



# Business Analysis



**tutorialspoint**

SIMPLY EASY LEARNING

[www.tutorialspoint.com](http://www.tutorialspoint.com)



<https://www.facebook.com/tutorialspointindia>



<https://twitter.com/tutorialspoint>

## About the Tutorial

---

Business Analysis is a subject which provides concepts and insights into the development of the initial framework for any project. It holds the key to guide key stakeholders of a project to perform business modelling in a systematic manner. This tutorial provides a brief overview of the concepts of business analysis in an easy to understand manner.

## Audience

---

This tutorial is meant for aspiring business analysts, and project owners or business owners, coordinators and project team members who often work closely with business analysts.

In addition, it will also be useful for anyone who is involved in capturing, writing, analyzing, or understanding requirements for Information Technology solutions, including Subject Matter Experts (SME), Business Process Managers, and Business Process Users.

## Prerequisites

---

To understand this tutorial, it is advisable to have a foundation level knowledge of business scenarios, process and domain knowledge pertaining to a few industries.

## Copyright & Disclaimer

---

© Copyright 2022 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at [contact@tutorialspoint.com](mailto:contact@tutorialspoint.com)

## Table of Contents

---

About the Tutorial.....	i
Audience .....	i
Prerequisites .....	i
Copyright & Disclaimer.....	i
Table of Contents .....	ii
<b>1. BUSINESS ANALYSIS – INTRODUCTION .....</b>	<b>1</b>
<b>2. SOFTWARE DEVELOPMENT LIFE CYCLE .....</b>	<b>4</b>
<b>Post SDLC Process .....</b>	<b>6</b>
<b>Role of Business Analyst during SDLC Process .....</b>	<b>7</b>
<b>3. ROLES OF BUSINESS ANALYSTS.....</b>	<b>8</b>
<b>Major Roles of a BA.....</b>	<b>8</b>
<b>Key Responsibilities of a Business Analyst.....</b>	<b>9</b>
<b>What a BA is Expected to Deliver? .....</b>	<b>10</b>
<b>4. BA TOOLS AND TECHNIQUES.....</b>	<b>12</b>
<b>Functional and Non-Functional Requirements .....</b>	<b>12</b>
<b>5. JAD SESSION.....</b>	<b>14</b>
<b>Use of a JAD Session.....</b>	<b>14</b>
<b>Participants in a JAD Session .....</b>	<b>14</b>
<b>6. REQUIREMENT GATHERING TECHNIQUES.....</b>	<b>16</b>
<b>7. FUNCTIONAL REQUIREMENTS DOCUMENT.....</b>	<b>18</b>
<b>Functional Requirements Deliverables .....</b>	<b>18</b>

8.	SOFTWARE REQUIREMENTS SPECIFICATION .....	20
9.	USE-CASES.....	22
	What is a Use-Case? .....	22
	Benefits of a Use-Case.....	22
	The Anatomy of a Use-Case.....	22
	Guidance for Use-Case Template .....	23
	Use-Case Definition.....	23
10.	USE-CASE DIAGRAMS .....	26
	Drawing Use-Case Diagrams.....	26
	Example – Withdrawal Use-Case .....	28
	Use-Case Template.....	31
11.	REQUIREMENTS MANAGEMENT .....	32
	Why Projects Fail.....	32
	Why Successful Teams do Requirements Management .....	32
	Let’s Start with the Basics .....	33
	Collaboration & Buy- In from Stakeholders .....	34
12.	PLANNING GOOD REQUIREMENTS.....	36
	Requirement Gathering and Analysis .....	36
	Eliciting Approach .....	37
	Different Types of Requirements .....	37
	Traceability & Change Management .....	38
	Idea Requirements Design Test Business Objectives .....	39
	Quality Assurance .....	39
	Obtaining Requirements Signoff .....	39

<b>13. BUSINESS MODELLING .....</b>	<b>41</b>
<b>Purpose of Business Modelling.....</b>	<b>41</b>
<b>Performing GAP Analysis.....</b>	<b>42</b>
<b>To Assess Proposed System.....</b>	<b>43</b>
<b>Guiding Principles for Business Modelling .....</b>	<b>43</b>
<b>Example of BA role in Modelling ERP Systems.....</b>	<b>43</b>
<b>Functional Business Analyst .....</b>	<b>44</b>
<b>Other Major Activities.....</b>	<b>44</b>
<b>Tool 1: Microsoft Visio .....</b>	<b>45</b>
<b>Tool 2: Enterprise Architect.....</b>	<b>47</b>
<b>Tool 3: Rational Requisite Pro .....</b>	<b>47</b>

# 1. Business Analysis – Introduction

## What is Business Analysis?

Business Analysis is the set of tasks, knowledge, and techniques required to identify business needs and determine solutions to enterprise business problems. Although, the general definition is similar, the practices and procedures may vary in various industries.

In Information technology industry, solutions often include a systems development component, but may also consist of process improvement or organizational change.

Business analysis may also be performed to understand the current state of an organization or to serve as a basis for the identification of business needs. In most cases, however, business analysis is performed to define and validate solutions that meets business needs, goals, or objectives.

## Who is a Business Analyst?

A business analyst is someone who analyzes an organization or business domain (real or hypothetical) and documents its business, processes, or systems, assessing the business model or its integration with technology. However, organizational titles vary such as analyst, business analyst, business systems analyst or maybe systems analyst.

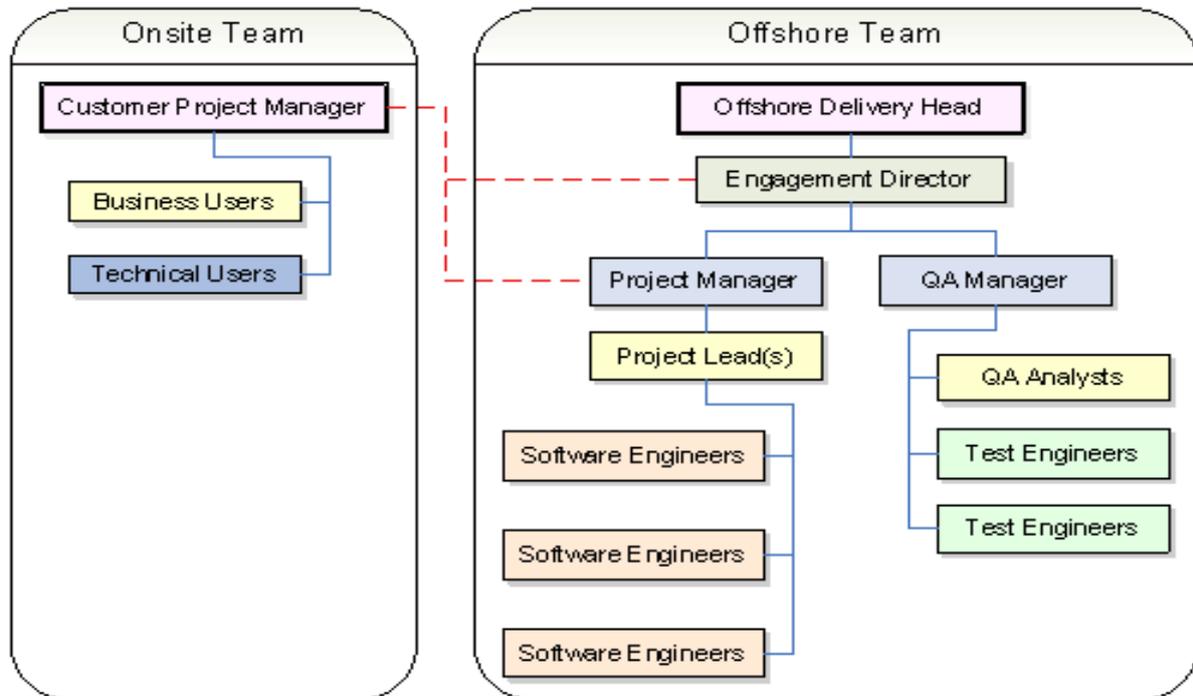
## Why a Business Analyst?

Organizations employ business analysis for the following reasons:

- To understand the structure and the dynamics of the organization in which a system is to be deployed.
- To understand current problems in the target organization and identify improvement potentials.
- To ensure that the customer, end user, and developers have a common understanding of the target organization.

In the initial phase of a project, when the requirements are being interpreted by the solution and design teams, the role of a Business analyst is to review the solutions documents, work closely with the solutions designers (IT team) and Project managers to ensure that requirements are clear.

