly understood. In a small project this role may merge with the design role.
4. Coding is the primary activity of this role. It is performed by the programmer in an iterative and incremental manner. This activity requires that the programmer writes classes, tests harnesses, conducts unit tests, and updates the reusable class libraries with the created components.
5. As required by most programming environments, the activity of coding is followed by the detailed activity of building the software. This requires the programmer to integrate the code written with associated libraries in order to make it runnable.
6. Having written the code and built the executable, it is necessary for the resultant module within the product to be tested in detail. This testing ensures that the testing progresses incrementally from component to system to integration tests. Incremental testing implies creating a component and testing it first. This is followed by creating another component (perhaps by another programmer) and testing it. Once the second component is created, it may be necessary to test them together. Once a significant number of components are created, they will have to be integrated with other systems (for example, legacy systems). Thus, the incremental creation and addition of components to the system is what is described in the testing activity.

Sufficiency

1. Reusable class libraries need additional checks in terms of generalization of code. When the classes and components are created, they need to be generalized incrementally. This generalization may happen in the second or third iteration of the current development, or in subsequent projects.
2. The system designer provides the sufficiency of process by supporting the programmer. The necessary aspect of the system designer role is discussed in the system design process-component. Here, he is supporting the programmer, thereby satisfying the sufficiency criteria in the process.
3. The business analyst, similar to the system designer, provides the sufficiency aspect of the process by supporting the programmer in explaining the requirements on a "need to know" basis.
4. The activity of system design appraisal provides information to the programmer on the languages and middleware recommended by the system designer. Optionally, the activity of coding influences the system design as well.
5. Similar to the influence of system design on coding and vice versa, there is influence of the architecture on coding and vice versa. This means the architectural decisions taken in the system architecture process-component provide the boundaries for coding.
6. The requirement model's appraisal may also be necessary, optionally, for the programmer who is trying to understand and keep in mind the use cases at the highest level. Although the activity of coding is driven by class diagrams in the MOSS, the influence of requirements should be considered, wherever relevant.
7. Environment creation, if treated separately, ensures that the development environment is given its due and separate importance. For small projects, however, it may not be necessary to treat this separately, and the activity may be performed by the programmer as a part of coding.

3.4.41 Prototyping Process-Component

The process-component for prototyping combines skills of requirements modeling, system designing, and programming in order to create prototypes; it also benefits the creation of models in the problem, solution, and

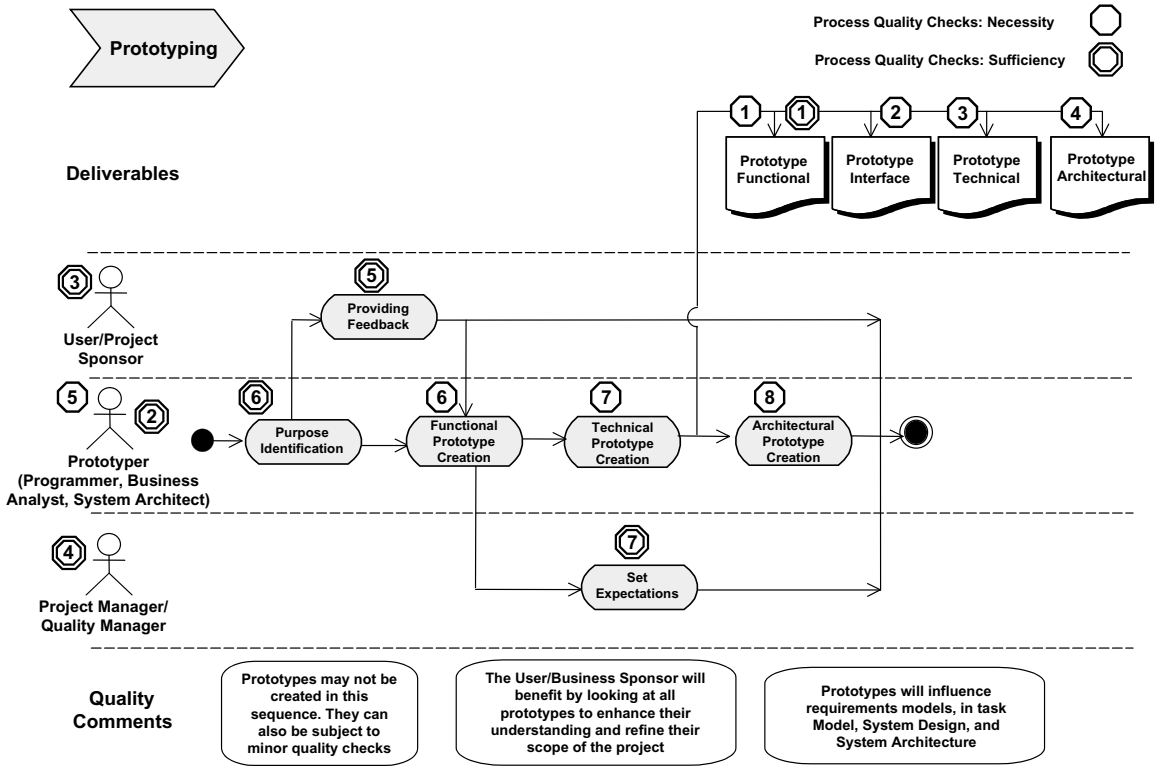


Figure 3.20 *Process-component for prototyping*

background spaces. Since this process-component for prototyping enhances the quality of all other deliverables, prototyping does not stand on its own. I have treated prototyping as a separate process-component because of the need to understand it on its own before using it to improve the quality of deliverables in the other process-components.

While prototypes enhance the overall quality of the system, they themselves should adhere to some quality requirements. For example, prototypes should not give the wrong impression of what can be achieved. This happens most commonly when a highly sophisticated prototype is produced in order to gain project approval; the implementation may not be able to sustain the promises of the prototypes.

In most practical software projects, the process-component for prototyping potentially produces three prototypes:

- The first deals with the functional needs of the users, which includes the needs for the interface, navigation, and overall functionality.
- The second deals with the selected technology and its suitability, such as languages, language compilers, reusable libraries, and databases.
- The third one is the architectural prototype, which deals with issues of security and performance. This architectural prototype also experiments with technologies such as Web application servers, e-services, and mobile services, to consider their appropriateness in the project or the overall organization. Therefore, in addition to the knowledge of the languages and databases, there is also a need to understand the overall environment for implementation.

3.4.42 Roles in Prototyping

- Prototyper (programmer, business analyst, system architect)
- User/project sponsor
- Project manager/quality manager

3.4.43 Activities and Tasks in Prototyping

Figure 3.21 describes the activities and tasks of the process-component of prototyping. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.4.44 Deliverables in Prototyping

- Prototype (functional)
- Prototype (interface)
- Prototype (technical)
- Prototype (architectural)

3.4.45 Quality Comments on Prototyping

Necessity

1. The functional prototype is a necessary aspect of any process because it helps set and manage expectations of users and business sponsors.
2. The interface prototype is usually a GUI prototype that may be produced along with the functional prototypes. It provides the

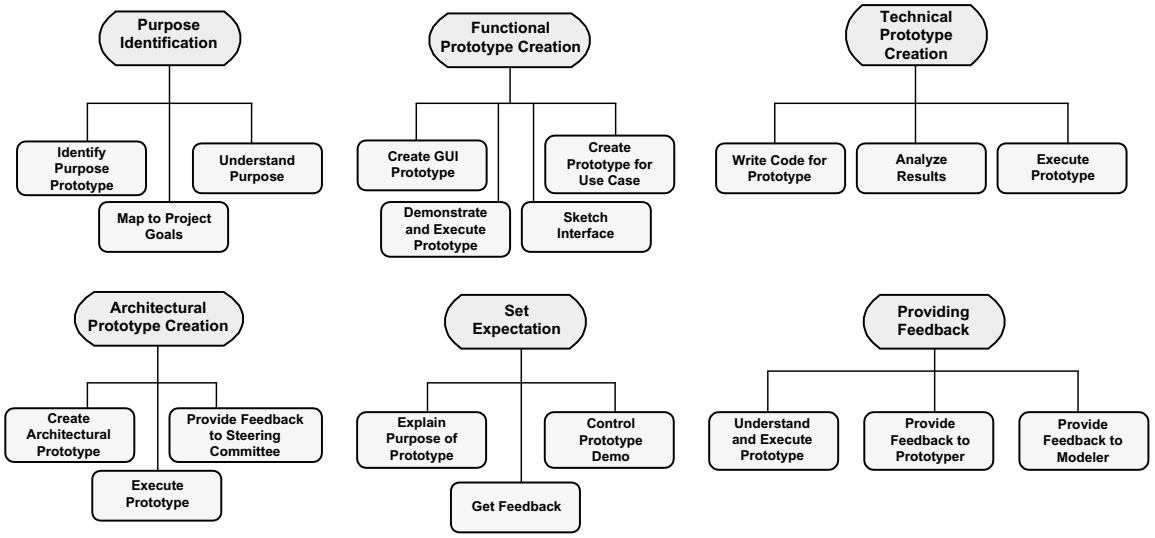


Figure 3.21 *Activities and tasks in prototyping*

"look and feel" of the system and should be attempted after at least some part of the functionality is clear. Otherwise, there is a risk that the functional requirements will be sidestepped for discussions related to the interface.

3. The technical prototype provides information to the system designer and the data modeler on the capabilities of the solution. It is important that this prototype is produced, however briefly, before the actual solution is implemented.
4. The architectural prototype provides information related to the architectural capabilities that already exist in the organization, its limitation, and how the system architecture fits with the overall enterprise architecture. For large projects, information and security architecture have separate influence and, therefore, benefit by creation of a prototype.
5. The role of a prototyper can be played by any of the other roles shown in Figure 3.20, depending on the type of prototype being created.

6. Functional prototype creation provides the necessary input into the requirements modeling exercise. However, this may not be an executable prototype.
7. Technical prototype creation usually has an executable that tests the capability of the technologies in providing the solution; it relates to the operational requirements.
8. Architectural prototype creation also relates to the operational requirements.

Sufficiency

1. The functional prototype needs to be iterated and checked more than once in order to reach a satisfactory level of acceptance. Because of the importance of the functional prototype in the project—especially in reducing misunderstandings between users and developers—this prototype should be carefully created and agreed upon for sufficient quality.
2. Prototyper is not just one person playing one role but, perhaps, more than one person playing multiple roles. While a prototyper is necessary in this process-component, it is the variation to this role that satisfies the sufficiency criteria.
3. The user or project sponsor provides sufficient depth to the prototyping exercise by providing detailed feedback to the prototypers on their requirements, as well as getting a good understanding of their own expectations.
4. The project manager or quality manager also provides the additional support in understanding the user expectations correctly.
5. Providing feedback is an activity that enables the prototypers to understand their own prototypes and iteratively improve them to enable the users to express their needs correctly.
6. Purpose identification helps to focus on the purpose of the prototype. It is only when the question "Why are we creating this prototype?" is answered that the prototyper can be comfortable with his work.
7. While a prototype is created for numerous purposes, setting the expectations of the users is one of the most important aims of prototype creation. This check provides the sufficiency criteria of rightly setting the expectations of the users.

