



Development Process Framework

Editor: Richard Veryard
For: SCIPIO Consortium
Status: Beta Version
Version: 0.91
Filename: SCIPIOdpf.pdf

Contents

INTRODUCTION	1
GENERAL	2
PERFORMANCE ASSESSMENT	5
ANALYSIS	7
DESIGN	9
DELIVERY	11
PROJECT SCENARIOS	13
DETAILED TASK STRUCTURE	18

Introduction

This guide describes the tasks of SCIPIO, and indicates their interdependencies. It is intended for three groups of reader:

<ul style="list-style-type: none"> ➤ SCIPIO practitioners, wishing to understand how the concepts and techniques can be applied on projects of various kinds. 	<ul style="list-style-type: none"> ➤ The guide offers several scenarios, to show how SCIPIO may be applied to a range of business, application and technological projects. ➤ The guide shows the purpose of each task in terms of its contribution to parallel or subsequent tasks. ➤ The guide shows the concepts and techniques applicable for each task.
<ul style="list-style-type: none"> ➤ Project managers, needing a basic understanding of the task structure of SCIPIO in order to plan and manage a SCIPIO project 	<ul style="list-style-type: none"> ➤ The guide provides a detailed task structure, which may be used to derive the content of project plans and schedules for a SCIPIO project.
<ul style="list-style-type: none"> ➤ People in various support roles, who need to know what SCIPIO practitioners are likely to be engaged in at different stages of a project in order to provide adequate support. This support may include skills training, software tools, development infrastructure and other resources. 	<ul style="list-style-type: none"> ➤ The guide indicates the knowledge, skills and resources needed at each stage of a SCIPIO project.

Assumptions

This guide does not provide advice about the project planning process itself. We assume that project planning and management will conform to DSDM. If you want to use an alternative method, you must make the appropriate adjustments yourself.

Acknowledgements

Valuable contributions to this document have been made by other members of the SCIPIO Consortium, especially Clive Mabey and Michael Mills.

Sources

SCIPIO draws upon several sources including ISO 9126 (Software Quality), ISO 10746 (ODP Reference Model), the Unified Modelling Language (UML) and DSDM.

General

Overall Structure

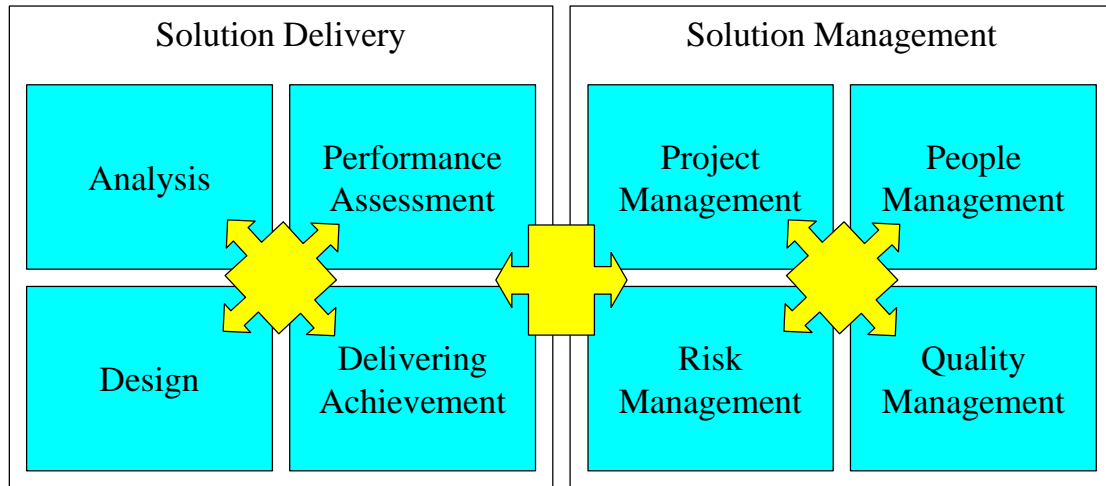


Figure 1: Overall Structure of SCIPIO

Interworking of old and new techniques

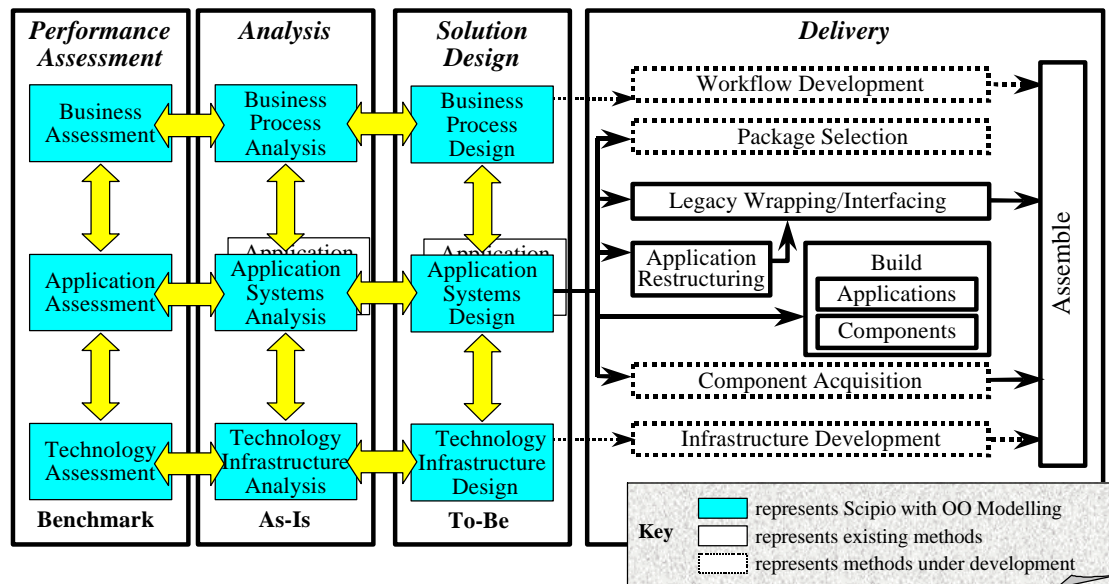


Figure 2: Solution Development Structure

A key feature of the method is that there is some communication or exchange of information between many of the activities. At this high level, no sequence can be inferred, and the diagram should not be read simply from left to right. Note, for example, that the assessment is iterative: some of the performance assessment subtasks can only be done after the relevant portions of the analysis and solution design have been done.

Working in Parallel

The essence of solution provisioning lies in the design of the solution and the planning of transition from the existing situation (As-Is). To illustrate how this might work we take a close look at Analysis and Solution Design. Although there is no single route through the method, one has been chosen for purposes of illustration. This presupposes that a business process is to be improved.