In a typical large-size IT organization, especially in a development environment, you can find On-site as well as offshore delivery teams having the above-mentioned roles. You can find a "Business Analyst" who acts as a key person who has to link both the teams.

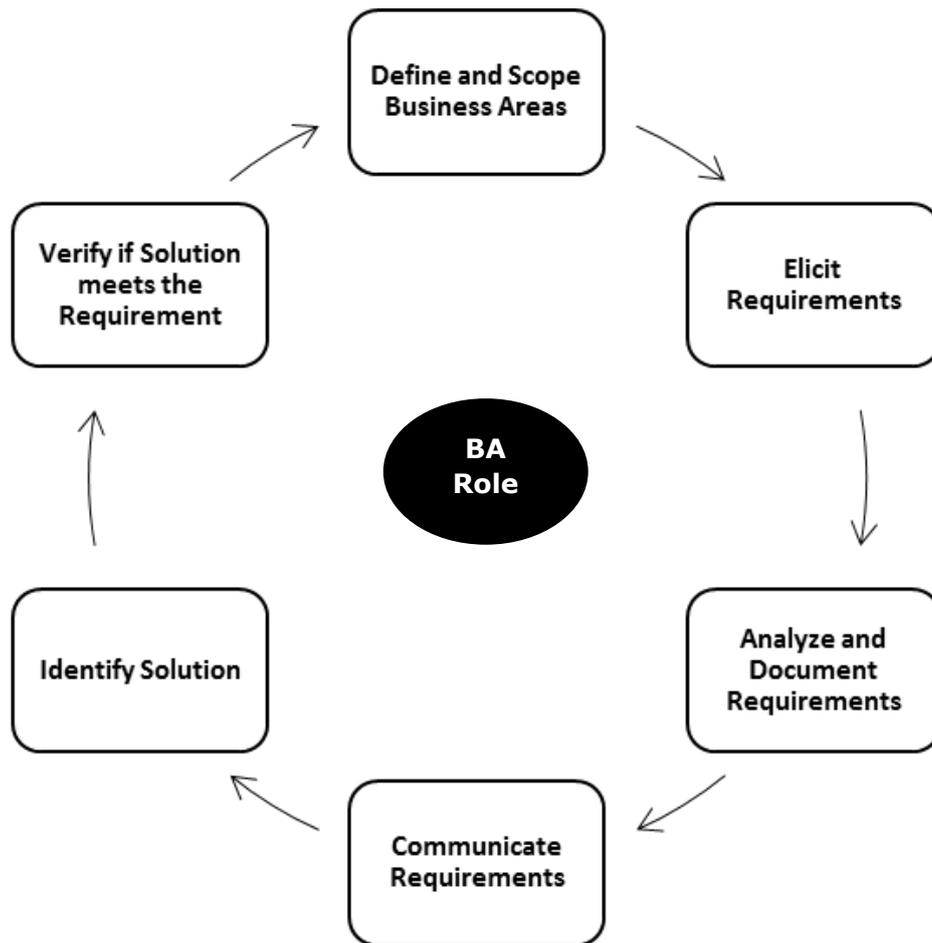


Sometimes, he would interact with Business users and at times technical users and finally to all the stakeholders in the projects to get the approval and final nod before proceeding with the documentation.

Hence, the role of BA is very crucial in the effective and successful jumpstart for any project.

### Role of an IT Business Analyst

The role of a Business analyst starts from defining and scoping the business areas of the organization, then eliciting the requirements, analyzing and documenting the requirements, communicating these requirements to the appropriate stakeholders, identifying the right solution and then validating the solution to find if the requirements meet the expected standards.



### How is it different from other Professions?

Business analysis is distinct from financial analysis, project management, quality assurance, organizational development, testing, training and documentation development. However, depending on the organization, a Business Analyst may perform some or all these related functions.

Business analysts who work solely on developing software systems may be called IT business analysts, technical business analysts, online business analysts, business systems analysts, or systems analysts.

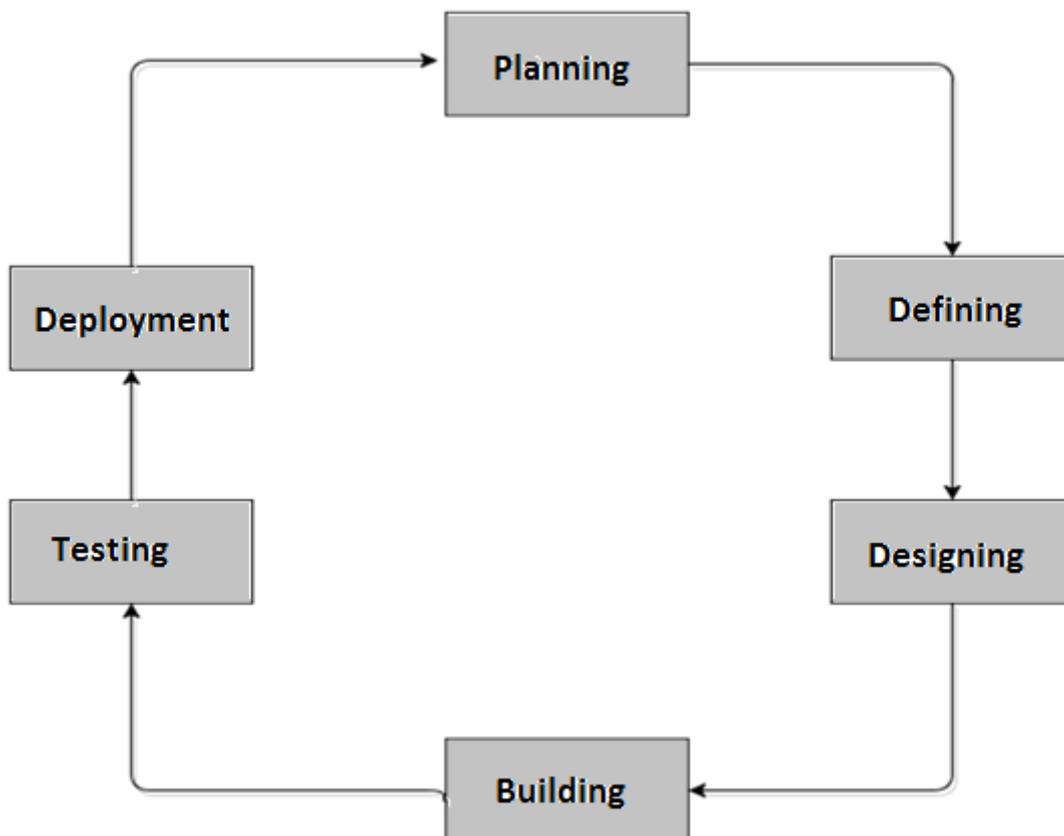
Business analysis also includes the work of liaison among stakeholders, development teams, testing teams, etc.

## 2. Software Development Life Cycle

Software Development Life Cycle (SDLC) is a process followed in a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. It defines a methodology for improving the quality of software and the overall development process.

- SDLC is a process used by IT analysts in order to develop or redesign high quality software system, which meets both the customer and the real-world requirement.
- It takes into consideration all the associated aspects of software testing, analysis and post-process maintenance.

The important phases of SDLC are depicted in the following illustration:



### Planning Stage

Every activity must start with a plan. Failing to plan is planning to fail. The degree of planning differs from one model to another, but it's very important to have a clear understanding of what we are going to build by creating the system's specifications.

## Defining Stage

In this phase, we analyze and define the system's structure. We define the architecture, the components, and how these components fit together to produce a working system.

## Designing Stage

In system design, the design functions and operations are described in detail, including screen layouts, business rules, process diagrams and other documentation. The output of this stage will describe the new system as a collection of modules or subsystems.

## Building Stage

This is the development phase. We start code generation based on the system's design using compilers, interpreters, debuggers to bring the system to life.

## Implementation

Implementation is a part of the Building Stage. In this phase, we start code generation based on the system's design using compilers, interpreters, debuggers to bring the system to life.

## Testing Stage

As different parts of the system are completed; they are put through a series of tests. It is tested against the requirements to make sure that the product is actually solving the needs addressed during the requirement phase.

- Test plans and test cases are used to identify bugs and to ensure that the system is working according to the specifications.
- In this phase, different types of testing like unit testing, manual testing, acceptance testing and system testing is done.