3.4.46 Change Management Process-Component

The process-component for change management supports all the changes that occur in the project. In addition to the sociological aspect of change, this process-component also deals with the critical job of providing support for the configuration and version management needs of the project. Versioning and version release is important in an IIP process. This is where the configuration management part of change management helps to put together the product releases. Therefore, this process-component is closely associated with that of process configuration and

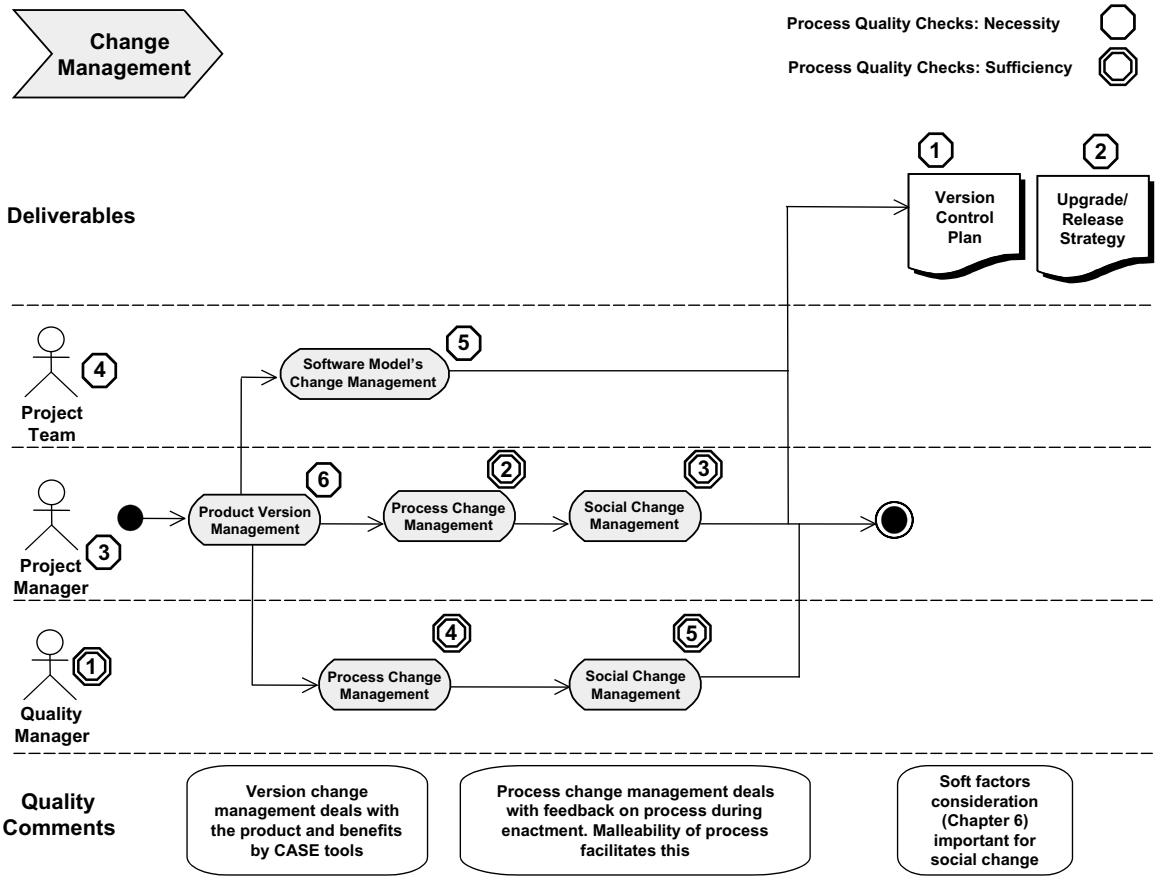


Figure 3.22 *Process-component for change management*

project management. Once an item such as a UML model or a class has reached a stable situation it should be placed under the change management process-component.

3.4.47 Roles in Change Management

- Project team
- Project manager
- Quality manager

3.4.48 Activities and Tasks in Change Management

Figure 3.23 describes the activities and tasks of the process-component of change management. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

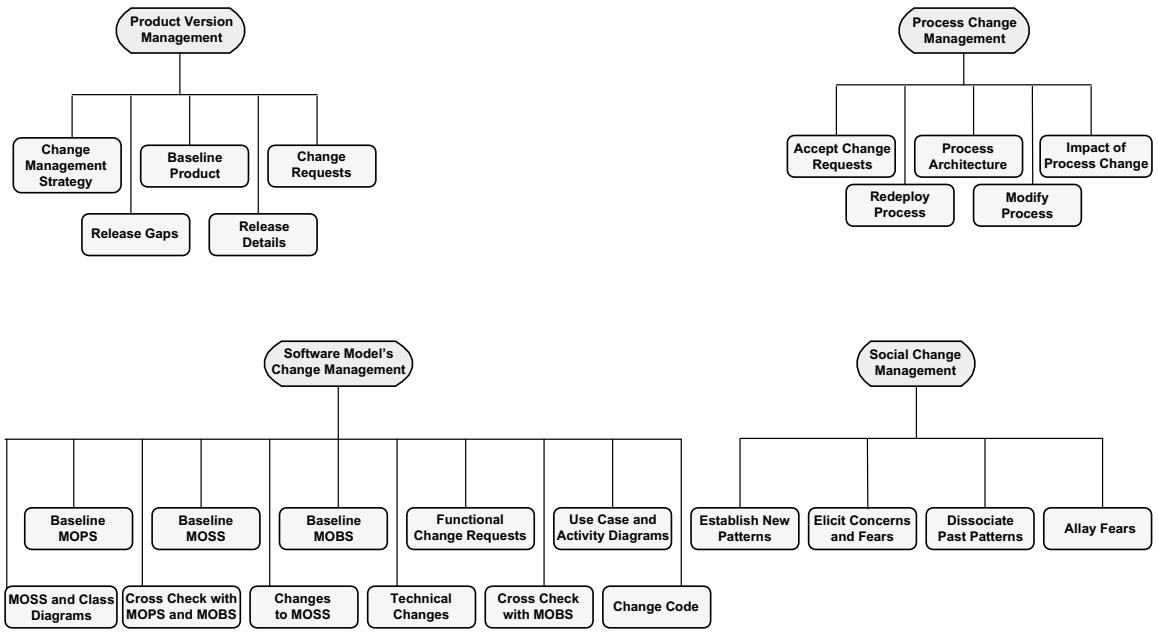


Figure 3.23 *Activities and tasks in configuration management*

3.4.49 Deliverables in Change Management

- Version control plan deals particularly with the software product version control. This is either a separate deliverable or part of the change management plan.
- Upgrade/release strategy is the end result of the effort made in change management. It deals in general with any upgrade to the software, environment, or teams. In particular, though, it deals with upgrades to the software models and products.

3.4.50 Quality Comments on Change Management

Necessity

1. The version control plan deals with the versioning of software releases. This is a necessary part of a good process because this plan decides on the version numbering and deployment of the software product.
2. Upgrade/release strategy is the result of the activities in change management. While mostly it affects a software model or product, this upgrade/release strategy can also be a change in team structure, organizational structure, and so on.
3. The project manager effectuates the change.
4. The project team undergoes the change in most cases.
5. The software model's change management will likely be the most important aspect of change management.
6. Version management ensures that the changes and upgrades brought about in the software are appropriately released in the user community.

Sufficiency

1. The quality manager provides additional support to the project manager in bringing about the changes. In some cases, though, the quality manager herself may bring about the change.
2. Process change management provides sufficiency in terms of process steps when change is brought about.
3. Social change management is usually associated with product change management, although the way in which it is dealt with is different from the product change management approach.

4. Process change management is supported by the quality manager.
5. Social change management is supported by the quality manager.

3.4.51 Enterprise Architecture Process-Component

The enterprise architecture (EA) process-component deals with the overall enterprise modeling and ensures that the system produced as a result of the project under consideration is able to operate with the existing systems of the enterprise. At a project level the activity of creating the enterprise architecture is limited, but ensuring that the system architecture conforms to the EA (resulting potentially in an EA Integration) is high.

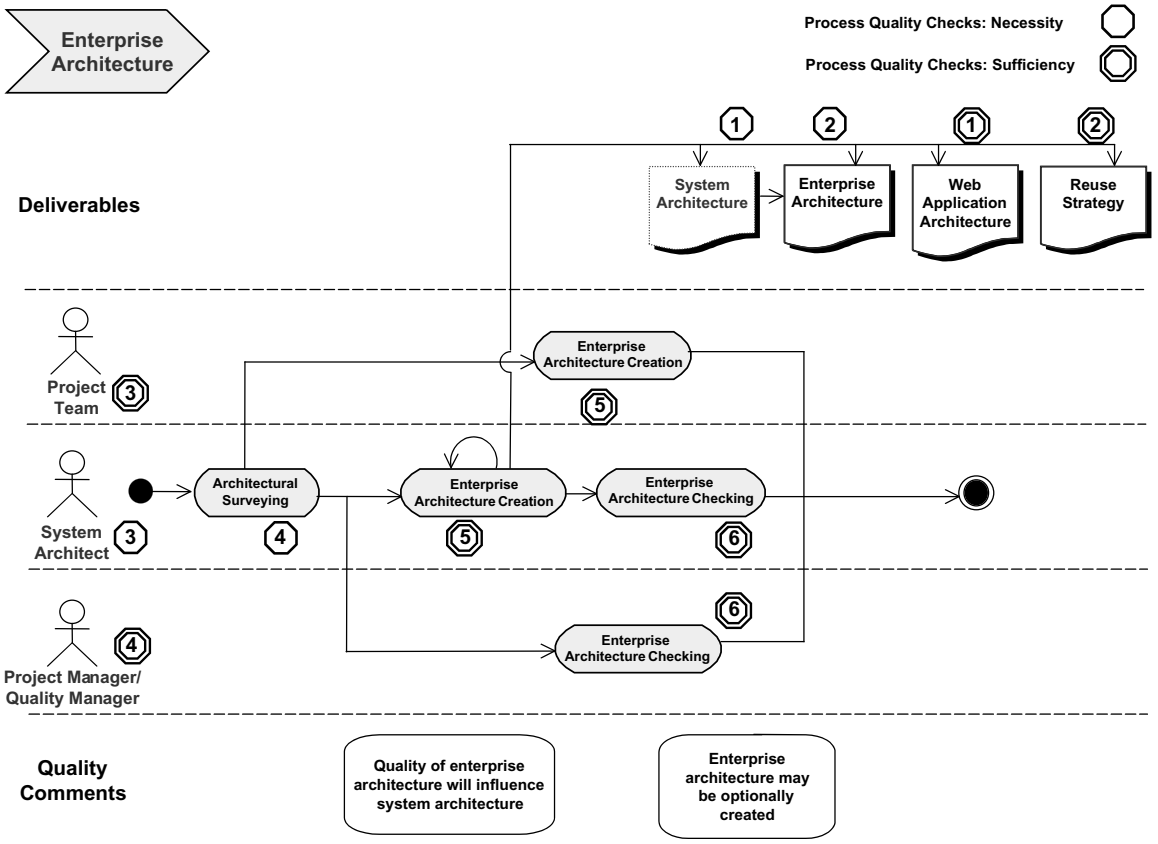


Figure 3.24 *Process-component for enterprise architecture*

3.4.52 Roles in Enterprise Architecture

- System architect provides expertise and experience in producing a good, robust enterprise architecture.
- Project team interacts with the architect to ascertain the mechanism to implement the architecture and highlights the possible limitations of the existing technical environment.
- Project manager/quality manager organizes and manages the creation of the enterprise architecture.