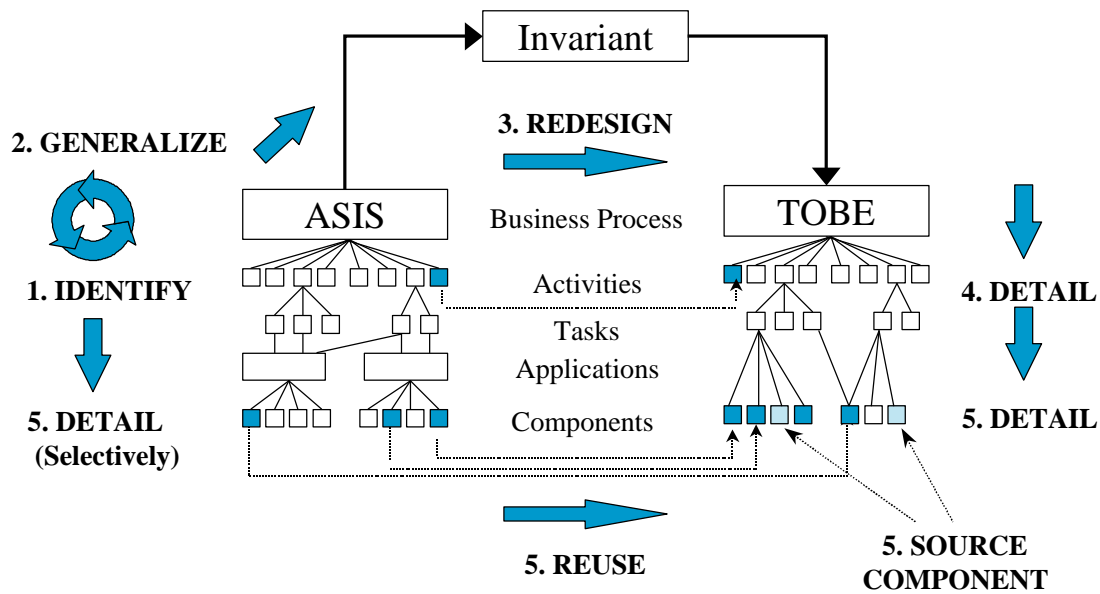


Figure 3: Detail: Analysis and Design Process

In rough terms, the steps are as follows. (A more formal task structure will be presented in a following chapter.)

1. Individual activities are understood through a process of investigation and communication with those who undertake the activity.
2. When sufficient information is available the individual activities are synthesized into a holistic and technology independent view of the business process.
3. Changes to the process are designed bearing in mind available technology and assumed possibilities for reuse. The new process design is recorded in a technology free manner.
4. The process is mapped to the underlying components by a process of decomposition into sub-processes, tasks, manual procedures and software components.
5. As the refinement of the new process proceeds validation is sought for assumptions about reuse of legacy systems. Candidates for reuse are selectively investigated in sufficient detail to establish what role they might play. As candidates are confirmed the design is adjusted to accommodate them. As this is going on component libraries both external and internal are also investigated.

The modelling and design work does not go strictly top-down, from broad structures to detailed specifications, nor strictly bottom-up, from detailed specifications to broad ('strategic') summaries, but middle-out. This can be seen from Figure 3. This means that each task or subtask may be executed at several levels in parallel.

Singular and Plural Tasks

Some tasks refer to a single object being analyzed or designed. For example Analyze Organization, or Map Current Application. In a real project, it may be necessary to carry out these tasks several times. There may be several separate organizations to be analyzed, or several current applications to be mapped. Wherever possible, such tasks are defined in the singular, so that they can be separately scheduled for each object. Metrics for such tasks are defined for a single object, and should be multiplied by the number of objects for which this task is required.

Project Management Concerns

Many methods call for iteration of tasks. A given design technique may be executed at one stage of the project to produce a 'first cut' deliverable, and then may be re-executed at a later stage of the project to produce a 'final' deliverable.

We regard the control of such iteration as a project management concern. For this reason, we do not explicitly specify iteration and related matters within the solution development task lists. Instead, we leave them to be 'plugged in' as part of the project management approach. In other words, the task iterations are to be generated when the project plan is created, based on the preferred project management approach.

In general, we recommend a RAD approach to iteration. The DSDM consortium, which has developed some industry-wide recommendations on RAD, suggests that, to control iteration, each important design technique be iterated three times. However, some situations or organizations may prefer to plug in an alternative approach.

Furthermore, the actual project management tasks, such as project planning, are assumed to be part of the project management plug-in, and are not contained within the Solution Provisioning task lists.

Quality Management Concerns

Similarly, the quality management tasks, such as quality planning, training, reviews and audits, are not contained in the Solution Provisioning task lists but are plugged in as part of the preferred Quality Management approach. The SCIPIO consortium is able to advise on a standard Quality Process, based on ISO 9000, TickIT and the SEI Capability Maturity Model, but SCIPIO is open to alternative quality management approaches being plugged in.

Production and Maintenance

We do not include Production (i.e. routine operation of the improved business process, application and/or infrastructure) within the scope of SCIPIO. However, Production typically spawns a number of minor improvement opportunities. These may either be managed as mini-projects or assembled into an ongoing maintenance program.

Performance Assessment

Overview

Performance Assessment (sometimes referred to as System Assessment) considers the characteristics of the system from several viewpoints, including:

- Business process performance (including cycle time, cost and quality of service);
- Application characteristics (typically the quality characteristics identified by ISO 9126: functionality, maintainability, usability, reliability, efficiency and portability);
- Technological infrastructure characteristics (including value-for money, technical performance and flexibility).

There are six types of performance assessment, which may be considered at different stages of the project. These are:

Evaluation: <i>An estimation or measurement of past or present performance.</i>	Prediction: <i>An estimation or declaration of future performance.</i>
1. Current performance of system. This is typically what triggers the improvement project.	2. Target performance (or objective setting) of system.
3. Benchmark performance of system. Benchmarks may be internal or external.	4. Simulated performance of current system. This is an optional step; it provides a confirmation of our analysis, by showing that our model of the existing system is consistent with the observed performance and is explained by it.
6. Actual performance of redesigned system. This is an evaluation of the success of the project, and may result in further improvement projects.	5. Simulated performance of redesigned system. This provides a prediction of the business benefits of a proposed solution, and an identification of any associated risks. This provides a confirmation that the solution is likely to work.

Thus many of the assessment subtasks can only be carried out after some analysis, design and/or delivery subtasks have been carried out.

Task Structure

Process Assessment	Application Assessment	Infrastructure Assessment
<p>Evaluating the performance of an existing business process, and/or predicting the performance of a proposed business process.</p>	<p>Evaluating the performance of an existing application, and/or predicting the performance and other characteristics of a proposed application.</p>	<p>Evaluating the performance and other characteristics of an existing technological infrastructure, and/or predicting the performance and other characteristics of a proposed technological infrastructure.</p>
<ul style="list-style-type: none"> ➤ Identify Business Process Objectives ➤ Review Performance of Current Business Process against Objectives ➤ Establish Business Process Benchmark ➤ Set Targets for Business Process Improvement ➤ Explain Process Performance ➤ Estimate Performance of Solution (simulation) ➤ Review Performance of Solution (post-implementation evaluation) 	<ul style="list-style-type: none"> ➤ Identify Application Objectives ➤ Review Current Application against Objectives ➤ Establish Application Benchmark ➤ Set Targets for Application Improvement ➤ Explain Application Characteristics ➤ Estimate Characteristics of Solution (simulation) ➤ Review Characteristics of Solution (post-implementation evaluation) 	<ul style="list-style-type: none"> ➤ Identify Infrastructure Objectives ➤ Review Current Infrastructure against Objectives ➤ Review Technology Trends ➤ Establish Infrastructure Benchmark ➤ Set Targets for Infrastructure Improvement ➤ Explain Infrastructure Characteristics ➤ Estimate Characteristics of Solution (simulation) ➤ Review Characteristics of Solution (post-implementation evaluation)

Analysis

Overview

The purpose of Analysis (sometimes known as Diagnosis) is to understand the structure of what already exists, and to identify the causes of any problems or restrictions. A series of models of the current situation is produced, showing how various people, roles, departments, companies and systems interact. These models (known as As-Is models) serve several purposes:

- They provide an explanation of the current level of performance, both in absolute terms and in comparison with benchmarks. They allow the immediate symptoms of performance problems to be traced to their underlying causes.
- They provide a basis for the design of an improved business process, and provide a basis for the possible restructuring and redeployment of parts of the current system to support the redesigned business process.