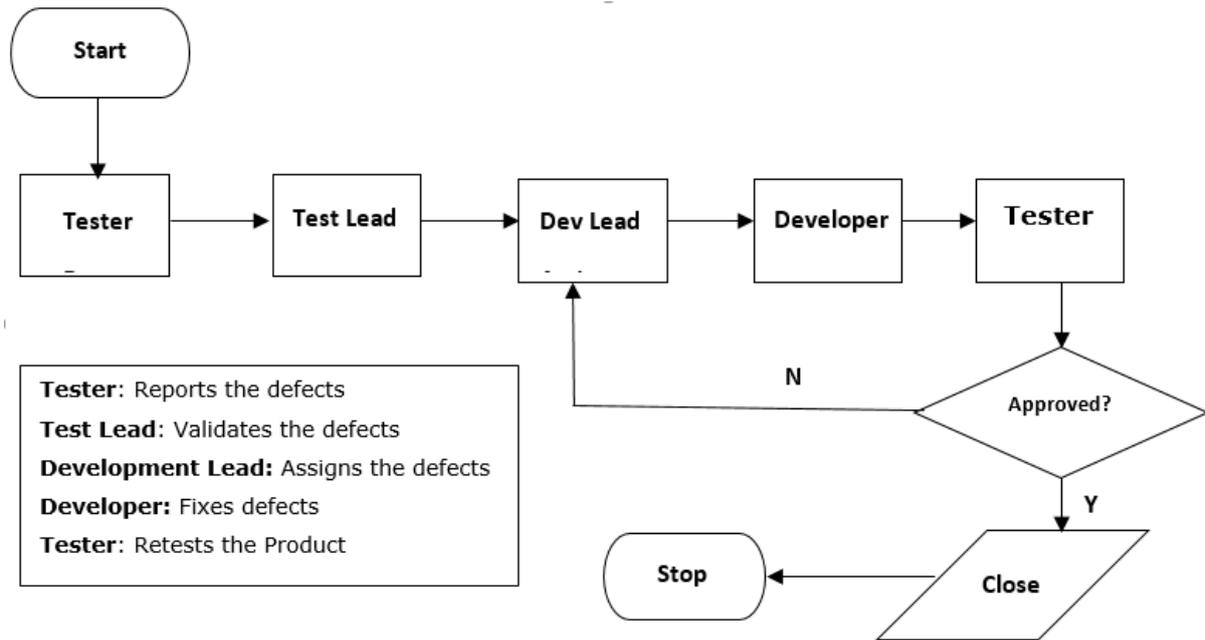
## Defect Tracking in Testing

Software test reports are used to communicate the results of the executed test plans. This being the case, a report should contain all test information that pertains to the current system being tested. The completeness of reports will be verified in walkthrough sessions.

Testing for a project seeks to accomplish two main goals:

- Detect failures and defects in the system.
- Detect inconsistency between requirements and implementation.

The following flowchart depicts the **Defect Tracking Process**:



To achieve the main goals, the testing strategy for the proposed system will usually consist of four testing levels.

These are unit testing, integration testing, acceptance testing, and regression testing. The following subsections outline these testing levels, which development team roles are responsible for developing and executing them, and criteria for determining their completeness.

## Deployment

After the test phase ends, the system is released and enters the production environment. Once the product is tested and ready to be deployed it is released formally in the appropriate market. Sometime product deployment happens in stages as per the organization's business strategy.

The product may first be released in a limited segment and tested in the real business environment (UAT- User acceptance testing). Then based on the feedback, the product may be released as it is or with suggested enhancements in the targeting market segment.

## Post SDLC Process

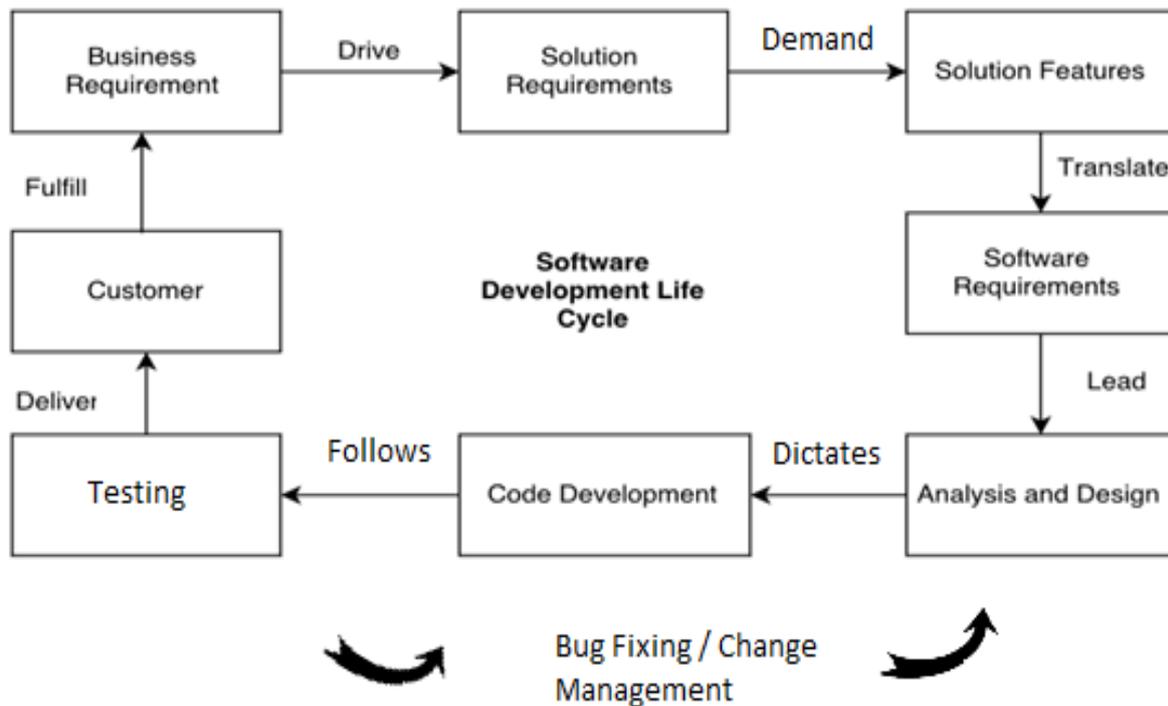
After the product is released in the market, its maintenance is done for the existing customer base.

Once in the production environment, the system will suffer modifications because of undetected bugs or other unexpected events. The system is evaluated and the cycle is repeated for maintaining the system.

## Role of Business Analyst during SDLC Process

As we can see the below diagram, BA is involved in driving business requirement and converting them to solution requirements.

He is involved in translating the solution features into software requirements. Then leads in analysis and designing phase, dictates in code development, then follows the testing phase during bug fixing as a change agent in the project team and ultimately fulfills the customer requirements.



## 3. Roles of Business Analysts

The role of a business analyst in an IT Project can be multi-fold. It is possible for project team members to have multiple roles and responsibilities. In some projects, the BA may take on the roles of the Business Intelligence Analyst, Database Designer, Software Quality Assurance Specialist, Tester, and/or Trainer when there are limited resources available.

It is also possible for a Project Coordinator, or an Application Development Lead, or a Developer to take on the role of the Business Analyst in specific projects.

Business Analysis overlaps heavily with analysis of requirements of the business to function as usual and to optimize how they function. Some examples of Business Analysis are:

- Creating Business Architecture
- Preparing a Business Case
- Conducting Risk assessment
- Requirements Elicitation
- Business Process Analysis
- Documentation of Requirements

### Major Roles of a BA

---

A key role of most business analysts is to liaison between the business and technical developers. Business analysts gets to work together with the business clients to gather/define requirements of a system or process to improve productivity while at the same time working with the technical teams to design and implement the system/process.

#### As a Contributor

The major responsibility of a BA is to contribute to the development of Business user's / key users in identifying business problems, needs and functions, understand stakeholders' concerns and requirements to identify improvement opportunities, and contribute business input for developing the business case for the IT system development project.

#### As a Facilitator

A Business Analyst is also supposed to facilitate/co-ordinate in the elicitation and analysis of requirements, collaborating and communicating with stakeholders and to manage their expectations and needs, and ensure the requirements are complete, unambiguous and map them to real-time business needs of an organization.

#### As an Analyst

Another important role would be to assess proposed system and organizational readiness for system implementation and providing support to users and coordinate with IT staff.