3.4.53 Activities and Tasks in Enterprise Architecture

Figure 3.25 describes the activities and tasks of the process-component of enterprise architecture. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

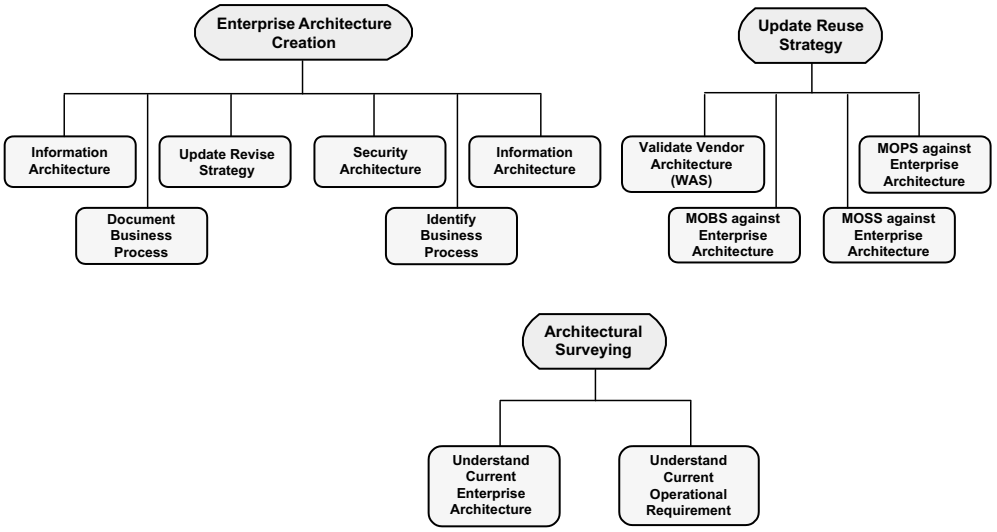


Figure 3.25 *Activities and tasks in enterprise architecture*

3.4.54 Deliverables in Enterprise Architecture

- Enterprise architecture is made up of the actual enterprisewide architecture and the document that outlines that architecture.
- System architecture is produced during the system architecture process-component and embellished here. Alternatively, creation of the initial system architecture may start here, followed by its rigorous upgrade in the system architecture process-component.

3.4.55 Quality Comments on Enterprise Architecture

Necessity

1. A primary necessity check ensures that the system architecture is iteratively cross checked against the enterprise architecture. This results in compliance of the system architecture with the enterprise architecture.
2. In most projects, the opportunity to create enterprise architecture is limited. However, each project influences the overall architecture of the enterprise, and may incite rethinking on the part of the architects in terms of, say, bandwidth requirements or database capacity.
3. The system architect is the primary role in this process-component, in terms of checking the influence of enterprise architecture on the system architecture. When decisions related to the enterprise and all its related systems and architecture are to be made, the role of system architect may be played by a senior technical person well versed in the enterprise architecture. The system architect ensures that the architecture of the system conforms to the enterprise architecture.
4. Architectural surveying deals with understanding the existing enterprise architecture to ensure that the system architecture is created within the bounds of the enterprise architecture. Operational requirements are also considered in surveying the overall architectural needs.
5. It is important to perform the activity of enterprise architecture checking while keeping in mind its potential effect on the project and the quality management aspect of the project.

Sufficiency

1. With the advent of Web-based solutions in almost all modern-day projects, it is important to consider the Web application architecture in overall enterprise architecture. Web architectures have the ability to influence, and many times change, the manner in which solutions are provided.
2. Reuse strategy is important at the enterprise level in terms of its influence on creating architecture versus buying middleware and Web architectures off the shelf. The reuse strategy influences the system architecture as well.
3. Some senior project team members will be involved in the creation of part of the enterprise architecture. Other team members should be aware of the enterprise architecture.
4. The project manager and the quality manager play supporting roles in ensuring that the enterprise architecture is cross checked against the architecture of the system. The project manager is involved, in particular, when there is a conflict between the system, enterprise architecture, and the ramification of this conflict on project cost and time estimates.
5. Creation of enterprise architecture is not a singular activity with a set completion time. Instead, enterprise architecture is created based on a number of projects, with each project improving and adding to the existing architecture. Sometimes, though, when not only the software system is newly built, but also the organization itself is new, there will be an opportunity to create completely new enterprise architecture. While the system architect is primarily responsible for this enterprise creation, the project team also joins in discussing and understanding the architecture.
6. Enterprise architecture is checked for its completeness and consistency by the system architect. The project manager and the quality manager must facilitate this checking and will have to be fully aware of it as the project progresses.

3.4.56 System Architecture Process-Component

System architecture deals with the architectural work in the background that ensures that the requirements and the designs are in accordance with the overall needs and availabilities of the project. The activities and tasks

in the system architecture process-component take a "bird's-eye" view of the requirements and design, ensuring the consistency and completeness of the MOBS.

3.4.57 Roles in System Architecture

- System architect
- System designer
- Project manager/quality manager

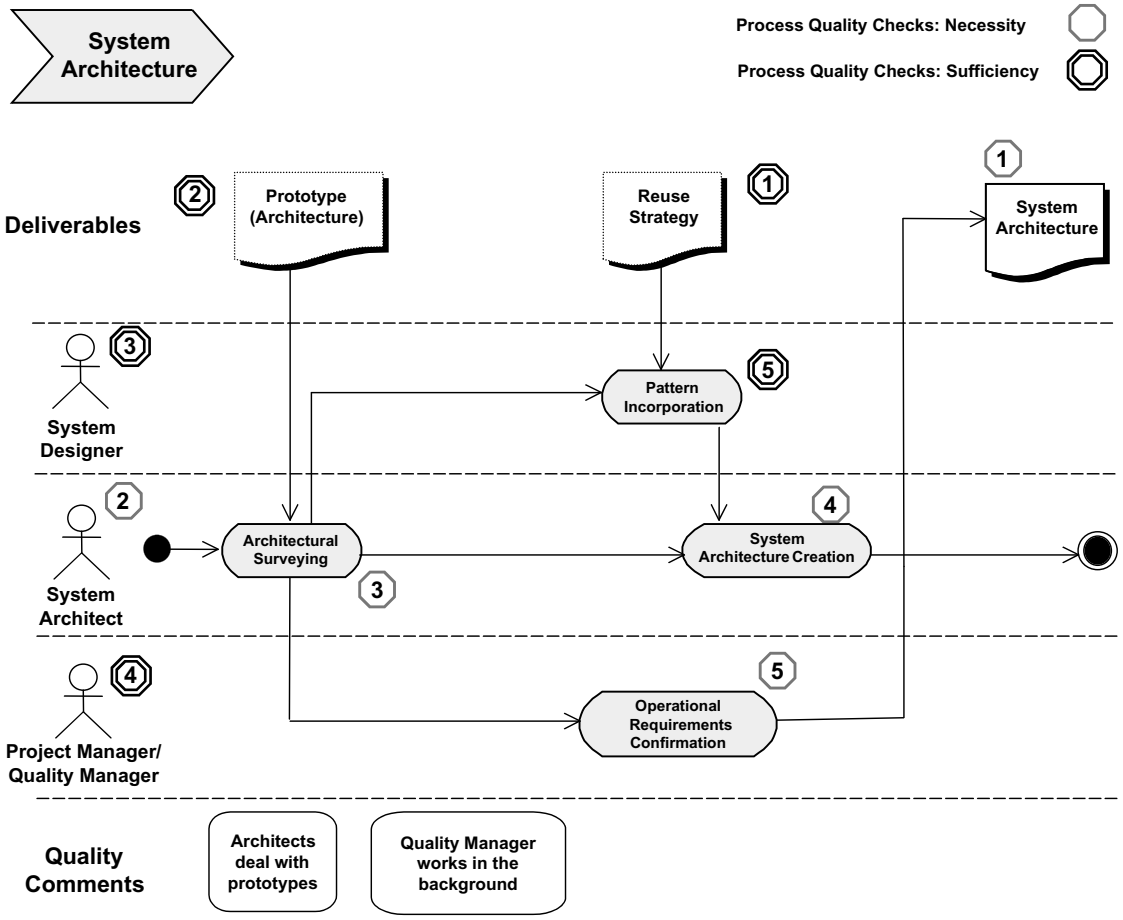


Figure 3.26 *Process-component for system architecture*

3.4.58 Activities and Tasks in System Architecture

Figure 3.27 describes the activities and tasks of the process-component of system architecture. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.4.59 Deliverables in System Architecture

- System architecture (solution architecture).
- Reuse strategy. As seen in the system architecture process-component diagram, the reuse strategy facilitates the incorporation of reusable architectures and designs in the architecture of the current system.
- Prototype architecture also provides valuable input into the system architecture.