Task Structure

Process Analysis	Application Analysis	Component Exploration	Infrastructure Analysis
<ul style="list-style-type: none"> ➤ Analyze Organization ➤ Analyze Tasks, Collaborations and Interfaces ➤ Document Workflow Paths and Mechanisms ➤ Document Business Rule Structure ➤ Document Business Responsibilities ➤ Document Process 	<ul style="list-style-type: none"> ➤ Map Current Application ➤ Document Application Components ➤ Document Application Interfaces and Protocols ➤ Document Application Responsibilities ➤ Map Process to Application 	<ul style="list-style-type: none"> ➤ Formulate Search Criteria ➤ Create Shortlist of Available Components ➤ Obtain Specification of Component ➤ Identify Technology Requirements and Constraints 	<ul style="list-style-type: none"> ➤ Map Infrastructure ➤ Document Infrastructure Components ➤ Document Infrastructure Interfaces and Protocols ➤ Map Application to Infrastructure

Process Analysis

This task develops a series of models to describe the existing process. By preference, SCIPIO uses an extension of the UML notation for OO modelling. However, if process models have been produced using previous methods and notations, this work can often be reused.

Application Analysis

This task develops a series of models to describe the structure and functionality of existing applications. By preference, SCIPIO uses an extension of the UML notation for OO modelling. However, if application models have been produced using previous methods and notations (including data models and data flow diagrams), this work can often be reused.

Analysis of legacy systems, although essential for reuse and transition purposes, is often difficult and tedious, especially when documentation is out-of-date, unreliable or completely lacking. SCIPIO therefore undertakes careful planning of legacy systems analysis activity, and ensures that details are produced on a just-as-required basis.

Infrastructure Analysis

This task develops a series of models to describe the existing technology infrastructure. By preference, SCIPIO uses an extension of the UML notation for OO modelling. However, if technology models have been produced using previous methods and notations, this work can often be reused.

Component Exploration

This task finds and analyses existing application packages and components for potential use. It yields a description of the functionality and other characteristics of a package or component, defined in terms of its vocabulary and behaviour.

It uses a requirements specification as input, which may be based on a preliminary design. This specification need not be complete: indeed, it is normal to carry out a preliminary search for available packages or components with a rough and partial specification.

Components may be found in existing in-house component libraries, or via the World Wide Web.

Design

Overview

Solution Design includes the **definition** of the solution as well as its **planning and sourcing**. The improved (To-Be) business process is represented in the same way as the current (As-Is) business process: as a set of interactions between people, roles, departments, companies and systems.

- The interactions may be improved and made more effective.
- Wholly new components may be introduced into the system. These may be business components, application components or technological components.
- Existing components may be adapted to fit the improved system context in which they are to operate. Opportunities are identified to make the existing components and their interfaces more generic, to enhance future adaptability.
- The business rules are reformed and mapped onto the components. Each component is assigned the responsibility for maintaining one or more rules.
- A holistic picture of the solution is produced.
- An implementation plan is developed, indicating the selected source for each new or modified component.

Task Structure

Process Design	Application Design	Infrastructure Design
➤ Determine Optimization Strategy	➤ Information Sourcing	➤ Envision Solution
➤ Business Rule Rationalization	➤ Component Identification	➤ Develop Upgraded Technical Architecture
➤ Business Responsibility Assignment	➤ Component Protocol Design	➤ Determine Re-use Opportunities
➤ Workflow Structure Design	➤ Component Responsibility Assignment	➤ Allocate Components to Platforms
➤ Workflow Mechanism Selection	➤ Data Storage Design	
➤ Solution Verification	➤ Component Specification	
➤ Release Planning	➤ Component Sourcing	

Process Design

This task uses an extension of UML for specifying the future process.

Application Design

This task uses an extension of UML for specifying the future application as a set of collaborating components. It yields a description of the solution as a combination of new application components and existing portions of the legacy system (which may require some wrapping or modification to work properly as components).

We can also link to traditional techniques, especially during the transition from previous tools and skills. For example, we can use entity-relationship modelling and process decomposition to specify the functional requirements of the future application. We can also use dialog flow diagramming and window design to specify the user interface and implementation of the future application. Such links to traditional techniques also allow for continuity and reuse of existing design work.

Infrastructure Design

This task uses an extension of UML for specifying the future technology infrastructure as a set of collaborating components.

Delivery

Overview

Delivery divides into **Component Acquisition, Component Development, Assembling** and **Commissioning**. The provisioning of the new components may follow a variety of paths, including:

- Existing components may be wrapped, repackaged or restructured.
- Modifications to existing components may be hand-crafted. Alternatively, if a suitable component already exists elsewhere, it may be cheaper and easier to replace the old component with an improved version satisfying the requirements.
- New components may be bought 'off-the-shelf' from a component library. Or a component with similar description may be bought and customized.
- Finally, and only where necessary, new components may be commissioned from component builders, either in-house or external.

These considerations apply to the provisioning not only of software application components but also of other pieces of the solution, including business process components (such as user guides and stationery) and infrastructure components (such as hardware and system software).

The development and maintenance of components will normally follow a prototyping approach. Furthermore, whole business solutions may be assembled and piloted before roll-out to the remainder of the target organization - this can be regarded as a business-level equivalent of a prototyping approach.

Note: the planning of development and implementation is regarded as part of project management.

Task Structure

Task	Description	Subtasks
Component Acquisition	Obtain application and technology components from external sources. Includes components supplied by customer.	<ul style="list-style-type: none"> ➤ Negotiate Ownership ➤ Take Delivery ➤ Verify Acquired Component
Acquired Component Implementation	This task includes the modification and installation of components, packages and templates from external sources (including customer).	<ul style="list-style-type: none"> ➤ Modify Component Design ➤ Set Component Parameters
New Component Construction	This task includes the construction of new components.	<ul style="list-style-type: none"> ➤ Build Component ➤ Unit-Test Component
Transition Component Construction	This task consists of wrapping and/or modifying legacy code to produce application components that offer the required interface/service. The legacy code usually (but not necessarily) remains on the original platform.	<ul style="list-style-type: none"> ➤ Wrap Legacy code ➤ Modify Legacy code
Component Publication	Make component (and/or the services it provides) available to a defined user community.	<ul style="list-style-type: none"> ➤ Define Terms of Availability ➤ Publish Component Specification ➤ Establish Component Delivery Mechanism
Solution Testing	Verifies that the solution satisfies the requirements, executes satisfactorily on the installed technological platform, and provides the expected support for the business process.	<ul style="list-style-type: none"> ➤ Define Test Plan ➤ Establish Test Environment ➤ Install Test Data Storage ➤ Execute Test Plan
Solution Deployment	Installs all new technology components. Configures technology infrastructure. Installs all application components on the appropriate technology components. Installs process components. Go live.	<ul style="list-style-type: none"> ➤ Install Components ➤ Configure System ➤ Install Live Data Storage ➤ Commission System

Project Scenarios

SCIPIO covers a broad range of project scenarios from application assembly to BPR. In this chapter, we identify four main solution-oriented scenarios, each with some variations.

- Business Process Improvement
- Reengineering
- Object Development from Scratch
- Middleware

SCIPIO identifies three different flavors of improvement, at each of the three levels (business process, application and technology). This is based on the wisdom that it usually makes sense (at all three levels) to simplify before integrating (“Don’t pave the cow paths!”), and to integrate before transforming. The four scenarios cover all nine variations, as shown in Figure 4.