To help review and provide inputs to the design of the proposed IT system from the business perspective, resolving issues/conflicts among stakeholders, help organize comprehensive and quality UAT through assisting users in developing test cases, and help organize training with the aim of ensuring the deployed IT system which is capable of meeting business needs and requirements as well as realizing the anticipated benefits.

Planning and monitoring the Business analysis activities for scope development, schedule and approach for performing the activities related to business analysis for the IT system development project, monitor the progress, coordinating with the Internal Project manager and report on revenue, profitability, risks and issues wherever appropriate.

## **Key Responsibilities of a Business Analyst**

---

The responsibility set of a business analyst would require him to fulfill different duties in different phases of a project and they are elucidated below:

### **Initiation Phase**

This phase will mark the beginning of a new project and a business analyst will vary out the following responsibilities:

- Assist in carrying out the cost-benefit analysis of the project.
- Understand the business case.
- Ascertain the feasibility of the solution/project/product.
- Help in creating the project charter.
- Identify the stakeholders in the project.

### **Planning Phase**

This phase will involve gathering the requirements and planning, how the project will be executed and managed. His responsibilities will include the below functions:

- Eliciting the requirements
- Analyze, organize and document requirements.
- Manage requirements by creating Use-cases, RTM, BRD, SRS, etc.
- Assess proposed solutions.
- Liaise and enhance communications with stakeholders.
- Assist in formulating the project management plans.
- Help in finding the project's scope, constraints, assumptions and risks.
- Assist in designing the user experience of the solution.

### **Executing Phase**

This phase marks the development of the solution as per the requirements gathered. The responsibilities include:

- Explain requirements to the IT/development team.
- Clarify doubts, concerns regarding the proposed solution to be developed.

- Discuss and prioritize project scope changes and gain agreement.
- Create beta tests scripts for initial testing.
- Sharing the developing modules with stakeholders and solicit their feedback.
- Following deadlines and manage stakeholder's expectations.
- Resolving conflicts and manage communications with the project team.

### Monitoring and Controlling Phase

In this phase, the project is measured and controlled for any deviations from the initial plans. This phase runs simultaneously to the execution phase.

- Developing test scripts and conducting comprehensive module and integration testing.
- Conducting UAT (use acceptance testing) and creating testing reports.
- Gain acceptance/approval of the deliverables from the client.
- Explain the change requests to the development team.
- Monitor the development of the change requests and verify their implementation as per the project's objective.

### Closing Phase

This phase marks the closure of the project. The responsibilities are:

- Presenting the completed project to the client and gain their acceptance.
- Create user-training manuals, any functional material and other instructional guides.
- Conduct elaborate integration testing in production environment.
- Create final product documentations, document project lessons learned.

### What a BA is Expected to Deliver?

---

A Business Analyst serves as the bridge between the business users and the technical IT people. Their presence will contribute significantly to the success of IT projects. There are many benefits of having a dedicated business analyst. A dedicated business analyst can:

- Delivers a clear project scope from a business point of view.
- Develop sound business cases and more realistic estimation of resources and business benefits.
- Prepares better reports on project scoping, planning and management in terms of costs and schedule, especially for large-scale IT projects.
- Produces clear and concise requirements, which in turn, helps provide clearer and more accurate requirements, if the IT project is outsourced

- Elicit the real business needs from users and effectively manage user expectations.
- Improves the quality of design for the proposed IT system so that it meets the user requirements.
- Ensures the quality of the system developed before passing on to end-users for review and acceptance.
- Arranges comprehensive quality test on the delivered systems and provide feedback to the technical IT people.

## 4. BA Tools and Techniques

A Business Analyst should be familiar with various analytical tools and related technologies when you are wearing the BA hat. I mean, if you are holding this position.

As we have already learnt earlier, business analysis is a process where you are trying to understand a business enterprise and identifying opportunities, problem areas, problems and meeting a wide range of people having wide range of roles and responsibilities like CEO, VP, Director and understanding their business requirements.

Fundamentally, there are 3 types of Business analysis which we can categorize into -

- **Strategic Analysis:** Strategic business analysis deals with pre-project work. It is the method or process of identifying business problems, devising business strategies, goals and objectives helping the top management. It provides management information reporting for effective decision making process.
- **Tactical Analysis:** It involves knowledge of specific business analysis techniques to apply at the right time in the appropriate project.
- **Operational Analysis:** In this type of Business analysis, we are focussed towards the business aspect by leveraging information technology. It is also a process of studying operational systems with the aim of identifying opportunities for business improvement.

For each type of analysis, there are a set of tools which are available in the market and based on organizational needs and requirements, these are to be used.

However, to materialize business requirements into understandable information, a good BA will leverage techniques Fact-Finding, Interviews, Documentation Review, Questionnaires, Sampling and Research in their day-to-day activities.

### **Functional and Non-Functional Requirements**

---

We can breakdown a requirement into two principal types like Functional and Non-functional requirements.

For all the technology projects, functional and non-functional requirements must be segregated and separately analyzed.

To define the proper tool and an appropriate technique might be a daunting challenge. Whether you are doing a brand-new application or making change to an existing application. Considering the right technique for the functional process is an art by itself.

An overview of the widely-used business analysis techniques which are currently in the market:

Processes	Techniques	Process Deliverables (Outcomes)
To Determine Functional and Non-Functional Requirements	<ul style="list-style-type: none"> <li>• JAD Sessions</li> <li>• Scenarios and Use-cases</li> <li>• Organizational Modeling</li> <li>• Scope Modeling</li> <li>• Functional Decomposition</li> <li>• Interviews</li> <li>• Observation (Job Shadowing)</li> <li>• Focus Groups</li> <li>• Acceptance and Evaluation</li> <li>• Sequence Diagrams</li> <li>• User Stories</li> <li>• Brainstorming</li> <li>• Storyboarding</li> <li>• Prototyping</li> <li>• Structured Walk-through</li> <li>• Event Analysis</li> <li>• Business Rule analysis</li> <li>• Requirements Workshops</li> <li>• Risk Analysis</li> <li>• Root Cause Analysis</li> </ul>	<p><b>Business Requirements Documents:</b></p> <ul style="list-style-type: none"> <li>• Business and Functional Requirements</li> <li>• Non-Functional Requirements</li> <li>• Business Rules</li> <li>• Requirements Traceability Matrix</li> </ul> <p><b>Common Template:</b></p> <ul style="list-style-type: none"> <li>• Business Requirements Document</li> </ul>

### Applicability of Tools and Process

Although there are a variety of tools and procedures available to business analysts, it all depends upon the current practices of the organization and how they would like to use it.

For example, **root-cause analysis** is used when there is a requirement to go deeper into a certain important area or function.

However, business requirements document is the most popular and accepted way to put the requirements in documentation format.

In the subsequent chapters, we will be discussing some of the above techniques in-depth.

# 5. JAD Session

Joint Application Development (JAD) is a process used to collect business requirements while developing new information systems for a company. The JAD process may also include approaches for enhancing user participation, expediting development and improving the quality of specifications. The intention of a JAD session is to pool in subject matter expert's/Business analyst or IT specialist to bring out solutions.

A Business analyst is the one who interacts with the entire group and gathers the information, analyses it and brings out a document. He plays a very important role in JAD session.

## Use of a JAD Session

---

JAD sessions are highly structured, facilitated workshops that bring together customer decision makers and IT staff to produce high quality deliverables in a short period.

In other words, a JAD Session enables customers and developers to quickly come to an agreement on the basic scope, objectives and specifications of a project or in case, not come to an agreement which means the project needs to be re-evaluated.

Simply put, JAD sessions can

- **Simplify:** It consolidates months of meetings and phone calls into a structured workshop.
- **Identify:** Issues and participants
- **Quantify:** Information and processing needs
- **Clarify:** Crystallize and clarify all requirements agreed upon in the session.
- **Unify:** The output from one phase of development is input to the next.
- **Satisfy:** The customers define the system; therefore, it is their system. Shared participation brings a share in the outcome; they become committed to the systems success.

## Participants in a JAD Session

---

The participants involved in a JAD session are as follows:

### Executive Sponsor

An executive sponsor is the person who drives the project – the system owner. They normally are from higher positions and are able to make decisions and provide necessary strategy, planning and direction.

## Subject Matter Expert

These are the business users and outside experts who are required for a successful workshop. The subject matter experts are the backbone of the JAD session. They will drive the changes.