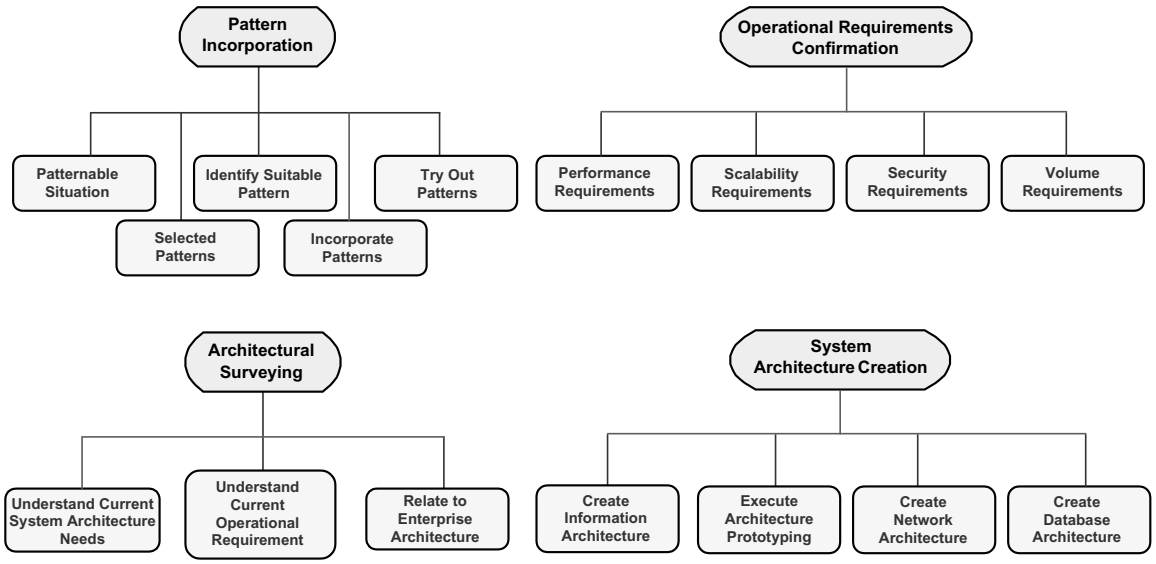


Figure 3.27 *Activities and tasks in system architecture*

3.4.60 Quality Comments on System Architecture

Necessity

1. System architecture is the main deliverable of this process-component. It is either a document outlining the architecture of the system or, especially in large projects, a suite of libraries and patterns on which the actual system is built.
2. The system architect plays the primary role of creating the system architecture.
3. Architectural surveying is an activity that takes stock of the existing enterprise architecture before relating it to the system architecture.
4. The creation of a system architecture deals with information, network, and database architectures, to name but a few. This activity results in the system architecture mentioned in step 1 above.
5. Operational requirements confirmation ensures that all operational requirements are formally incorporated in the system architecture. While other activities in this process-component continue to take input from the operational requirements, this specific activity is intensely focused on ensuring that the system architecture can handle the operational requirements.

Sufficiency

1. Reuse strategy is an iteratively produced deliverable that is updated even during the enterprise architecture process-component. Here, in system architecture, the reuse strategy provides valuable and sufficient input to enable the reuse of patterns and designs.
2. Architectural prototype, created in the prototyping process-component, is used here to help create and verify the system architecture.
3. The system designer plays the supporting role to the system architect in verifying the implementability of the system architecture.
4. The project manager/quality manager also plays the supporting role in facilitating the creation and verification of the system architecture.
5. The pattern incorporation activity provides sufficient depth to the system architectural work by enabling identification, experimentation, and adoption of known patterns. These known patterns are

not restricted to published patterns. They can also include patterns that were discovered and documented in previous projects within the organization.

3.4.61 Deployment Process-Component

See figure 3.28.

3.4.62 Roles in Deployment

- The project manager organizes the deployment and release of the product after taking input from the change management process-component.

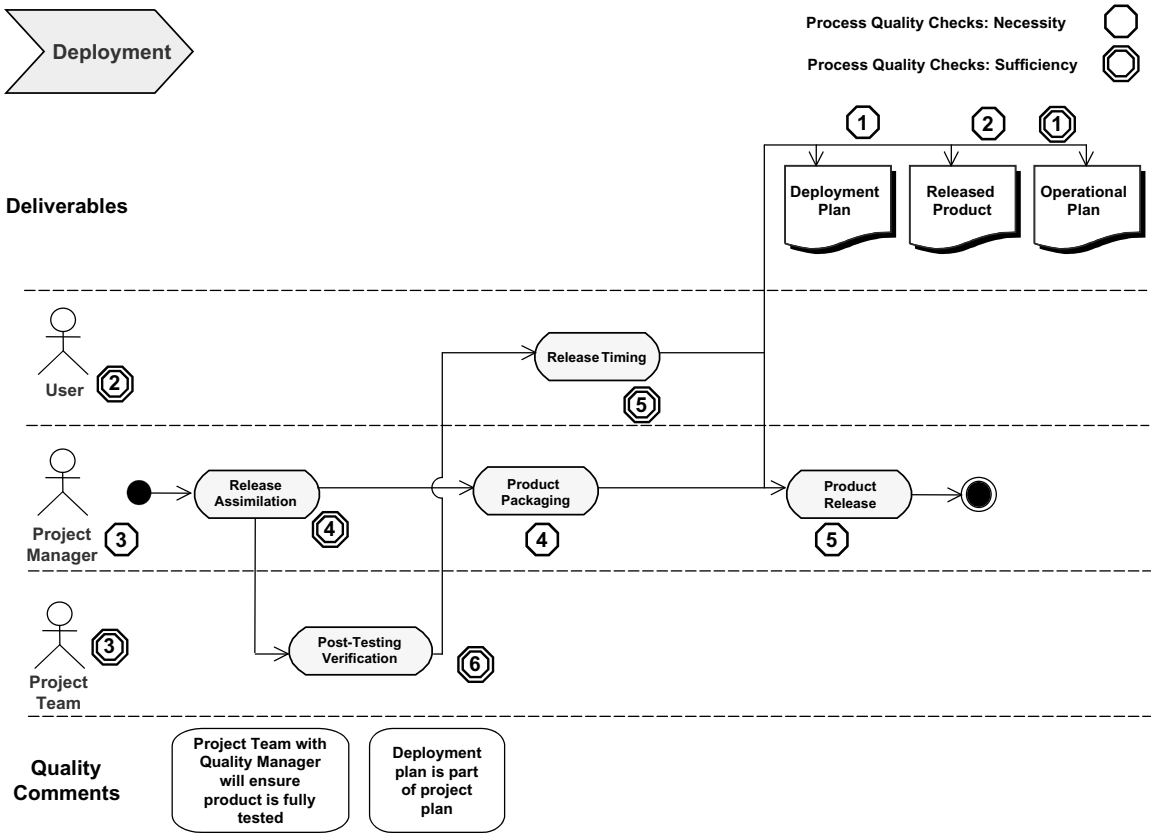


Figure 3.28 *Process-component for deployment*

- The project team participates in deployment and post-deployment reviews.
- The user is involved in ensuring that the deployment of the system is smooth, especially from the sociological angle.

3.4.63 Activities and Tasks in Deployment

Figure 3.29 describes the activities and tasks of the process-component of deployment. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.4.64 Deliverables in Deployment

- The deployment plan deals with the strategies of sending the product out in the real world. This deployment plan also deals with issues related to switching the business over from one product to another. It may be a part of the overall project plan.
- The released product is the final software product that is released to the users.
- The operational plan provides important information in deployment, as it contains details of the system's requirements when it goes out in operation.

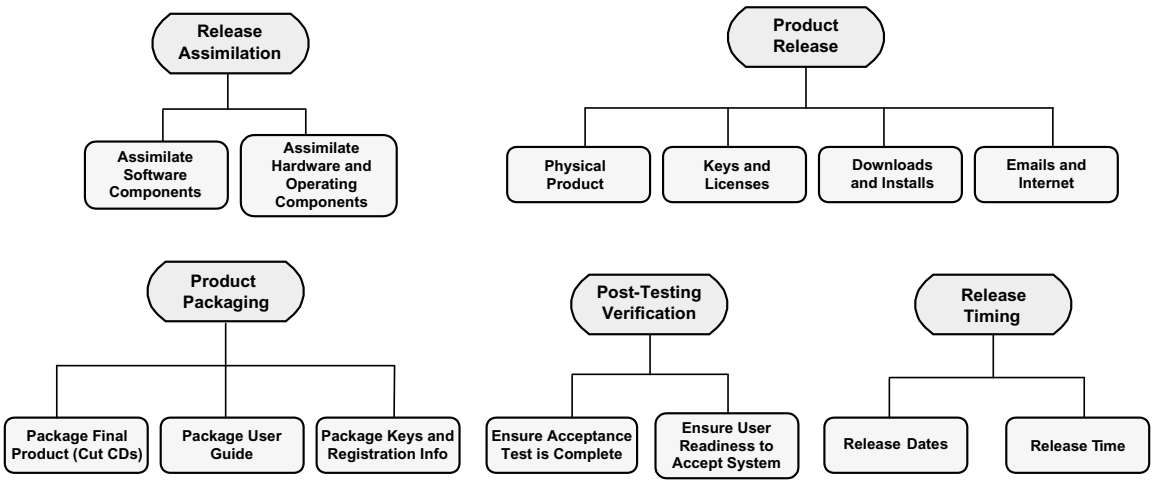


Figure 3.29 *Activities and tasks in deployment*

3.4.65 Quality Comments on Deployment

Necessity

1. It is necessary to ensure that the deployment plan is in accordance with the expectations of the users. This plan must consider the issues of new deployment as well as of switching over, when a replacement product is introduced. It is also necessary to consider supporting materials such as help and user manuals and initial training to support the product being deployed.
2. Check that the released product is in a fully deployable format. This means that not only is the product fully tested in the development environment, but it is also litmus-tested in the production environment before being deployed.
3. The project manager is in a continuous coordination role in this process-component.
4. Product packaging is important in cases where the product is released in physical forms such as CDs and Zip disks. In these cases, packaging of the product, its associated licensing, and so on, is crucial.
5. This is the activity of actually releasing the product to the users.

Sufficiency

1. Check the operational plan for additional information on the system in terms of its operations. For example, backups and mirroring are common requirements of a system in operation and should be specified and adhered to for the system to be of good quality in operation.
2. The user representative facilitates deployment of the system, especially socially, wherein it is made acceptable and promoted within the larger user community.
3. The project team is involved in getting the product ready for release. While there will be very little programming or design-related activities here, the technical aspect of the deployment is still critical and is handled by the project team.
4. Release assimilation becomes important in cases where the product has to be physically sent to the users or when the product is packaged and put on shelves for sale (for example, shrink-wrapped software).

5. Release timing provides for the sufficiency of release by ensuring that the release is properly coordinated. This becomes important when a new module or system is released to replace an existing system that is working on a 24 × 7 basis (online and available all the time).
6. Post-testing verification ensures that the integration and acceptance testing is complete and that the users are ready to accept the system.