<p>Business Process</p>	<p>Simplification</p> <p>Using IT to eliminate redundant steps in a business process, or to reduce excessive variety.</p>	<p>Integration</p> <p>Using workflow management tools and new software components to link business processes together more effectively.</p> <p style="text-align: center;"><i>Scenario: Business Process Improvement</i></p>	<p>Transformation</p> <p>Radical BPR</p>
<p>Application</p>	<p>Simplification</p> <p>Using CBD in a tightly focused way to eliminate specific problems in legacy systems.</p> <p style="text-align: center;"><i>Scenario: Reengineering</i></p>	<p>Integration</p> <p>Using middleware to link information systems together more effectively.</p>	<p>Transformation</p> <p>Building new Information System solutions using commercially available components.</p> <p style="text-align: center;"><i>Scenario: Object Development from Scratch</i></p>
<p>Infrastructure</p>	<p>Simplification</p> <p>Replacing unnecessary complications or variations in the technological infrastructure.</p> <p style="text-align: center;"><i>Scenario: Middleware</i></p>	<p>Integration</p> <p>Linking heterogeneous technologies together more effectively.</p>	<p>Transformation</p> <p>Satisfying new technical requirements, or exploiting new technological opportunities.</p> <p style="text-align: center;"><i>Scenario: Object Development from Scratch</i></p>

Figure 4: Four Project Scenarios, covering nine kinds of improvement

Scenario: Business Process Improvement

Process Simplification <i>OR</i>	Process Integration <i>OR</i>	Process Transformation	SCIPPIO Tasks
Start with an existing business process, whose performance is problematic in some way - too expensive, too slow, too error-prone. <i>OR</i>	Start with a series of poorly coordinated business activities. ----->	----->	<i>Process Assessment</i>
<i>OR</i>	----->	Start with an entirely new process concept.	<i>Determine Optimization Strategy</i>
Model the as-is process, to identify which specific subprocesses and components are causing the problems.	Model the current mechanisms (if any) that link these activities.		<i>Process Analysis</i> <ul style="list-style-type: none"> • <i>Analyse Organization</i> • <i>Analyse Tasks, Collaborations and Interfaces</i> • <i>Document Workflow Paths and Mechanisms</i> • <i>Document Business Rule Structure</i> • <i>Document Business Responsibilities</i>
Produce an abstract model to show what is essential about the process. <i>OR</i>	Create an overall business process that brings all these activities together. ----->	Analyse the new concepts to identify its key components (at the business level).	<i>Process Analysis</i> <ul style="list-style-type: none"> • <i>Document Process</i> <i>Process Design</i> <ul style="list-style-type: none"> • <i>Business Responsibility Assignment</i>
Prune the process to cut out whatever is causing the problems, and replace with simpler, more robust components.	Design an improved mechanism for linking these activities. This may typically involve workflow management software.		<i>Process Design</i> <ul style="list-style-type: none"> • <i>Determine Optimization Strategy</i> • <i>Business Rule Rationalization</i> • <i>Business Responsibility Assignment</i> • <i>Workflow Structure Design</i> • <i>Workflow Mechanism Selection</i>

<i>Process Simplification</i> <i>OR</i>	<i>Process Integration</i> <i>OR</i>	<i>Process Transformation</i>	<i>SCIPIO Tasks</i>
			<i>Application Design (first cut)</i> <ul style="list-style-type: none"> • <i>Information Sourcing</i> • <i>Component Identification</i> • <i>Component Protocol Design</i> • <i>Component Responsibility Assignment</i> • <i>Component Specification</i> • <i>Component Sourcing</i>
Analyse the existing systems to discover how many of the required components already exist.	Analyse the existing systems to discover how many of the required components already exist.	Analyse the existing systems to discover how many of the required components already exist.	<i>Application Analysis</i> <ul style="list-style-type: none"> • <i>Document Application Components</i> <i>Application Design (second cut)</i>
Estimate the performance of the redesigned process.	Estimate the performance of the redesigned process.	Estimate the performance of the new process.	<i>Process Assessment</i> <ul style="list-style-type: none"> • <i>Estimate performance of solution (simulation)</i> <i>Application Assessment</i> <ul style="list-style-type: none"> • <i>Estimate characteristics of solution (simulation)</i> <i>Application Design (third cut)</i>
Implement the new components one by one, to yield a simpler, more robust process with the desired performance.	Modify existing components and/or acquire new components to work within the improved mechanism. Reconfigure the business process.	Acquire new components as required. Install new components. Implement new business process.	<i>Process Design</i> <ul style="list-style-type: none"> • <i>Release Planning</i> <i>Application Design</i> <ul style="list-style-type: none"> • <i>Component Sourcing</i> <i>Delivery</i>

Scenario: Reengineering

<i>Application Simplification</i>	<i>Application Integration</i>	<i>SCIPIO Tasks</i>
Start with an existing application that is problematic in some way - fails to satisfy important business requirements, too expensive or too slow to maintain, too error-prone.	Start with a series of poorly coordinated computer systems. There may be high levels of data duplication, complex data conversions, or other inefficiencies.	<i>Application Assessment</i>
Model the as-is application, to identify which specific subprocesses and components are causing the problems.	Identify the current mechanisms (if any) that link these systems. (The current mechanisms will often be manual - in the worst case, they may involve rekeying data from one computer system into another computer system.)	<i>Application Analysis</i>
Prune the application to cut out whatever is causing the problems, and replace with simpler, more robust components.	Design an overall application architecture that brings all these systems together. Design improved mechanisms for linking these systems. This may typically involve open distributed computing.	<i>Application Design</i>
Implement the new components one by one, to yield a simpler, more robust application with the desired characteristics.	Modify existing components and/or acquire new components to work with the chosen mechanism. Reconfigure the applications.	<i>Delivery</i>

Scenario: Middleware

<i>Infrastructure Simplification</i>	<i>Infrastructure Integration</i>	<i>SCIPIO Tasks</i>
Start with an existing infrastructure that is problematic in some way - fails to satisfy important business requirements, too expensive or too slow to maintain, too error-prone.	Start with a series of poorly coordinated platforms and protocols.	<i>Infrastructure Assessment</i>
Model the as-is infrastructure, to identify which specific platforms and protocols are causing most of the problems.	Identify the current mechanisms (if any) that link these platforms and protocols.	<i>Infrastructure Analysis</i>
Rationalize the infrastructure to phase out whatever is causing the problems, and replace with simpler, more robust technologies.	Design an overall technical architecture that brings all these systems together.	<i>Infrastructure Design</i> <ul style="list-style-type: none"> • <i>Envision Solution</i> • <i>Develop Upgraded Technical Architecture</i> • <i>Determine re-use opportunities</i>
Implement the new components one by one, to yield a simpler, more robust infrastructure with the desired characteristics.	Introduce new middleware to implement the technical architecture. Modify existing components and/or acquire new components to work with the chosen mechanism. Reconfigure the infrastructure.	<i>Infrastructure Design</i> <ul style="list-style-type: none"> • <i>Develop Upgraded Technical Architecture</i> • <i>Determine re-use opportunities</i> <i>Delivery</i>