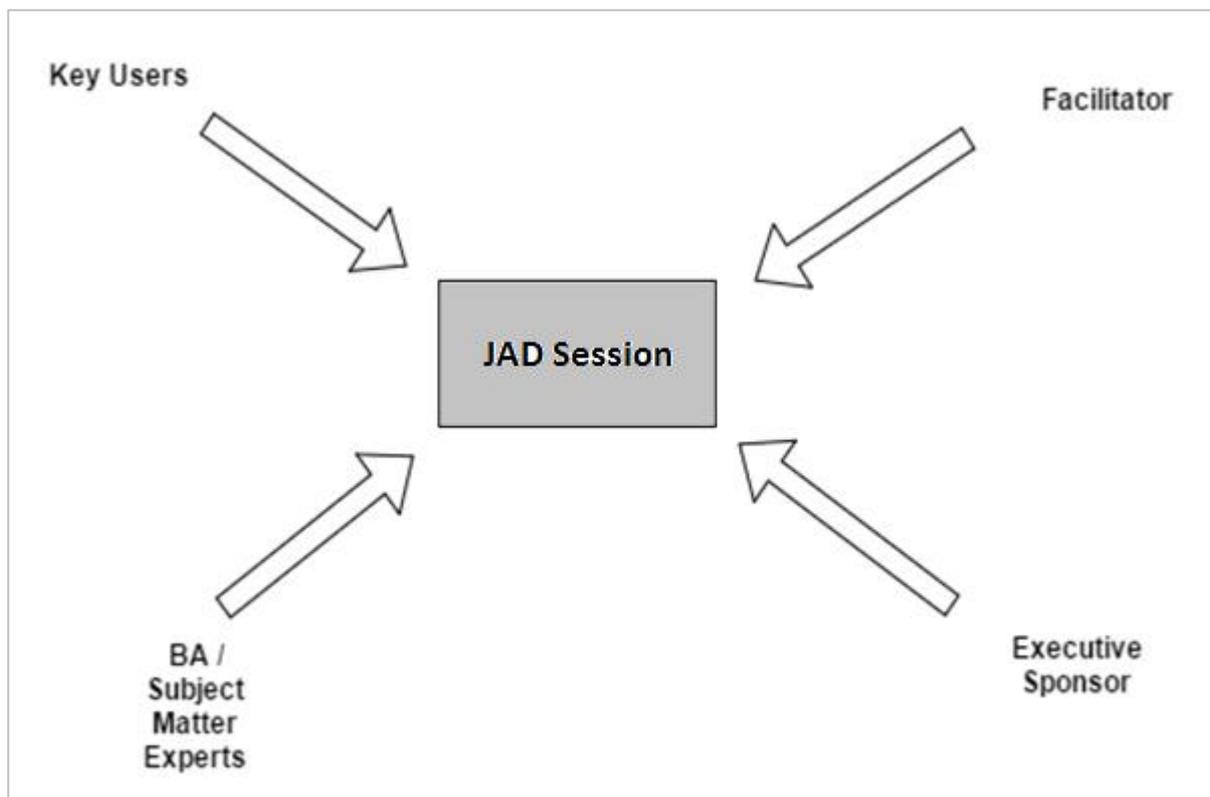
## Facilitator

He chairs the meeting; he identifies issues that can be solved as part of the meeting. The facilitator does not contribute information to the meeting.

## Key Users

Key users or also called as super users in some instances have been used interchangeably and differs from company to company still. Key users are generally the business users who are more tightly aligned to the IT project and are responsible for the configuration of profiles of their team members during the projects.

For Example: Suppose John is a key user and Nancy, Evan are users of a SAP system. In this instance, Nancy and Evan does not have access to change the functionality and profile whereas John being a Key user has access to edit profile with more authorizations.



The JAD approach, in comparison with the more traditional practice, is thought to lead to faster development times and greater client satisfaction, because the client is involved throughout the development process. In comparison, in the traditional approach to systems development, the developer investigates the system requirements and develops an application, with client input consisting of a series of interviews.

## 6. Requirement Gathering Techniques

Techniques describe how tasks are performed under specific circumstances. A task may have none or one or more related techniques. A technique should be related to at least one task.

The following are some of the well-known requirements gathering techniques:

### **Brainstorming**

Brainstorming is used in requirement gathering to get as many ideas as possible from group of people. Generally used to identify possible solutions to problems, and clarify details of opportunities.

### **Document Analysis**

Reviewing the documentation of an existing system can help when creating AS-IS process document, as well as driving gap analysis for scoping of migration projects. In an ideal world, we would even be reviewing the requirements that drove creation of the existing system – a starting point for documenting current requirements. Nuggets of information are often buried in existing documents that help us ask questions as part of validating requirement completeness.

### **Focus Group**

A focus group is a gathering of people who are representative of the users or customers of a product to get feedback. The feedback can be gathered about needs/opportunities/problems to identify requirements, or can be gathered to validate and refine already elicited requirements. This form of market research is distinct from brainstorming in that it is a managed process with specific participants.

### **Interface analysis**

Interfaces for a software product can be human or machine. Integration with external systems and devices is just another interface. User centric design approaches are very effective at making sure that we create usable software. Interface analysis – reviewing the touch points with other external systems is important to make sure we don't overlook requirements that aren't immediately visible to users.

### **Interview**

Interviews of stakeholders and users are critical to creating the great software. Without understanding the goals and expectations of the users and stakeholders, we are very unlikely to satisfy them. We also have to recognize the perspective of each interviewee, so that, we can properly weigh and address their inputs. Listening is the skill that helps a great analyst to get more value from an interview than an average analyst.

## Observation

By observing users, an analyst can identify a process flow, steps, pain points and opportunities for improvement. Observations can be passive or active (asking questions while observing). Passive observation is better for getting feedback on a prototype (to refine requirements), where active observation is more effective at getting an understanding of an existing business process. Either approach can be used.

## Prototyping

Prototyping is a relatively modern technique for gathering requirements. In this approach, you gather preliminary requirements that you use to build an initial version of the solution - a prototype. You show this to the client, who then gives you additional requirements. You change the application and cycle around with the client again. This repetitive process continues until the product meets the critical mass of business needs or for an agreed number of iterations.

## Requirement Workshops

Workshops can be very effective for gathering requirements. More structured than a brainstorming session, involved parties collaborate to document requirements. One way to capture the collaboration is with creation of domain-model artifacts (like static diagrams, activity diagrams). A workshop will be more effective with two analysts than with one.

## Reverse Engineering

When a migration project does not have access to sufficient documentation of the existing system, reverse engineering will identify what the system does. It will not identify what the system should do, and will not identify when the system does the wrong thing.

## Survey/Questionnaire

When collecting information from many people – too many to interview with budget and time constraints – a survey or questionnaire can be used. The survey can force users to select from choices, rate something (“Agree Strongly, agree...”), or have open ended questions allowing free-form responses. Survey design is hard – questions can bias the respondents.

# 7. Functional Requirements Document

The Functional Requirements Document (FRD) is a formal statement of an application's functional requirements. It serves the same purpose as a contract. Here, the developers agree to provide the capabilities specified. The client agrees to find the product satisfactory if it provides the capabilities specified in the FRD.

Functional requirements capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform. The document should be tailored to fit a particular project's need. They define things such as system calculations, data manipulation and processing, user interface and interaction with the application.

The Functional Requirements Document (FRD) has the following characteristics:

- It demonstrates that the application provides value in terms of the business objectives and business processes in the next few years.
- It contains a complete set of requirements for the application. It leaves no room for anyone to assume anything which is not stated in the FRD.
- It is solution independent. The ERD is a statement of what the application is to do—not of how it works. The FRD does not commit the developers to a design. For that reason, any reference to the use of a specific technology is entirely inappropriate in an FRD.

The functional requirement should include the following:

- Descriptions of **data** to be entered into the system
- Descriptions of **operations** performed by each screen
- Descriptions of **work-flows** performed by the system
- Descriptions of **system reports** or other outputs
- Who can enter the **data** into the system?
- How the system meets applicable **regulatory requirements?**

The functional specification is designed to be read by a general audience. Readers should understand the system, but no technical knowledge should be required to understand this document.

## Functional Requirements Deliverables

---

A Business Requirements Document (BRD) consists of:

- **Functional Requirements:** A document containing detailed requirements for the system being developed. These requirements define the functional features and capabilities that a system must possess. Be sure that any assumptions and constraints identified during the Business Case are still accurate and up to date.