3.4.66 Training Process-Component

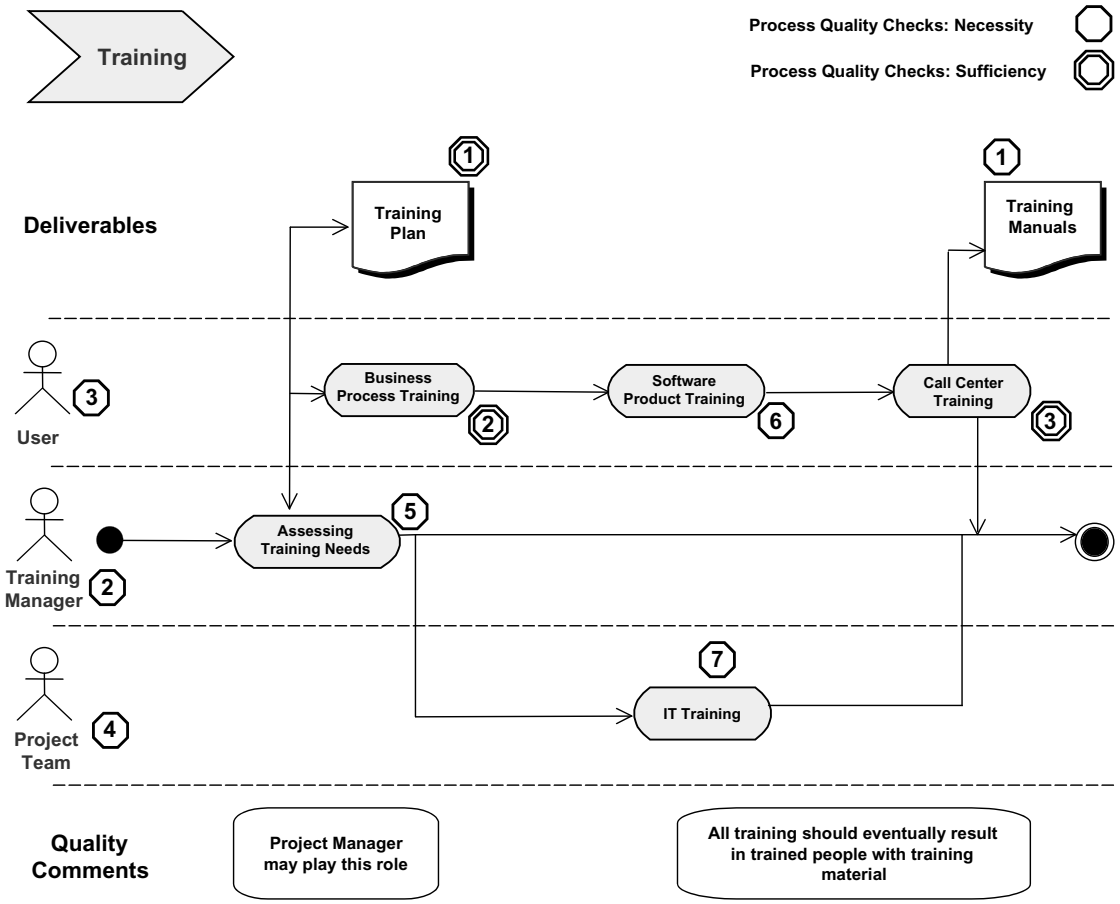


Figure 3.30 *Process-component for training*

3.4.67 Roles in Training

- Training manager, who organizes the training. This role may be played by the project manager or a member of the project team.
- Project team plays a dual role here. Initially, the project team needs technical and other related training. Later, during the deployment of the product, this team is also responsible for creation of a training mechanism for the user. Selected members from the project team can be made responsible for creation of appropriate training packages. User representatives on the project can provide invaluable assistance in creating such training packages.
- Users, who will undergo training and learn to use the system effectively.

3.4.68 Activities and Tasks in Training

Figure 3.31 describes the activities and tasks of the process-component in training. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

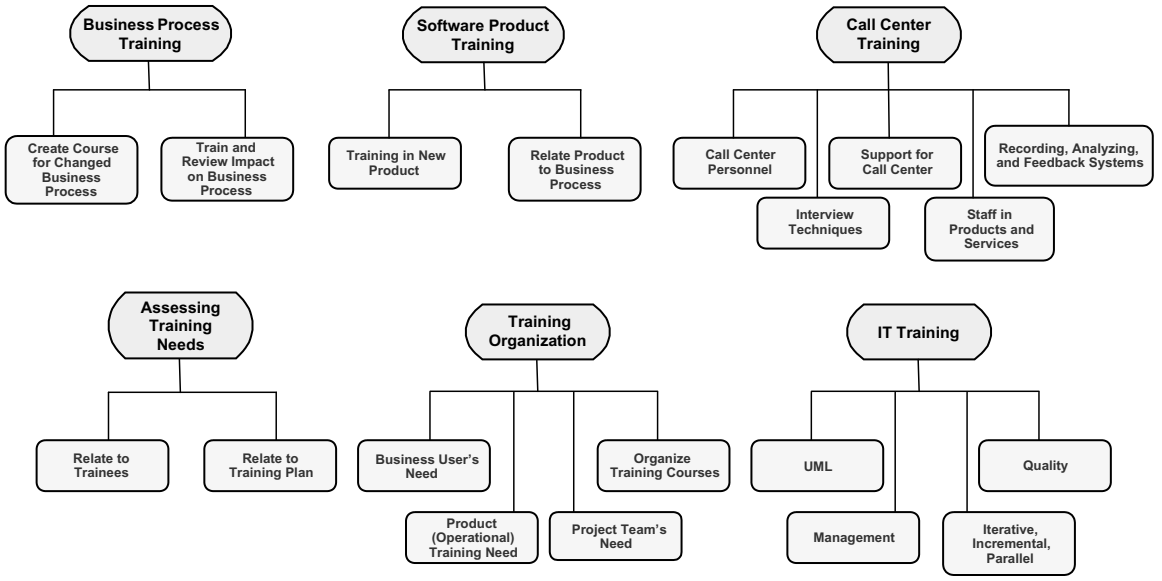


Figure 3.31 *Activities and tasks in training*

3.4.69 Deliverables in Training

- The training plan contains descriptions of what type of training is required, who are the target participants of the training, and most importantly, when the training is conducted.
- The training materials are the handouts or other accompanying materials provided to the participants of a training course. This training material need not be printed paper only, as it is common to have CDs and videos accompanying training courses. These additional tools containing training materials are especially helpful in conducting user training in the software product delivered.

3.4.70 Quality Comments on Training

Necessity

1. The training material is the prime material accompanying a training course. This step ensures that such material is readily available and is provided in a format acceptable to the users.
2. The training manager should be fully aware of the training needs of the organization, the suppliers of the training needs, and the timing of the training. In small projects the project manager may also play this role; large projects will greatly benefit if this responsibility is assigned to a separate individual.
3. Users will have to be trained not only in the use of the new system but also in the altered business procedures as a result of the new system.
4. The project team undergoes training depending on its skill levels and needs. In a large project, it is important to train the team in relevant modeling (UML) and processes early. The project team will also deliver relevant training on their product to the user.
5. Assessing the training needs of all players in the project should be done carefully to ensure that the training relates to the needs of the trainees. It is also important to ensure the support of the management by referring to the training plan.
6. Software product training is undertaken to enable the user to start using the system.
7. IT training is aimed at the project team to equip them with their needs for modeling, process, and technical skills.

Sufficiency

1. The training plan is part of the overall project plan, and describes the training needs of the project. This includes a range of training needs from technical training to be provided to the technical team through end user training on the software product.
2. Business process training ensures that the users are able to carry out the changes in conducting business.
3. Call center training is required if the new system requires the organization to provide that support.

3.4.71 Reuse Process-Component

The process-component for reuse provides the crucial background set of activities and tasks that actively encourage reuse at all levels within the project. This includes not only the well-known code level reuse (through

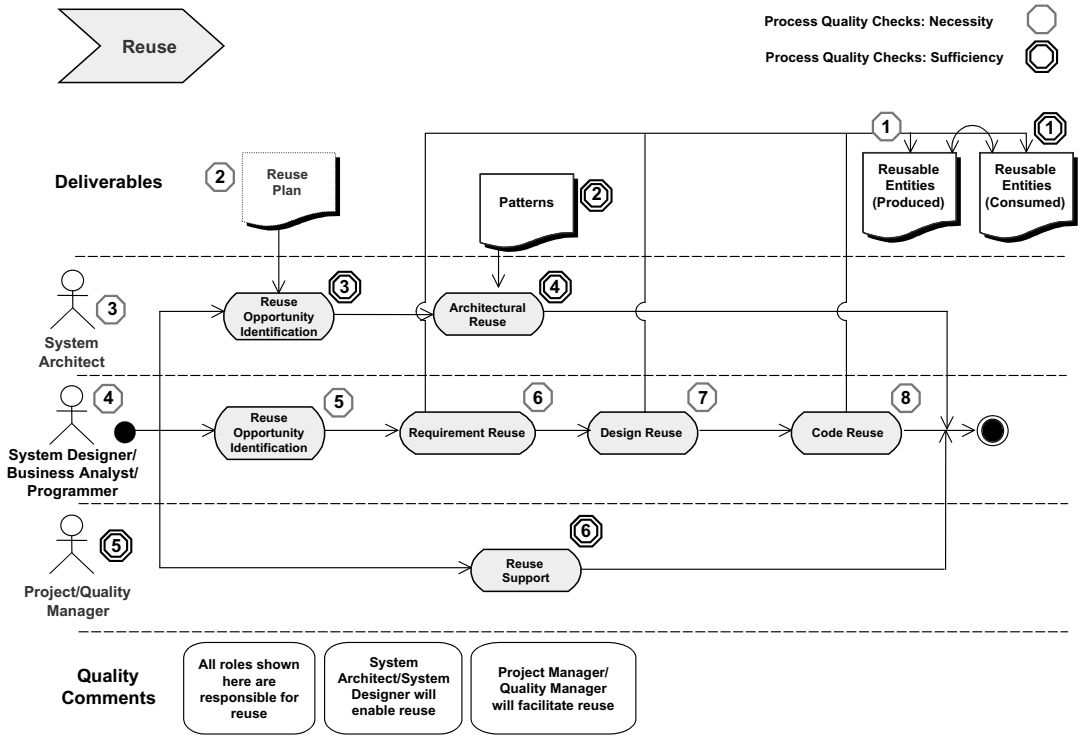


Figure 3.32 *Process-component for reuse*

classes and components in the domain of object technology) but also includes the reuse of requirements, architecture, design, and testing.