Scenario: Object Development from scratch

<i>Application Transformation</i>	<i>Infrastructure Transformation</i>	<i>SCIPIO Tasks</i>
Start with an entirely new application requirement.	----->	<i>Process Design</i>
		<ul style="list-style-type: none"> • <i>Determine Optimization Strategy</i>
<i>OR</i>	Start with an entirely new technological requirement or opportunity.	<i>Infrastructure Design</i>
		<ul style="list-style-type: none"> • <i>Envision Solution</i>
Analyse the new requirement to identify its key components.	Analyse the new requirement or opportunity to identify its key components.	<i>Process Design</i>
Analyse the existing systems to discover how many of these components already exist.	Analyse the existing systems to discover how many of these components already exist (probably externally).	<i>Application Analysis</i>
		<ul style="list-style-type: none"> • <i>Component Identification</i>
Acquire new components as required.	Acquire new components as required.	<i>Application Design</i>
		<ul style="list-style-type: none"> • <i>Information Sourcing</i> • <i>Component Identification</i> • <i>Component Protocol Design</i> • <i>Component Responsibility Assignment</i> • <i>Component Specification</i>
Install new components.	Install new components.	<i>Delivery</i>
Implement new application.	Implement new infrastructure.	<i>Delivery</i>

Detailed Task Structure

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
Process Assessment	Identify Business Process Objectives	Identify all the business objectives for the process.	To provide a comprehensive basis for assessment.	Include <ul style="list-style-type: none"> constraints (what cannot be changed) opportunities (what may be changed) goals (what must be changed). 	Input from process owner Assumes that the scope of the business process has been (at least provisionally) defined.	Describe Business Process Objective Determine Business Process Metric	For each objective: description how it is measured
	Review Performance of Current Business Process against Objectives	Measure or estimate performance of process separately against each objective. Document management attitude to any identified process weaknesses.	To identify the priorities for improvement.	Consider the strengths of the current business process, as well as its weaknesses. Does the process perform consistently, or are there peaks and troughs of performance? Which of the identified weaknesses are most critical to the business? Which must be addressed most urgently?	Business Process Objectives Performance Data if available. Otherwise direct observation of the business process for measurement.	Assemble Available Performance Data Conduct Performance Measurements Determine Current Performance Level Determine Current Performance Variations Assess Performance Adequacy Assess Priority of Action	For each objective: <ul style="list-style-type: none"> current level achieved (with any important variations) judgment of adequacy judgment of priority

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
	Establish Business Process Benchmark	Identify external or internal best-in-class for this process. Measure their achievements against our objectives.	To stimulate improvement effort. To identify realistic project targets. And to identify specific practices that can be considered for adoption.	Are the results obtainable from published figures, or from a benchmarking club, or do they have to be estimated? How reliable is the estimate? How much do we know (or can find out) about how they actually achieve these results? How fair is the comparison with the benchmark?	Business Process Objectives	Define Benchmark Strategy and Scope Identify Benchmark Targets Establish External Contacts Define Benchmark Checklist Obtain Benchmark Information Obtain Collateral Information Identify Issues and Opportunities Analyze Benchmark Results	For each benchmark and objective: <ul style="list-style-type: none"> • current level achieved (with any important variations) • any significant differences between their situation and ours • objectives • priorities • constraints
	Set Targets for Business Process Improvement	Define goals for the project in terms of the business process objectives.	Required for project management.	Consider what has been achieved elsewhere, when setting targets.	Business Process Objectives Current Business Process Performance Business Process Benchmarks	Determine Achievable Performance Level Estimate Costs and Benefits of Performance Improvement Determine Cost-Effective Performance Level Set Improvement Timescale	For each objective: <ul style="list-style-type: none"> • how much improvement? • by when?

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
	Explain process performance.	When the business process has been modelled, check that the AS IS model is consistent with and explains the current performance.	Validates the analysis.	Is all process cycle time accounted for? Is all the resource expenditure accounted for? Can the actual quality of service be explained in terms of the AS IS model?	AS IS models (from Analysis)	Establish Simulation Approach (manual / automated) Develop Simulation Model Execute Simulation Model Analyze Simulation Results	The simulation of the AS IS model should yield results consistent with the actual process performance. This task may result in changes to the AS IS model.
	Estimate performance of solution (simulation)	When the process has been redesigned, estimate its performance against the business process objectives. In complex or high-risk projects, simulation tools may be used.	Validates the solution before you start building it.	Is the new process going to work? What is it likely to achieve against the business process objectives? Will the improvement targets be met?	TO BE models (from Design)	Establish Simulation Approach (manual / automated) Develop Simulation Model Execute Simulation Model Analyze Simulation Results	The simulation of the TO BE model should yield results consistent with the project targets. This task may result in changes to the TO BE model, and/or to the project targets.
	Review performance of solution (post-implementation evaluation)	When the redesigned process has been implemented, measure the actual performance achieved against each of the business process objectives.	Evaluates the success of the project. Identifies lessons and opportunities for future projects.	Is the new process working? Have improvement targets been met?	Project targets Actual measurements of the improved business process.	Check solution behaviour Measure solution performance Compare solution performance with improvement targets Identify outstanding actions and opportunities	This task may result in a further improvement cycle, and/or to changes in the solution development process itself.

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
Application Assessment	Identify Application Objectives	<p>Identify all the objectives for the application.</p> <p>Consider what the business or technology may require from applications in future as well as the current requirements</p>	To provide a comprehensive basis for assessment.	<p>Include</p> <ul style="list-style-type: none"> constraints (what cannot be changed) opportunities (what may be changed) goals (what must be changed). <p>Consider all aspects of application quality and performance, including: functionality, usability, maintainability, efficiency, reliability and portability.</p>	<p>In some projects, the objectives may be derived from the business process objectives. On other projects, they will be given as part of the terms of reference.</p> <p>Assumes that the scope of the application has been (at least provisionally) defined.</p>	<p>Describe Application Objective</p> <p>Determine Application Quality Metric</p>	<p>For each objective:</p> <ul style="list-style-type: none"> description how it is measured
	Review Current Application against Objectives	Measure or estimate quality of the application separately against each objective.	To identify the priorities for improvement.	<p>Consider the strengths of the current application, as well as its weaknesses.</p> <p>Does the application perform consistently, or are there peaks and troughs of performance?</p> <p>Which of the identified weaknesses are most critical to the business? Which must be addressed most urgently?</p>	<p>Application Objectives</p> <p>Performance Data if available. Otherwise direct measurement of the application.</p>	<p>Assemble Available Quality Data</p> <p>Conduct Quality Measurements</p> <p>Determine Current Quality Level</p> <p>Determine Current Quality Variations</p> <p>Assess Application Adequacy</p> <p>Assess Priority of Action</p>	<p>For each objective:</p> <p>current level achieved (with any important variations)</p> <p>judgment of adequacy</p> <p>judgment of priority</p>

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
	Establish Application Benchmark	Identify any equivalent application (either elsewhere in the company or outside), with which the quality of this application can be compared. Measure their achievements against our objectives.	To stimulate improvement effort. To identify realistic project targets. And to identify specific practices that can be considered for adoption.	Are the results obtainable from published figures, or from a benchmarking club, or do they have to be estimated? How reliable is the estimate? How much do we know (or can find out) about how they actually achieve these results? How fair is the comparison with the benchmark?	Application Objectives	Define Benchmark Strategy and Scope Identify Benchmark Targets Establish External Contacts Define Benchmark Checklist Obtain Benchmark Information Obtain Collateral Information Identify Issues and Opportunities Analyze Benchmark Results	For each benchmark and objective: current level achieved (with any important variations) any significant differences between their situation and ours <ul style="list-style-type: none"> • objectives • priorities • constraints
	Set Targets for Application Improvement	Define goals for the project in terms of the application objectives.	Required for project management.	Consider what has been achieved elsewhere, when setting targets. Relate application improvement targets to business process improvement targets.	Application Objectives Business Process Improvement Targete Application Benchmark Results	Determine Achievable Quality Level Estimate Costs and Benefits of Quality Improvement Determine Cost-Effective Quality Level Set Improvement Timescale	For each objective: how much improvement? by when?