- **Business Process Model:** A model of the current state of the process ("as is" model) or a concept of what the process should become ("to be" model)
- **System Context Diagram:** A Context Diagram shows the system boundaries, external and internal entities that interact with the system, and the relevant data flows between these external and internal entities.
- **Flow Diagrams** (as-is or to-be): Diagrams graphically depict the sequence of operations or the movement of data for a business process. One or more flow diagrams are included depending on the complexity of the model.
- **Business Rules and Data Requirements:** Business rules define or constrain some aspects of the business and are used to define data constraints, default values, value ranges, cardinality, data types, calculations, exceptions, required elements and the relational integrity of the data.
- **Data Models:** Entity Relationship Diagrams, Entity Descriptions, Class Diagrams
- **Conceptual Model** - High level display of different entities for a business function and how they relate to one another.
- **Logical Model:** Illustrates the specific entities, attributes and relationships involved in a business function and represents all the definitions, characteristics, and relationships of data in a business, technical, or conceptual environment.
- **Data Dictionary and Glossary:** A collection of detailed information on the data elements, fields, tables and other entities that comprise the data model underlying a database or similar data management system.
- **Stakeholder Map:** Identifies all stakeholders who are affected by the proposed change and their influence/authority level for requirements. This document is developed in the origination phase of the Project Management Methodology (PMM) and is owned by the Project Manager but needs to be updated by the project team as new/changed Stakeholders are identified throughout the process.
- **Requirements Traceability Matrix:** A table that illustrates logical links between individual functional requirements and other types of system artifacts, including other Functional Requirements, Use-cases/User Stories, Architecture and Design Elements, Code Modules, Test Cases, and Business Rules.

## 8. Software Requirements Specification

A Software Requirements Specification (SRS) is a document, which is used as a communication medium between the customers. A software requirement specification in its most basic form is a formal document used in communicating the software requirements between the customer and the developer.

An SRS document concentrates on **WHAT** needs to be done and carefully avoids the solution (**how to do**). It serves as a contract between development team and the customer. The requirements at this stage is written using end user terminology. If necessary, later a formal requirement specification will be developed from it.

SRS is a complete description of the behavior of a system to be developed and may include a set of use-cases that describes the interactions, the users will have with the software.

### Purpose of SRS

SRS is a communication tool between Customer / Client, Business Analyst, System developers, Maintenance teams. It can also be a contract between purchaser and supplier.

- It will give firm foundation for the design phase
- Supports project management and control
- Helps in controlling and evolution of system

A software Requirement specification should be Complete, Consistent, Traceable, Unambiguous, and Verifiable.

The following should be addressed in system specification:

- Define the functions of the systems
- Define the Hardware / Software Functional Partitioning
- Define the Performance Specification
- Define the Hardware / Software Performance Partitioning
- Define Safety Requirements
- Define the User Interface (user's manual)
- Provide Installation Drawings/Instructions
- Software Requirement specification template

## Revision History

Date	Description	Author	Comments
<date>	<Version 1>	<Your Name>	<First Revision>

## Document Approval

The following software requirements specification has been accepted and approved by the following:

Signature	Printed Name	Title	Date
	<Your Name>	Lead Software Eng.	
	David	Instructor	

# 9. Use-Cases

One of the nine diagrams of UML's are the Use-case Diagram. These are not only important but necessary requirement for software projects. It is basically used in Software life cycles. As we know there are various phases in the development cycle and the most used phase for Use-cases would be during the requirements gathering phase.

## What is a Use-Case?

---

A use-case describes a sequence of actions, performed by a system that provides value to an actor. The use-case describes the system's behavior under various conditions as it responds to a request from one of the stakeholders, called the **primary actor**.

The actor is the Who of the system, in other words he the end user.

In software and systems engineering, a use-case is a list of steps, typically defining interactions between a role (known in UML as an "actor") and a system, to achieve a goal. The actor can be a human or an external system.

A use-case specifies the flow of events in the system. It is more concerned with what is performed by the system in order to perform the sequence of actions.

## Benefits of a Use-Case

---

A use-case provides the following benefits:

- It is an easy means of capturing the functional requirement with a focus on value added to the user.
- Use-cases are relatively easy to write and read compared to the traditional requirement methods
- Use-cases force developers to think from the end user perspective
- Use-case engage the user in the requirement process

## The Anatomy of a Use-Case

---

Name	:	Descriptive name that illustrates the purpose of the use-case.
Description	:	Describes what the use-case does in couple of sentences.
Actor	:	List any actors that participate in the use-case
Pre-condition	:	Conditions that must be met prior to starting the use-case
Flow of events	:	Description of interaction between the system and the actor contains main flows and alternate flows.
Post Condition	:	Describe the state of the system after a use-case has run its course.

## Guidance for Use-Case Template

---

Document each use-case using the template given in the end of this chapter. This section provides a description of each section in the use-case template.

### Use-Case Identification

- **Use-Case ID:** Give each use-case a unique numeric identifier, in hierarchical form: X.Y. Related use-cases can be grouped in the hierarchy. Functional requirements can be traced back to a labelled use-case.
- **Use-Case Name:** State a concise, results-oriented name for the use-case. These reflect the tasks the user needs to be able to accomplish using the system. Include an action verb and a noun. Some examples:
  - View part number information.
  - Manually mark hypertext source and establish link to target.
  - Place an order for a CD with the updated software version.

### Use-Case History

Here, we mention about the names of the people who are the stakeholders of the Use-case document.

- **Created By:** Supply the name of the person who initially documented this use-case.
- **Date Created:** Enter the date on which the use-case was initially documented.
- **Last Updated By:** Supply the name of the person who performed the most recent update to the use-case description.
- **Date Last Updated:** Enter the date on which the use-case was most recently updated.

## Use-Case Definition

---

The following are the definitions of the key concepts of Use-Case:

### Actor

An actor is a person or other entity external to the software system being specified who interacts with the system and performs use-cases to accomplish tasks. Different actors often correspond to different user classes, or roles, identified from the customer community that will use the product. Name the actor(s) that will be performing this use-case.

### Description

Provide a brief description of the reason for and outcome of this use-case, or a high-level description of the sequence of actions and the outcome of executing the use-case.

## Preconditions

List any activities that must take place, or any conditions that must be true, before the use-case can be started. Number each precondition. Examples:

- User's identity has been authenticated.
- User's computer has sufficient free memory available to launch task.

## Post Conditions

Describe the state of the system at the conclusion of the use-case execution. Number each post condition.

### Examples

- Document contains only valid SGML tags.
- Price of item in database has been updated with new value.

## Priority

Indicate the relative priority of implementing the functionality required to allow this use-case to be executed. The priority scheme used must be the same as that used in the software requirements specification.

## Frequency of Use

Estimate the number of times this use-case will be performed by the actors per some appropriate unit of time.

## Normal Course of Events

Provide a detailed description of the user actions and system responses that will take place during execution of the use-case under normal, expected conditions. This dialog sequence will ultimately lead to accomplishing the goal stated in the use-case name and description. This description may be written as an answer to the hypothetical question, "How do I <accomplish the task stated in the use-case name>?" This is best done as a numbered list of actions performed by the actor, alternating with responses provided by the system.

## Alternative Courses

Document other, legitimate usage scenarios that can take place within this use-case separately in this section. State the alternative course, and describe any differences in the sequence of steps that take place. Number each alternative course using the Use-case ID as a prefix, followed by "AC" to indicate "Alternative Course". Example: X.Y.AC.1.

## Exceptions

Describe any anticipated error conditions that could occur during execution of the use-case, and define how the system is to respond to those conditions. Also, describe how the system is to respond if the use-case execution fails for some unanticipated reason. Number each exception using the Use-case ID as a prefix, followed by "EX" to indicate "Exception". Example: X.Y.EX.1.

## **Includes**

List any other use-cases that are included ("called") by this use-case. Common functionality that appears in multiple use-cases can be split out into a separate use-case that is included by the ones that need that common functionality.

## **Special Requirements**

Identify any additional requirements, such as nonfunctional requirements, for the use-case that may need to be addressed during design or implementation. These may include performance requirements or other quality attributes.

## **Assumptions**

List any assumptions that were made in the analysis that led to accepting this use-case into the product description and writing the use-case description.

## **Notes and Issues**

List any additional comments about this use-case or any remaining open issues or TBDs (To Be Determined) that must be resolved. Identify who will resolve each issue, the due date, and what the resolution ultimately is.

## **Change Management and Version control**

Version control is the management of changes to documents, large websites, and other collection of information. Changes are usually identified by a number or letter code, termed as revision number or revision level. Each revision is associated with a timestamp and person making the change.