3.4.72 Roles in Reuse

- System architect, who knows enough about the overall technical environment to facilitate both creation and consumption of reusable components.
- System designer/business analyst/programmer
- Project manager/quality manager

3.4.73 Activities and Tasks in Reuse

Figure 3.33 describes the activities and tasks of the process-component in reuse. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

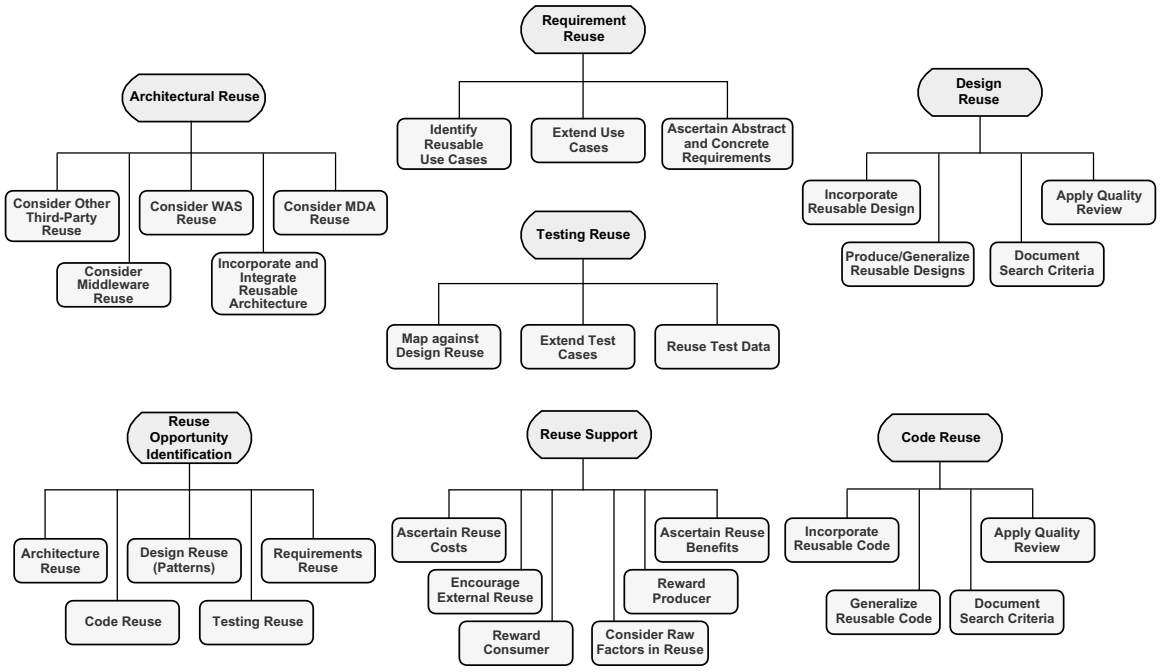


Figure 3.33 *Activities and tasks in reuse*

3.4.74 Deliverables in Reuse

- The reuse plan helps to organize the process-component of reuse within the project.
- Reusable entities (consumed) are models and documentation of reusable components that have been inserted in the architecture and the software design within the current project.
- Reusable entities (produced) are models and documentation of potential reuse components produced by this project for future use.

3.4.75 Quality Comments on Reuse

Necessity

1. Reusable entities are produced by the project.
2. The reuse plan iteratively provides input into enabling reuse opportunity identification.
3. The system architect provides necessary support in terms of enabling reuse strategies by identifying opportunities for reuse and actually reusing the architectural aspects of the system.
4. The system designer, business analyst, and programmer are involved in their respective aspects of reuse.
5. Reuse opportunity identification is an activity carried out by people of relevant roles.
6. Requirement reuse is important and is easily facilitated by the UML's use case-based approach.
7. Design reuse is also facilitated by the object-oriented aspect of the UML and, in particular, design patterns and class diagrams. It is important to note that both production and consumption of reuse is encouraged in design reuse.
8. Code reuse is well known and the first attempt at reuse by the software community.

Sufficiency

1. Reusable entities consumed by the new projects provide for the sufficiency aspect of reuse.
2. Patterns—both published and internally created—are crucial for analysis- and design-level reuse in any project.

3. Reuse opportunity identification is additionally carried out by the system architect.
4. Architectural reuse enables reuse of Web application servers and other middleware-type architectures.
5. The project manager and quality manager facilitate reuse and work on the reward structures supporting reuse.
6. Reuse support deals with the managerial activities of rewarding reuse and also promoting the policy of "reuse rather than build."

3.5 The Quality Process

3.5.1 Quality Management Process-Component

The quality management process-component strives to bring together the social and methodological aspects of the project with a focus on quality. Some of the responsibilities of this process-component are planning for the project while keeping quality in mind, identifying the standards (including the UML standards for modeling), setting the expectations of the users, and, most importantly, getting the right people together for the quality work.

3.5.2 Roles in Quality Management

- Quality manager organizes and manages the quality function including creation of the quality environment.
- User highlights the quality expectations and provides feedback on the product/team's ability to meet those expectations.
- Quality team/quality analysts carry out the quality functions.

3.5.3 Activities and Tasks in Quality Management

Figure 3.35 describes the activities and tasks of the process-component in quality management. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.5.4 Deliverables in Quality Management

- The quality plan handles the overall approach to quality within the project.

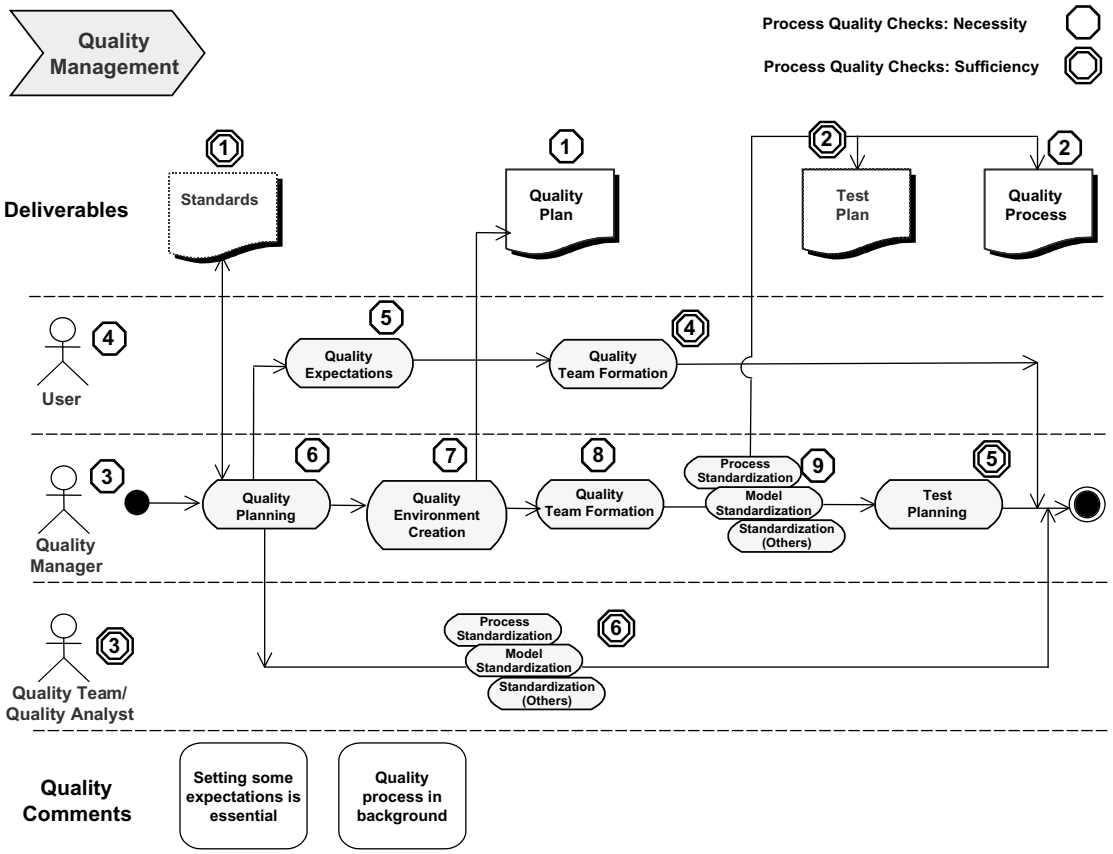


Figure 3.34 *Process-component for quality management*

- The test plan handles the management aspect of testing within the project.
- The Quality Software Process describes the software process that can be customized and followed by the project members.
- The standards (interim/input) exist at all levels within the project, including design, documentation, coding, and testing standards, to name but a few.

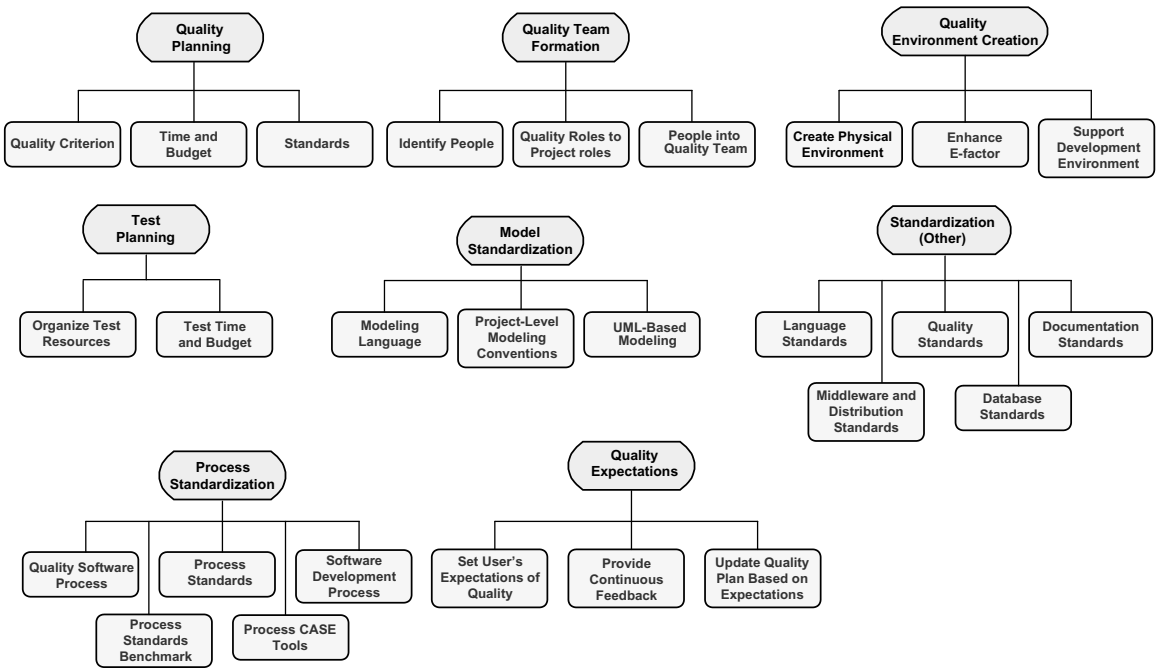


Figure 3.35 *Activities and tasks in quality management*

3.5.5 Quality Comments on Quality Management

Necessity

1. The quality plan includes all documentation related to organizing the quality function. This includes all resources and schedules related to quality management, as well as to the project itself. This can be considered the main controlling deliverable for all quality-related activities in the project.
2. The quality process is part of the overall QSP, but specifically deals with the process-components related to quality.
3. The quality manager is responsible for overall planning, execution, and tracking of the quality functions—supported by the project manager.
4. The user participates in explaining her quality needs and expectations, as well as providing input into quality activities.