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
	Explain Application Characteristics	When the application has been modelled, check that the model is consistent with and explains the current quality of service.	Validates the analysis	Are all the response time and processing delays accounted for? Is all the resource expenditure accounted for? Can the actual quality of service be explained in terms of the model?	AS IS models (from Analysis)	Establish Simulation Approach (manual / automated) Develop Simulation Model Execute Simulation Model Analyze Simulation Results	This task may result in changes to the AS IS model.
	Estimate characteristics of solution (simulation)	When the application has been redesigned, estimate its behaviour against the application objectives. In complex or high-risk projects, simulation tools may be used.	Validates the solution before you start building it.	Is the new application going to work? What is it likely to achieve against the application objectives? Will the improvement targets be met?	TO BE models (from Design)	Establish Simulation Approach (manual / automated) Develop Simulation Model Execute Simulation Model Analyze Simulation Results	This task may result in changes to the TO BE model.
	Review characteristics of solution (post-implementation evaluation)	When the redesigned application has been implemented, measure the actual behaviour achieved against each of the application objectives.	Evaluates the success of the project. Identifies lessons and opportunities for future projects.	Is the new application working? Have improvement targets been met?	Project targets Actual measurements of the improved application.	Check solution behaviour Measure solution performance Compare solution performance with improvement targets Identify outstanding actions and opportunities	This task may result in a further improvement cycle, and/or to changes in the solution development process itself.

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
Infrastructure Assessment	Identify Infrastructure Objectives	<p>Identify all the objectives for the infrastructure.</p> <p>Consider what the business or applications may require from technology in future as well as the current requirements</p>	To provide a comprehensive basis for assessment.	<p>Include</p> <ul style="list-style-type: none"> constraints (what cannot be changed) opportunities (what may be changed) goals (what must be changed). <p>Consider all aspects of system quality, including functionality, usability, maintainability, efficiency, reliability and portability.</p>	In some projects, these objectives may be derived from the business process or application objectives. On other projects, they will be given as part of the terms of reference.	<p>Describe Infrastructure Objective</p> <p>Determine Infrastructure Quality Metric</p>	<p>For each objective: description</p> <p>how it is measured</p>
	Review Current Infrastructure against Objectives	Measure or estimate quality of the infrastructure separately against each objective.	To identify the priorities for improvement.	<p>Consider the strengths of the current infrastructure, as well as its weaknesses.</p> <p>Does the technology perform consistently, or are there peaks and troughs of performance?</p> <p>Which of the identified weaknesses are most critical to the business? Which must be addressed most urgently?</p>	Infrastructure Objectives	<p>Assemble Available Quality Data</p> <p>Conduct Quality Measurements</p> <p>Determine Current Quality Level</p> <p>Determine Current Quality Variations</p> <p>Assess Infrastructure Adequacy</p> <p>Assess Priority of Action</p>	<p>For each objective:</p> <ul style="list-style-type: none"> current level achieved (with any important variations) judgment of adequacy judgment of priority
	Review Technology Trends	Consider the likely future capabilities of the technologies currently being used.	To identify future opportunities and threats	Are these expected capabilities likely to be available within a timeframe to satisfy future requirements?	Infrastructure Objectives		

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
	Establish Infrastructure Benchmark	Identify any equivalent infrastructure (either elsewhere in the company or outside), with which the quality of this infrastructure can be compared. Measure their achievements against our objectives.	To stimulate improvement effort. To identify realistic project targets. And to identify specific practices that can be considered for adoption.	Are the results obtainable from published figures, or from a benchmarking club, or do they have to be estimated? How reliable is the estimate? How much do we know (or can find out) about how they actually achieve these results? How fair is the comparison with the benchmark?	Infrastructure Objectives	Define Benchmark Strategy and Scope Identify Benchmark Targets Establish External Contacts Define Benchmark Checklist Obtain Benchmark Information Obtain Collateral Information Identify Issues and Opportunities Analyze Benchmark Results	For each benchmark and objective: <ul style="list-style-type: none"> • current level achieved (with any important variations) • any significant differences between their situation and ours <ul style="list-style-type: none"> • objectives • priorities • constraints
	Set Targets for Infrastructure Improvement	Define goals for the project in terms of the infrastructure objectives.	Required for project management.	Consider what has been achieved elsewhere, when setting targets. Relate infrastructure improvement targets to business process and application improvement targets.	Infrastructure Objectives Infrastructure Benchmark Results	Determine Achievable Quality Level Estimate Costs and Benefits of Quality Improvement Determine Cost-Effective Quality Level Set Improvement Timescale	For each objective: <ul style="list-style-type: none"> • how much improvement? • by when?

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
	Explain Infrastructure Characteristics	When the infrastructure has been modelled, check that the model is consistent with and explains the current quality of service.	Validates the analysis	Are all the response time and processing delays accounted for? Is all the resource expenditure accounted for? Can the actual quality of service be explained in terms of the model?	AS IS model	Establish Simulation Approach (manual / automated) Develop Simulation Model Execute Simulation Model Analyze Simulation Results	This task may result in changes to the AS IS model.
	Estimate characteristics of solution (simulation)	When the infrastructure has been redesigned, estimate its behaviour against the application quality objectives. In complex or high-risk projects, simulation tools may be used.	Validates the solution before you start building it.	Is the new infrastructure going to work? What is it likely to achieve against the infrastructure objectives? Will the improvement targets be met?	TO BE model	Establish Simulation Approach (manual / automated) Develop Simulation Model Execute Simulation Model Analyze Simulation Results	This task may result in changes to the TO BE model.
	Review characteristics of solution (post-implementation evaluation)	When the redesigned infrastructure has been implemented, measure the actual behaviour achieved against each of the infrastructure objectives.	Evaluates the success of the project. Identifies lessons and opportunities for future projects.	Is the new infrastructure working? Have improvement targets been met?	Delivered Solution	Check solution behaviour Measure solution performance Compare solution performance with improvement targets Identify outstanding actions and opportunities	This task may result in a further improvement cycle, and/or to changes in the solution development process itself.

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
Process Analysis	Analyze Organization	Understand the actual structure of the organization.	Identifies the internal stakeholders in the current process, and their responsibilities and relationships. This will be important for planning any changes.	Do roles correspond to persons or departments? Who controls whom? What are the organizational patterns: <ul style="list-style-type: none"> • management by objectives • self-managing teams 	Scope (provisionally) defined	Identify people and groups Identify roles Identify organizational relationships	Organizational Structure Diagram RAEW matrix
	Analyze Tasks, Collaborations and Interfaces	Understand the activities of the organization.	Since the current mechanisms for communication between activity steps are likely to change, we need a model of the process that is independent of mechanisms and sequence.	For each activity, identify the collaborating actors: <ul style="list-style-type: none"> • human actor • external relationship • internal relationship • upstream activity • downstream activity • technology support for activity 	Scope (provisionally) defined	Decompose Process into Sub-Processes Decompose Sub-Processes into Activities Detail Collaborations Analyze interfaces between organization units Analyze interfaces between tasks Analyze user interfaces	Collaboration or Exchange Diagram