# 10. Use-Case Diagrams

An important part of the Unified Modeling Language (UML) is the facilities for drawing use-case diagrams. Use-cases are used during the analysis phase of a project to identify and partition system functionality. They separate the system into actors and use-cases. Actors represent roles that can be played by users of the system.

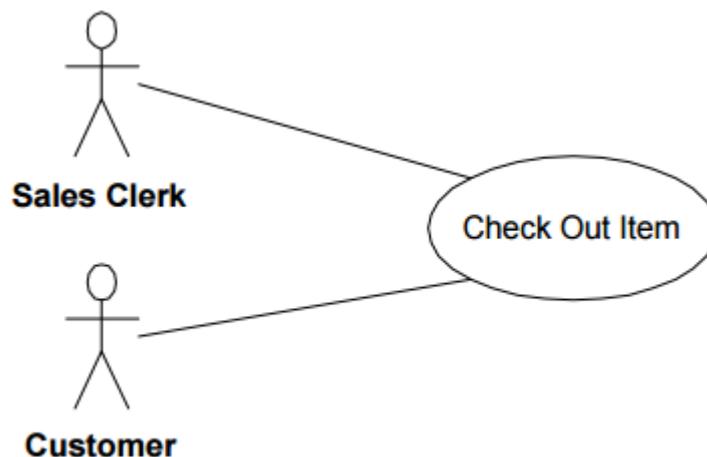
Those users can be humans, other computers, pieces of hardware, or even other software systems. The only criterion is that they must be external to the part of the system being partitioned into use-cases. They must supply stimuli to that part of the system, and they must receive outputs from it.

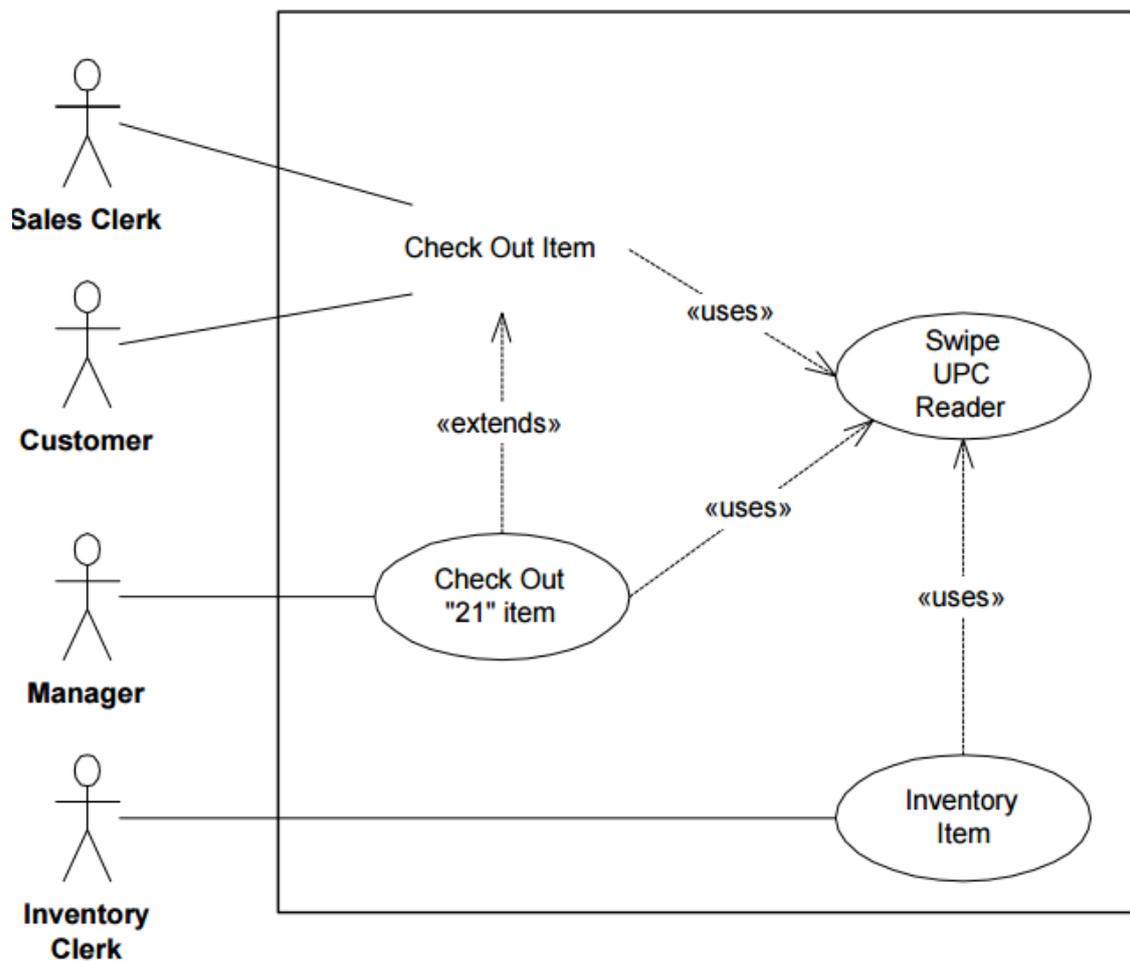
Use-cases represent the activities that actors perform with the help of your system in the pursuit of a goal. We need to define what those users (actors) need from the system. Use-cases should reflect user needs and goals, and should be initiated by an actor. Business actors, Customers participating in the business use-case should be connected to the use-case by association.

## Drawing Use-Case Diagrams

---

The Figure below shows what a use-case might look like in UML schematic form. The use-case itself looks like an oval. The actors are drawn as little stick figures. The actors are connected to the use-case with lines.





**Use-case 1:** Sales Clerk checks out an item

- Customer sets item on counter.
- «uses» Swipe UPC Reader.
- System looks up UPC code in database procuring item description and price
- System emits audible beep.
- System announces item description and price over voice output.
- System adds price and item type to current invoice.
- System adds price to correct tax subtotal

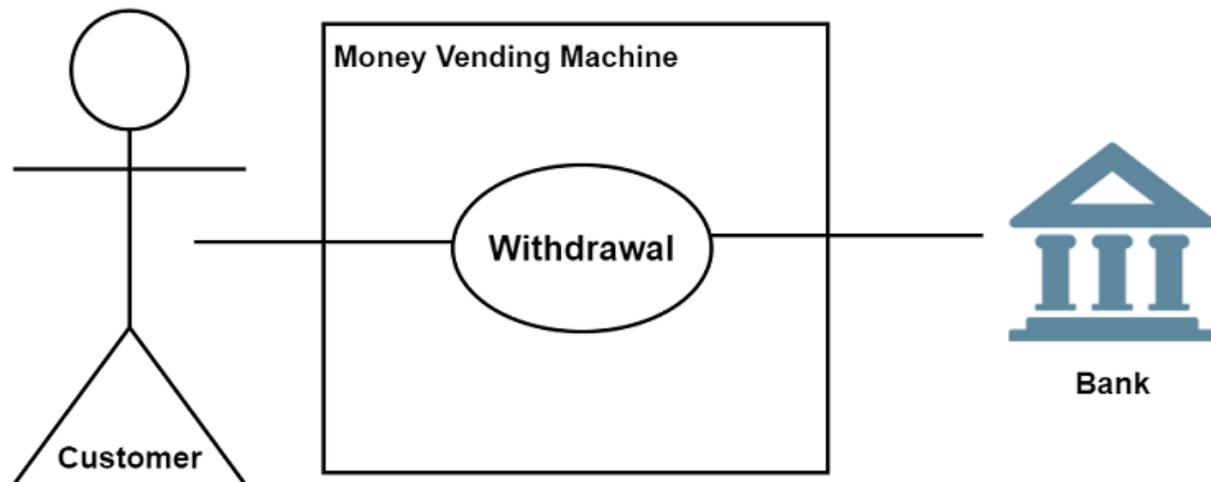
So, the «uses» relationship is very much like a function call or a subroutine.

The use-case being used in this fashion is called an abstract use-case because it cannot exist on its own but must be used by other uses cases.

## Example – Withdrawal Use-Case

The goal of a customer in relation to our money vending machine (ATM) is to withdraw money. So, we are adding **Withdrawal** use-case. Withdrawing money from the vending machine might involve a bank for the transactions to be made. So, we are also adding another actor – **Bank**. Both actors participating in the use-case should be connected to the use-case by association.