5. The quality expectations are created and continuously updated by the user with the quality manager.
6. The quality planning activity undertakes planning and documentation of all quality functions.
7. The quality environment creation deals with the creation of the physical and technical environment for quality activities.
8. The quality team formation includes staffing, organizing, and motivating the people who perform quality roles in the projects.
9. The process/model/other standardizations deal with creation and implementation of all relevant standards within the project.

Sufficiency

1. The standards document provides a reference point for all standards within the project. In addition to the process and model standards, there are standards related to languages and databases, which are referred to in this iteratively produced standards document.
2. The test plan deals with the quality control aspect of testing. It is created here especially from the resourcing point of view.
3. The quality team and quality analyst are responsible for following the standards. At this stage, though, they provide input into the project and the organizational-level standards.
4. The quality team formation is achieved by users who may decide to become part of the quality team, facilitating direct and continuous input in terms of quality expectations.
5. The test planning activity creates and updates the test plan.
6. The process/model/other standardizations are updated and followed by the quality team.

3.5.6 Quality Assurance Process-Component

Quality assurance follows the process-component of quality management and it undertakes the actual effort of assuring the quality of the models and processes in the project. As with most other process-components, the quality assurance process-component is not an independent process-component, but rather is intertwined with the process-components that

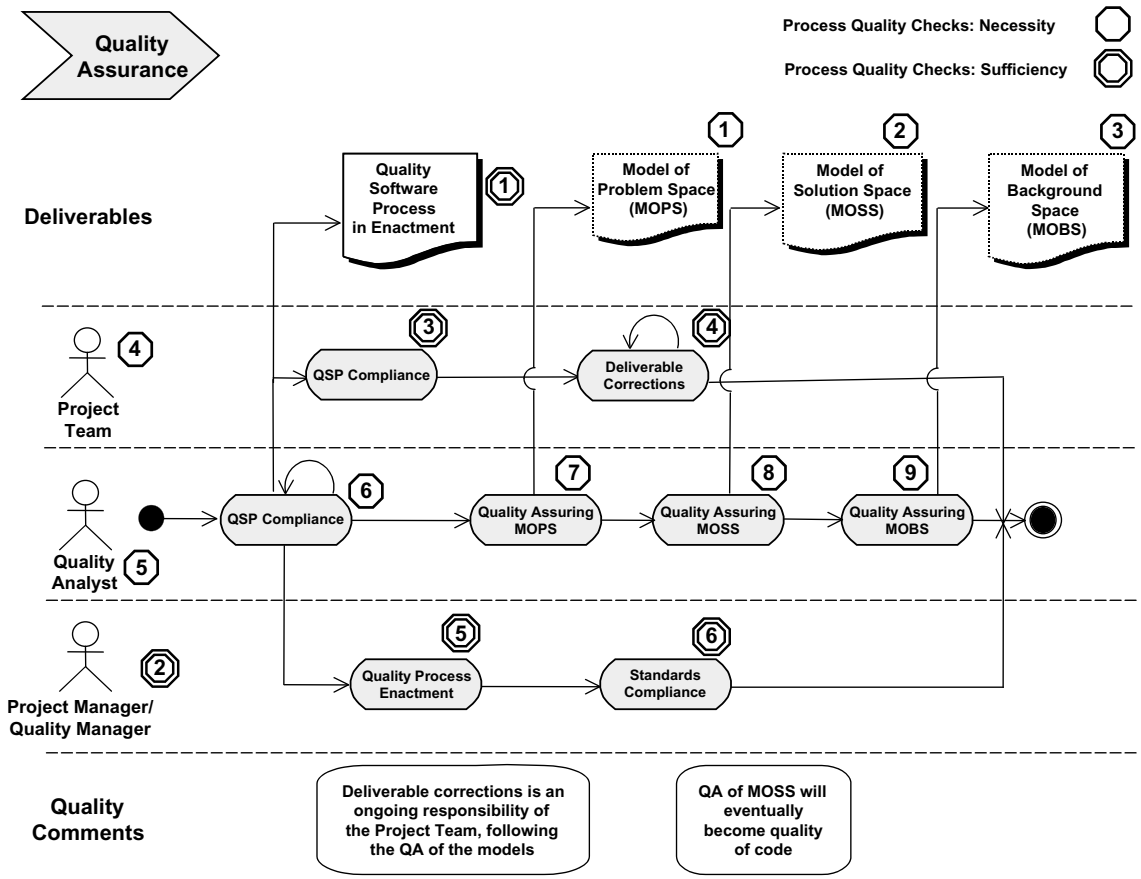


Figure 3.36 *Process-component for quality assurance*

produce the MOPS, MOSS, and MOBS. The project manager and the quality manager play key supporting roles in the execution of this process-component.

3.5.7 Roles in Quality Assurance

- Quality analyst
- Project team
- Project manager/quality manager

3.5.8 Activities and Tasks in Quality Assurance

Figure 3.37 describes the activities and tasks of the process-component in quality assurance. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.5.9 Deliverables in Quality Assurance

- Quality Software Process in enactment. The QSP provides the basis for enacting the quality aspect of the project. This is made up of the templates, deliverables, activities, and tasks (including the ones mentioned here) that focus on quality.
- MOPS. The Model Of Problem Space is subject to the rigors of quality assurance activities and tasks.
- MOSS. The Model Of Solution Space is subject to the rigors of quality assurance activities and tasks.

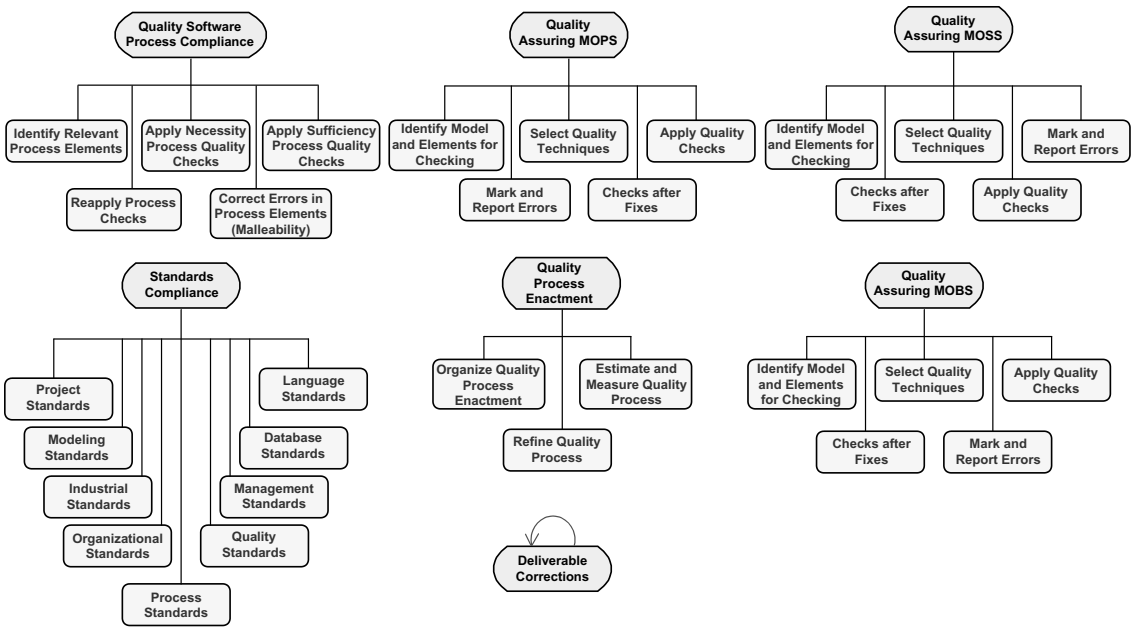


Figure 3.37 *Activities and tasks in quality assurance*

- MOBS. The Model Of Background Space is subject to the rigors of quality assurance activities and tasks.

3.5.10 Quality Comments on Quality Assurance

Necessity

1. The MOPS contains the relevant UML diagrams, descriptions, and specifications that are subjected to the quality checks. It is essential that this model is at a suitable level of completion (ideally at the end of an iteration) before quality checks are applied to it. As a result of the quality activities, the MOPS is updated.
2. The MOSS contains the solution-level UML diagrams. This model is iteratively produced; therefore, its quality checks also iterate with the other two models.
3. The MOBS is also subject to quality checks (a list of suggested quality checks appears in the accompanying CD for all three models including MOBS).
4. The project team plays the supporting role to the quality analyst in his attempts to ensure quality of the models.
5. The quality analyst is the primary role in this process-component. This role ensures that all the quality-related activities and tasks are carried out at the correct time and by the right people. In large practical projects, more than one person plays this role.
6. Quality Software Process compliance ensures the compliance of the process as it is enacted. The necessary, sufficient, and malleable aspects of the process itself are assured here. This activity is carried out by the quality analyst.
7. Quality Assuring MOPS undertakes the quality checks for MOPS as suggested on the accompanying CD.
8. Quality Assuring MOSS undertakes the quality checks for MOSS as suggested on the accompanying CD.
9. Quality Assuring MOBS undertakes the quality checks for MOBS as suggested on the accompanying CD.

Sufficiency

1. The Quality Software Process in enactment is not a document-based deliverable but rather represents the entire QSP in enactment.

2. In addition to the necessary roles played by the quality analyst, project manager, and quality manager, these managerial roles provide the organizational support for the quality effort.
3. Quality Software Process compliance is supported by the project team.
4. Deliverable correction is a reminder activity. It reminds the project team that they will continue to make corrections to the models, executables, and other deliverables in the project.
5. In the quality process enactment, the project manager and the quality manager enact the process configured earlier in the process-configuration process-component.
6. The standards compliance is also organized by the quality manager, but may be enforced by the quality analyst.

3.5.11 Quality Control Process-Component

While the process-component for quality control is mentioned here as a part of the overall quality process, it is discussed in great detail in Chapter 6.

3.5.12 Roles in Quality Control

- Tester
- Modeler/programmer/user
- Quality manager

3.5.13 Activities and Tasks in Quality Control

Figure 3.39 describes the activities and tasks of the process-component in quality control. Refer to the accompanying CD for a tabular form of these activities and tasks to enable you to create your own customized project plan.

3.5.14 Deliverables in Quality Control

- The test plan contains the organizational aspect of testing. This includes details of the people and the schedules of testing, as well as what is expected of the testing process.
- The test design provides a more tactical view of testing. Test designs can be organized around subsystems or packages.