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
	Document Workflow Paths and Mechanisms	Analyze how the separate activity steps are joined together to make a complete process. Process control tasks should be included, as well as process execution tasks.	Performance problems in a process (whether to do with speed or reliability) can often be traced to weak communication between activity steps, rather than to weak performance of a given step.	How are messages passed between steps of the process? Where are the queues in the process? How are they managed? How is a given work item allocated to one of many interchangeable servers or caseworkers? How is the progress and completion of work controlled?		Identify exchanges Decompose exchanges into separate flows Document flow mechanisms and protocols	Activity Diagram (Process Flow)
	Document Business Rule Structure	Identify the business rules actually implemented by the current work and procedures.	Allows business rules to be explicitly revalidated and rationalized before being re-implemented in the new solution.	What are the implicit objectives and constraints of each task or component? What is supposed to happen for each logical combination of conditions?		Identify explicit and implied rules for each role. Identify explicit and implied rules for each component. Consolidate rules into rule diagram.	Rule Diagram
	Document Business Responsibilities	Identify where the responsibility lies for guaranteeing each rule	Identifies overlaps and gaps in the current implementation of the business rules Helps the analyst to ensure continuity of rule-governed behaviour during the transition to the new solution.	How is each rule enforced?	Rule Diagram Organizational Structure Diagram		Matrix mapping between Rule Diagram and Organizational Structure Diagram

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
	Document Process	Understand the actual overall structure of the process.	Provides a high-level summary of the process, which should be meaningful to business management.		This task may be started at any time, but cannot be completed until the other tasks in this section are complete.		Activity Diagram
Application Analysis	Map Current Application	Identify subsystems of the current application, to be documented as components.	Provides starting point for planning improvements to the application	How reliable is system documentation? Does system documentation identify meaningful clusters of legacy application? Can more meaningful clusters be identified after some detailed analysis?			Collaboration Diagram: Application Map
	Document Application Components	Describe the functionality and performance of each identifiable subsystem of the current application systems	Helps determine which components can be redeployed in the new or improved application, and which components need to be altered or replaced.			Abstract collaborations to single actions. Abstract groups to single objects. Specify abstract actions with postconditions.	Class Diagrams
	Document Application Interfaces and Protocols	Identify how the application components talk to each other at runtime.	Helps fix the interfaces between the components that are going to remain and the components that are going to be replaced.	Are there performance or quality of service assumptions / constraints on the interfaces?			Sequence Diagram

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
	Document Application Responsibilities	Identify which business rules are currently encapsulated in software	Identifies overlaps and gaps in the current implementation of the business rules. Helps the analyst to ensure continuity of rule-governed behaviour during the transition to the new solution.		Rule Diagram Application Map		Matrix mapping between Rule Diagram and Application Map
	Map Process to Application	Understand the relationship between the actual structure of the process and the actual structure of the current applications.	Helps to justify changes to the application in terms of improvements to the business process.		Process Flow Diagram Application Map		Matrix mapping between Process Flow Diagram and Application Map
Component Exploration	Formulate Search Criteria	Define the procurement requirements.	Understand what we're looking for.				Search Criteria
	Create Shortlist of Available Components	Determine the names, sources and costs of components that match the search criteria.	Limit the search to components that might be relevant to the solution.		Search Criteria		Component Shortlist
	Obtain Specification of Component	Get details of the behaviour and vocabulary of the component.	Enables the designer to create detailed designs of solutions that would include this component.		Component Shortlist		Ideally we can produce a Class Diagram for the component.
	Identify Technology Requirements and Constraints	Get details of the operational prerequisites of the component.	Understand the practicalities of acquiring this component, and the costs of using it.		Component Shortlist		Running costs. Preconditions for installation.

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
Infrastructure Analysis	Map Current Infrastructure	Identify parts of the current infrastructure, to be documented as infrastructure components.	Provides starting point for planning improvements to the infrastructure	As for application map			Collaboration Diagram: Infrastructure Map As Is
	Document Infrastructure Components	Describe the functionality and performance of each chunk of the current infrastructure	Helps determine which components can be redeployed in the new or improved application, and which components need to be altered or replaced.				Type Diagrams
	Document Infrastructure Protocols	Identify how the infrastructure components talk to each other at runtime.	Helps fix the interfaces between the components that are going to remain and the components that are going to be replaced.	Sequence diagrams address information flow behaviour requirement scenarios for centralized access, distributed access, and connection-free or on-demand access.			Sequence Diagram
	Map Application to Infrastructure	Understand the relationship between the actual structure of the current application and the actual structure of the current infrastructure.	Helps to justify changes to the infrastructure in terms of improvements to the business process and/or application.		Application Map Infrastructure Map		Matrix mapping between Application Map and Infrastructure Map
Process Design	Determine Optimization Strategy	Identify the process improvement pattern(s) to be applied.	Establishes a coherent style for the design. Provides consistent direction to individual designers.	for example: Simplification, Generalization, Integration			

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
	Business Rule Rationalization	Restructure or simplify the Rule Diagram.	Ensures that essential rules are implemented reliably and cost-effectively.	Are all business rules still appropriate? Are there any rules that cost more to enforce than is justified by their value to the business? Are any business rules duplicated or redundant?	Start from Rule Diagram (As Is)		Rule Diagram (To Be)
	Business Responsibility Assignment	Identify where the rules will be implemented.	Ensures the solution covers the business rules completely and cost-effectively.	Does each activity have a meaningful cluster of responsibilities?	Rule Diagram Process Flow Diagram		Mapping between Rule Diagram and Process Flow Diagram
	Workflow Structure Design	Identify the flows of work between tasks and organization units.	Ensure that the solution is sufficiently flexible. Provide basis for acceptance testing and user training. Establish control model for workflow engine (if appropriate).	Is it necessary or appropriate to fix the workflow? How much flexibility is possible?	Start from Process Flow Diagram (As Is)		Process Flow Diagram (To Be)
	Workflow Mechanism Selection	Identify how the work will flow.	In some situations, it is appropriate to specify mechanisms in advance. In many situations, it is necessary to estimate required communication capacity (both human and technological).	Are selected mechanisms robust enough? Are contingency mechanisms required?	Workflow Structure Design		Process Flow Diagram (To Be) Sequence Diagram (To Be)

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
	Solution Verification	Check solution meets requirements.	Reduce risk that the project fails to meet its objectives. Check solution before investing in acquisition or development.	Concentrate on the components whose acquisition involves the greatest expected leadtime, cost or risk.			This task may result in changes to any of the diagrams representing the solution.
	Release Planning	Cluster the changes to the process into several self-contained steps or releases.	Ensure the solution can be easily implemented with limited disruption to ongoing business and technical operations.	Consider other business events with which the implementation may need to be coordinated.			Annotated Process Flow Diagram and Type Diagram for each release. Release Schedule
Application Design	Information Sourcing	Defines where the application will obtain data from, either at install-time or at run-time.	Ensure completeness and correctness of information to support business needs.	What are the key queries? What data are currently available to support these queries?	Data Stores Information Needs		Matrix mapping between queries and data stores.
	Component Identification	Identifies the components from which the application will be constructed. The various tasks that the software actors perform are candidate actions for software components. We can detail a component in terms of its operations and vocabulary. This is the external design.	To confirm that the external design is at a sufficiently fine level of granularity to be implemented as a single component, it is necessary to further decompose its actions.	Is the external design at a sufficiently fine level of granularity to be implemented as a single component? Consider how the lower level items, or interface types, might be packaged into components.	Components in existing system. These may be explicit (actual) or hidden (potential). Components readily available from other sources.	Identify External Design of Application Component Decompose the Actions Package Actions into Implementable Components	List of components, with outline description of functionality and other requirements