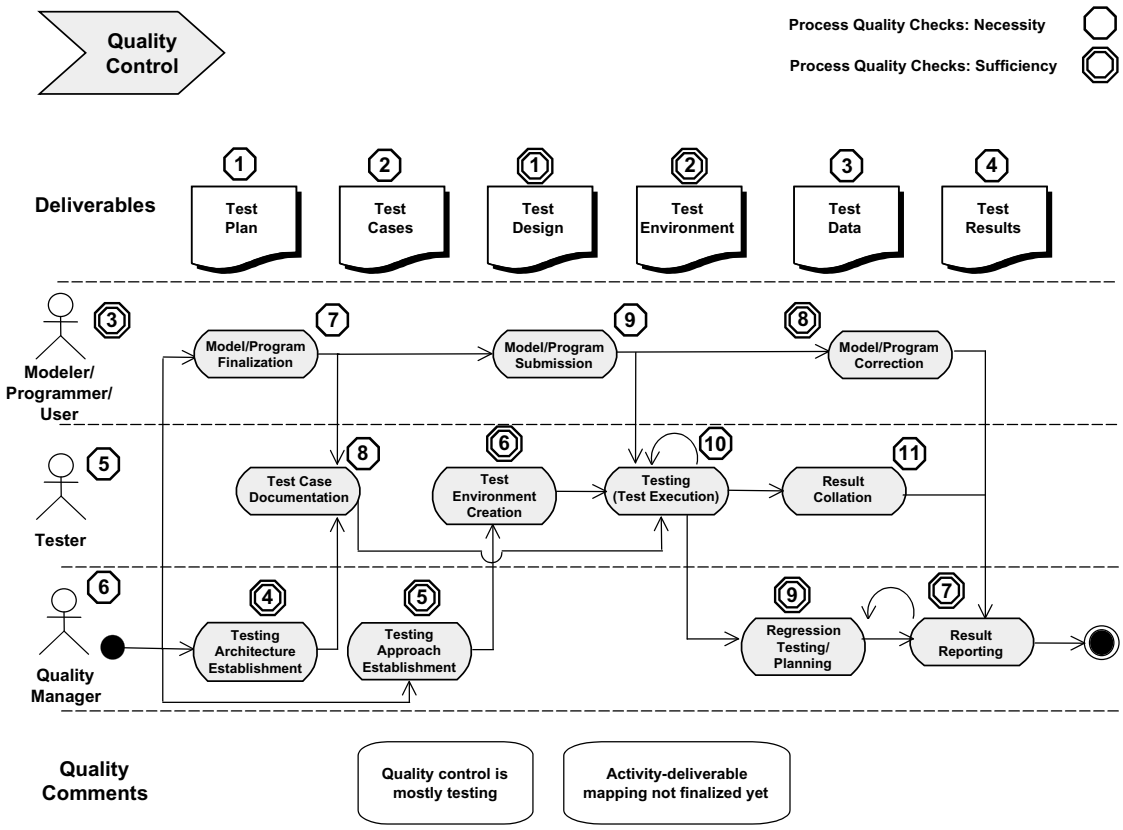


Figure 3.38 *Process-component for quality control*

- The test environment is the physical environment that needs to be created before testing can begin. It is also the software environment (like the test databases and machines) that needs to be created for testing.
- Test cases are the basic units of tests. They can be technical or business test cases. They contain, in addition to the steps to be executed in testing, inputs and expected results.
- Test data is created, based on the sampling mechanisms (discussed in detail in Chapter 6), to ensure correct execution of tests—especially the ones with dependencies on another.

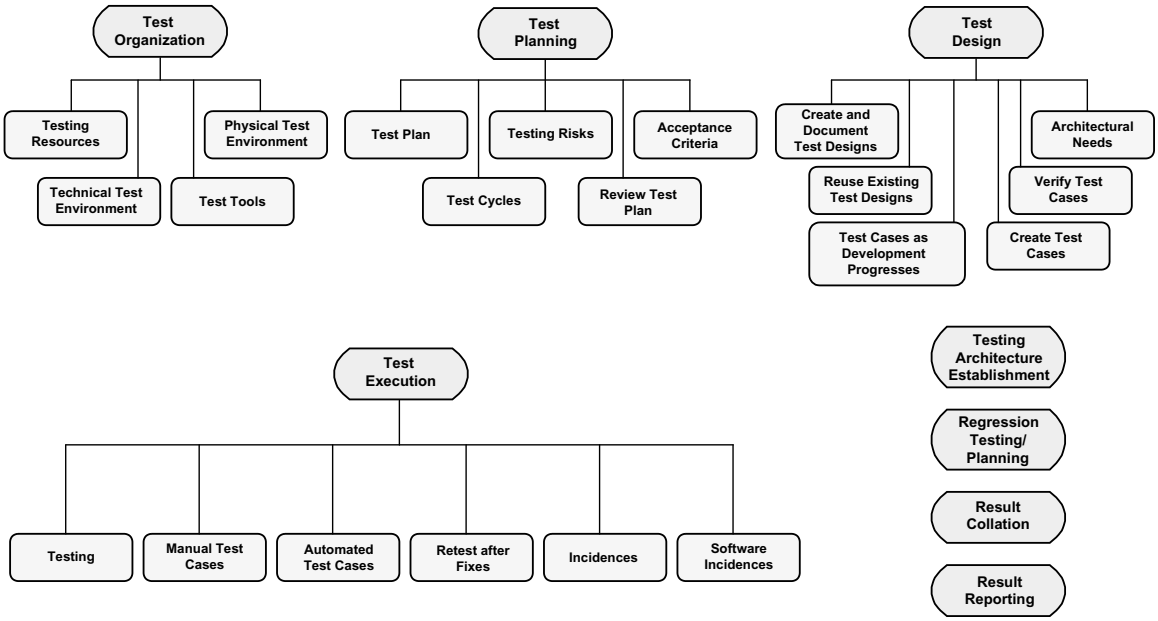


Figure 3.39 *Activities and tasks in quality control*

- Test results are documented, collated, and used for reporting purposes. They can also be used to anticipate areas of the system that need more corrections, or even rewrites.

3.5.15 Quality Comments on Quality Control

Necessity

1. The test plan is necessary for proper testing organization.
2. Test cases are the basic units of tests and have to be created by users, testers, and programmers.
3. Test data is the data on which tests are carried out. This is either the input in a test case or the test database.
4. Test results should be carefully documented and analyzed.
5. The tester is the primary actor in this process-component. The tester is played by more than one person, and depending on

whether it is a technical or business testing, this role is played by a programmer, a business analyst, or a user.

6. The quality manager supports the quality control activities.
7. The model/program finalization is necessary before proper testing commences.
8. The test case documentation must be carried out before testing can proceed.
9. The model/program submission is made to the test manager, who then proceeds with the testing.
10. The test execution is the testing of the model, executable, or whatever artifact is submitted for testing.
11. The result collation is achieved preferably by using a tool to analyze the result, collate, and report.

Sufficiency

1. The test design provides the sufficiency criteria, as good test designs based around the system design ensure that the testing is modular.
2. The test environment focuses on creating a good test environment. It ensures that testing progresses in an orderly manner and takes less time than testing without the proper environment.
3. The modelers, programmer, and user provide all necessary support to the programmer or, at times, play the tester role, as well.
4. Testing the architecture establishment provides the additional impetus to testing by providing a well-organized basis for conducting the tests. This means organizing the databases, software, and operating systems for testing to commence.
5. Testing the approach establishment ensures that all participants in the test teams are aware of the testing approach. This can be, at a high level, a decision to intensely test data, but not functionality, and vice versa.
6. Test environment creation supports the physical creation of the environment.
7. Reporting results is further analysis and reporting of the results from tests carried out.

8. Model/program correction is correcting whatever has been found in error.
9. Regression testing/planning is redoing tests after the corrections have been made by the developers/modelers.

3.6 Bibliographic Notes

The use of process CASE tools is invaluable in deploying a process in a large organization. Check UML CASE Tools for some popular process-based CASE tools.

Most process CASE tools provide their own variation to the process-components as well as the ability to maintain the process. This is done by taking the feedback from the developers and other users of the process, considering the type of the project, and then customizing the process.

Proponents of eXtreme Programming and Agile methodologies may not fully concur with what has been described in this chapter. But this description is scalable and works effectively with a large number of individuals in a project. Combining the nimbleness of Agile methodologies with the process-components described here is my most judicious and favored approach.

3.7 Frequently Asked Questions (FAQs)

Q1: Are there aspects in a process apart from the "what," the "who," and the "how"?

A1: Yes, and they are related to the timing of execution of the process—in other words, the "when." I have described the three parts of a process in order to simplify the understanding of a process. They are not the only aspects in a process. Furthermore, the issues of creating a process environment itself, the corresponding process-based CASE tool, along with the issues of transitioning to a new process, are some important practical considerations to consider. These practical issues are discussed in Chapter 4.

Q2: Is understanding the process metamodel important for process-based work in a project?

A2: The metamodel I have described in Figure 3.2 is only to create a theoretical understanding of how process-components are put together. You can work a process without worrying about the logic behind the metamodel. However, if your role in a project is that of a process engineer, then this understanding of a process metamodel is invaluable in your work of creating a SEP and adopting it. The same applies to the role of a process CASE tool developer.

Q3: How do I apply the process-components?

A3: You don't apply the process-components as they are described in this chapter. Rather, you create a project task plan based on the requirements of your particular project, its size and type, and how you want to iterate within the project by taking help and guidance from the process-components. This is also described in Chapter 4.

Q4: How do the process-components described here relate to the Agile methodologies like eXtreme Programming and Crystal?

A4: Agile methodologies eschew the formal descriptions and possible bureaucracy of complete process. However, in practice, I find it highly advisable to combine the process-components described in this chapter with the principles of Agility.

3.8 Exercises

E3.1: How does a software process differ from a quality process? Describe how together the software process and the quality process form a good QSP.

E3.2: Which process-components would you use in creating the MOPS?

E3.3: Describe the relevance of prototyping process-components in MOSS.

E3.4: Describe why the project management process-component applies to all modeling spaces.

E3.5: Name two important deliverables produced by the business analyst.

E3.6: Name two important deliverables produced by the test manager.

E3.7: Name all the tasks involved in the activity of use case modeling.

E3.8: Name all the tasks involved in the activity of business class modeling.

E3.9: To which activity does the task of "create state chart diagrams" belong? Why, according to you, is the activity named as such?

E3.10: In which process-component is the system actually produced?

E3.11: How does the reuse process-component apply to other process-components where reuse takes place?

E3.12: Where is the incremental and iterative aspect of a process handled?

E3.13: Describe the responsibilities of a training manager.

E3.14: In which types of projects would you not envisage a steering committee?

3.9 References

Jacobson, I., et al. *Object-Oriented Software Engineering: A Use Case Driven Approach,* Reading, Mass.: Addison-Wesley, 1992.