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
	Component Protocol Design	Defines how components will talk to each other.	Needed for specifying component interfaces.	Are there standard patterns or protocols that can be used across the project?	Component List		Sequence Diagram
	Component Responsibility Assignment	Defines which business rules will be encapsulated in which software components.	Ensures all business rules are covered by solution. We want to design components with high internal cohesion and low coupling with other components.	Does each component have a meaningful cluster of responsibilities?	Rule Diagram Component List		Matrix mapping between Rule Diagram and list of Components
	Data Storage Design	Specify the structure and mechanisms for persistent data storage.	Ensures the solution will possess data integrity.				
	Component Specification	Produces a detailed description of the vocabulary and behaviour of all new components.	Enables new components to be found or built.	Initially Assume One Component per Type However, Components Need Each Other. They Will Be Developed and Sold in Suites Executables May Be Delivered as Coarser Grained Components		Complete the Specification of All Components	Type Diagram, with Rule Diagrams to specify pre- and post-conditions.
	Component Sourcing	Plans the acquisition of new components, as Reuse, Buy or Build.	Get the most rapid and cost-effective solution.	What is budget for procurement? What is budget for development? What are timescale and cost implications of each option?			Plan

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
Infrastructure Design	Envision Solution	Create a picture of the new infrastructure.		How to achieve an appropriate level of flexibility.			Collaboration Diagram: Infrastructure Map To Be
	Develop Upgraded Technical Architecture	As the various technology components are identified they need to be fitted into an architecture.		It is here that technology needs to look not only at the information access required but the technology behaviour aspects as well. This is the strength of open distributed computing and its attractiveness to large companies especially at the enterprise level.	Components	Allocate each Component to a Platform	
	Determine re-use opportunities	Examine opportunities to replace or enhance other business processes and their technological support both now and in the future. The cost justification of certain components may require a more business management viewpoint.	The key project objective is to solve the business process issue. This task checks whether we have a plan that achieves a timely implementation of a solution for the current business problem.	If possible, the technology layer should provide benefits across the entire enterprise. Can we identify longer term or broader opportunities? To whom (beyond the current project team) should these potential benefits be presented?			
	Allocate Components to Platforms	Decide how the application components and services will be supported by specific infrastructure components.	To produce a technically efficient and reliable solution.		Component Architecture Technology Architecture Technology Objectives		Deployment Diagram

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
Component Acquisition	Negotiate Ownership	Agree price and other terms for acquisition.	Ensure this project benefits from using this component.	Consider total cost of ownership. Consider support and upgrade arrangements.			Component Acquisition Agreement
	Take Delivery	Obtain application and technology components from external sources, including customer-supplied product.	Make correct version of acquired component available to developers.				Component and its documentation
	Verify Acquired Component	Test or inspect delivered component.	Check that the correct version has been delivered. Confirm that the component is fit-for-purpose.	Can this be done prior to contract?	Component Specification Component and its documentation		Non-conformance log
Acquired Component Implementation	Modify Component Design	Customize component to requirements.	Sometimes it's more cost-effective to alter one component than to alter all its neighbours.	But consider the implications for support and future upgrades.	Specification of Required Component Specification of Acquired Component Component		Modified Component
	Set Component Parameters	Set initial conditions or options for component.	Align interfaces of connecting components. May also want to reduce variety and complexity of test and operations.	What are the operational options built into the component? Does component supplier offer any advice?	Component Documentation		

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
New Component Construction	Build Component	Develop a new component.	Satisfy a requirement that is not covered by existing components or code. Support a new interface or service.	Is it worth building for reuse?	Component Specification	Confirm technology platform Write code	Component
	Unit-Test Component	Test component in isolation.	Check that the component conforms to its specification.	Use of test harness. Range of operating parameters	Component Component Specification		Tested Component
Transition Component Construction	Wrap Legacy Code	Write an interface or bridge to a defined chunk of legacy code.	Reuse legacy code to produce application components that offer the required interface/service.	The legacy code usually (but not necessarily) remains on the original platform.	Legacy Code Component Specification		Component
	Modify Legacy Code	Alter a chunk of legacy code so that it satisfies new requirements at the same time as it continues to play its role in the legacy system.	Reuse and generalize legacy code to produce application components that offer the required interface/service.	The legacy code usually (but not necessarily) remains on the original platform. The legacy system will now access this code through the component interface.	Legacy Code Component Specification		Component

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
Component Publication	Define Terms of Availability	Establish who can have access to a given component (or set of components), under what terms.	Both producer and user should gain something from the wider use of this component. We also need to set limits to the proper or appropriate use of the component.	What is the likely value of this component to various potential users? What are the costs, benefits and risks of making this component available to potential users? How should the costs and risks be shared with potential or actual users of the component?		Identify any risks of over-extended use. Define scope of availability (to whom) Define service guarantees (if any) Define charges Define support arrangements	Component Access Terms
	Create Component Collateral	Produce other materials to be delivered with component.	Enhance proper use of component. Reduce support costs.		Component Specification Component		Component Collateral - may include ReadMe files, SetUp routines and other documentation.
	Publish Component Specification	Make specification of component or service available to potential users, either within the same company or outside.	Allows potential users to identify and locate the component.	May use third party repositories or brokerage services.	Component Specification Component Access Terms		

Major Task	Task	Description	Purpose / Motivation	Key Decisions / Issues	Dependency / Assumption / Resource	Subtasks	Outputs/Outcome / Deliverables
	Establish Component Delivery Mechanism	Defines procedure for authorized users to access the component or service itself.	Facilitate proper use of component, while inhibiting improper use.	If wide use is desired or expected, an automatic delivery and control mechanism may be required.	Component Component Collateral Component Access Terms	Define procedure for identifying and authorizing user Define procedure for confirming that user accepts terms of use Define mechanism for delivering component and collateral to user Define charging mechanism	
Solution Testing	Define Test Plan	Specify how the solution will be tested, including: <ul style="list-style-type: none"> ▪ assembly test ▪ computer system test ▪ performance test ▪ user acceptance test 	Ensure that the tests will be effective in verifying solution, and will use resources efficiently.	Test against specification. Test against intended improvements and user expectations.	Specification	Define Test Strategy Create Test Cases Document Expected Results	Test Plan
	Establish Test Environment	Install all components required for a given series of tests within a controlled test environment	Gain the maximum relevant information from a series of tests. Minimize impact of tests on production environment.		Component Architecture Components		
	Install Test Data Storage	Generate and install persistent data storage for the test environment.	Allows data integrity and persistence characteristics to be tested.	Is it appropriate to replicate live data in the test environment?	Data Storage Design Test Data		

<i>Major Task</i>	<i>Task</i>	<i>Description</i>	<i>Purpose / Motivation</i>	<i>Key Decisions / Issues</i>	<i>Dependency / Assumption / Resource</i>	<i>Subtasks</i>	<i>Outputs/Outcome / Deliverables</i>
	Execute Test Plan	Check that the system produces the expected results under test conditions.	Verifies that the solution satisfies the requirements, executes satisfactorily on the installed technological platform, and provides the expected support for the business process.	Does it execute satisfactorily on the installed technological platform? Does it provide the expected support for the business process?	Test Plan	Execute Test Cases Compare actual results with expected results Analyse and resolve discrepancies Document outstanding risks and issues	Test Log Known characteristics of operational solution.
Solution Deployment	Install Components	Install all components required for this release of the solution.	Prepares for release.	Do all components have to be installed simultaneously? Does a distributed component have to be simultaneously installed in all locations?	Component Architecture Components (available and tested)	Install technology components. Install application components on the appropriate technology components. Install process components.	
	Configure System	Configure technology infrastructure. Set operational parameters.	Prepares for release.				
	Install Live Data Storage	Load or convert data into production databases.	Prepares for release.				
	Commission System	Commence operation. Go live.					Desired Improvements