Money vending machine provides Withdrawal use-case for the customer and Bank actors.



### *Scenario of a Typical Banking Transaction*

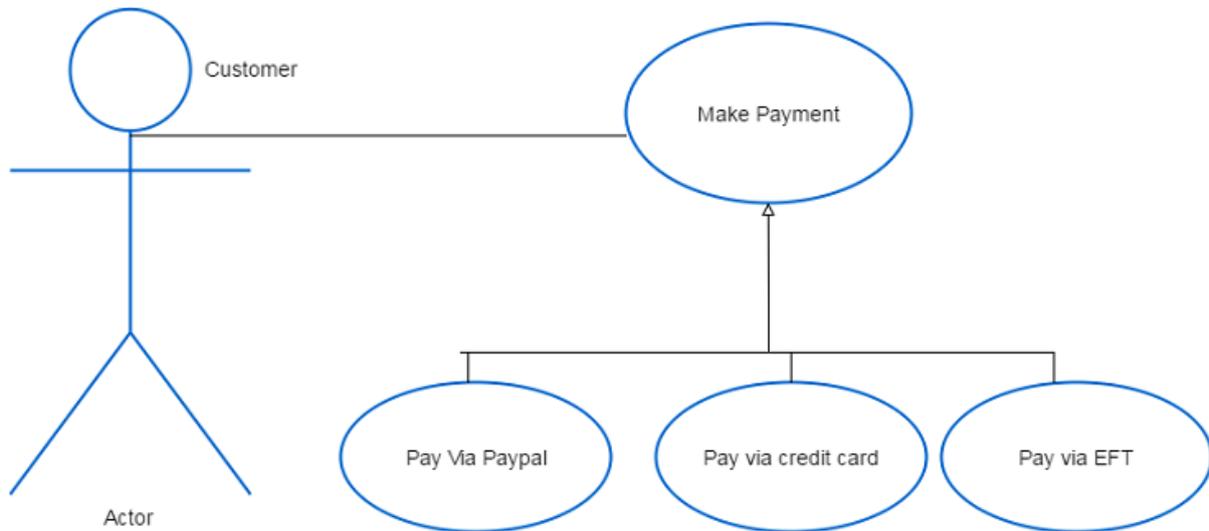
## Relationships between Actors and Use-Cases

Use-cases could be organized using following relationships:

- Generalization
- Association
- Extend
- Include

## Generalization between Use-Cases

There may be instances where actors are associated with similar use-cases. In such case a Child use-case inherits the properties and behavior of the parent use. Hence we need to generalize the actor to show the inheritance of functions. They are represented by a solid line with a large hollow triangle arrowhead.



## Association between Use-Cases

Associations between actors and use-cases are indicated in use-case diagrams by solid lines. An association exists whenever an actor is involved with an interaction described by a use-case.

## Extend

There are some functions that are triggered optionally. In such cases the extend relationship is used and the extension rule is attached to it. Thing to remember is that the base use-case should be able to perform a function on its own even if the extending use-case is not called.

Extend relationship is shown as a dashed line with an open arrowhead directed from the extending use-case to the extended (base) use-case. The arrow is labeled with the keyword «extend».

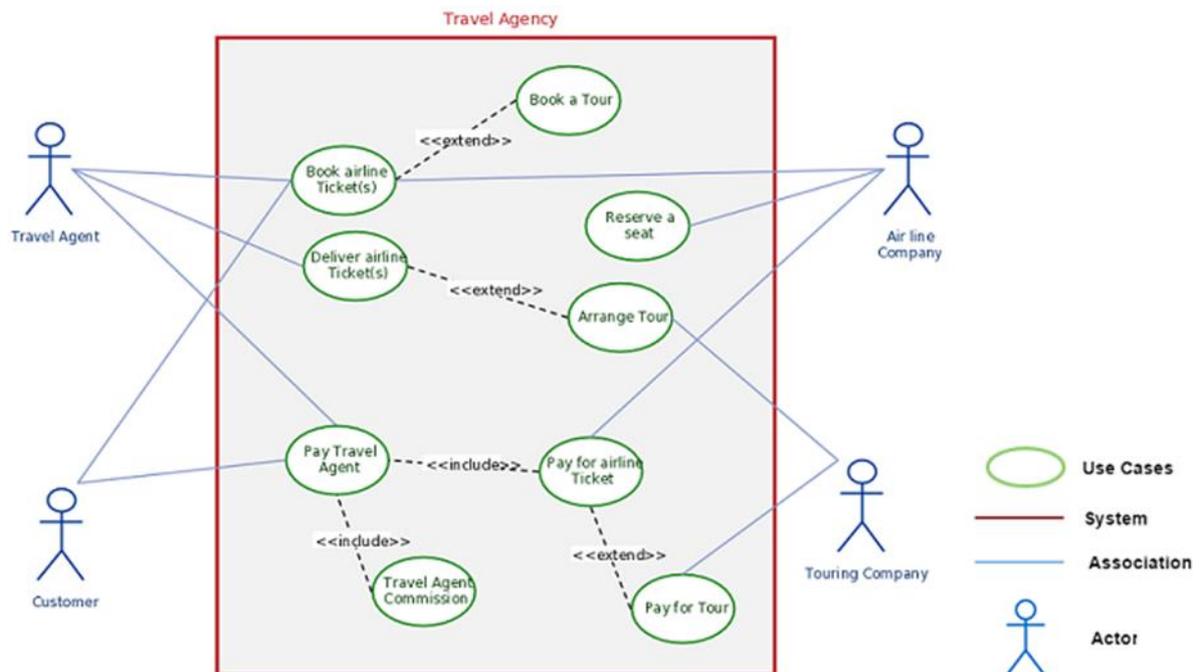
## Include

It is used to extract use-case fragments that are duplicated in multiple use-cases. It is also used to simplify large use-case by splitting it into several use-cases and to extract common parts of the behaviors of two or more use-cases.

Include relationship between use-cases which is shown by a dashed arrow with an open arrowhead from the base use-case to the included use-case. The arrow is labeled with the keyword «include».

Use-cases deal only in the functional requirements for a system. Other requirements such as business rules, quality of service requirements, and implementation constraints must be represented separately.

The diagram shown below is an example of a simple use-case diagram with all the elements marked.



### Basic Principles for Successful Application of Use-cases

- Keep it simple by telling stories
- Be productive without perfection
- Understand the big picture
- Identify reuse opportunity for use-cases
- Focus on value
- Build the system in slices
- Deliver the system in increments
- Adapt to meet the team's needs

## Use-Case Template

---

Here, we have shown a sample template of a Use-Case which a Business Analyst can fill so that the information can be useful for the technical team to ascertain information about the project.

Use-case ID:			
Use-case Name:			
Created By:		Last Updated By:	
Date Created:		Date Last Updated:	
Actor:			
Description:			
Preconditions:			
Post conditions:			
Priority:			
Frequency of Use:			
Normal Course of Events:			
Alternative Courses:			
Exceptions:			
Includes:			
Special Requirements:			
Assumptions:			
Notes and Issues:			

# 11. Requirements Management

Gathering software requirements is the foundation of the entire software development project. Soliciting and gathering business requirements is a critical first step for every project. In-order to bridge the gap between business and technical requirements, the business analysts must fully understand the business needs within the given context, align these needs with the business objectives, and properly communicate the needs to both the stakeholders and development team.

The key stakeholders wish that someone could explain customer / client requirements in plain English. Will this benefit them from understanding the value at a high-level? This will be the main-focus area, as they will try to map the documentation with the requirements and how BA could communicate in the best possible way.

## Why Projects Fail

---

There are many reasons why projects fail but some of the common areas include the below

- Market and Strategy Failure
- Organizational and Planning Failures
- Quality Failures
- Leadership and Governance failures
- Skills, Knowledge and competency failures
- Engagement, team work and communication failures

### Major Issues

Too often projects fail due to poorly managed requirements

Complex and changing communication hurdles

At the core of the issue is that projects are increasingly complex, changes occur and communication is challenging.

## Why Successful Teams do Requirements Management

---

Requirements management is about keeping your team **in-sync** and providing **visibility** to what is going on within a project.

It is critical to the success of your projects for your whole team to understand what you are building and why – that's how we define requirements management. The "why" is important because it provides context to the goals, feedback and decisions being made about the requirements.

This increases predictability of future success and potential problems, allowing your team to quickly course correct any issues and successfully complete your project on time and

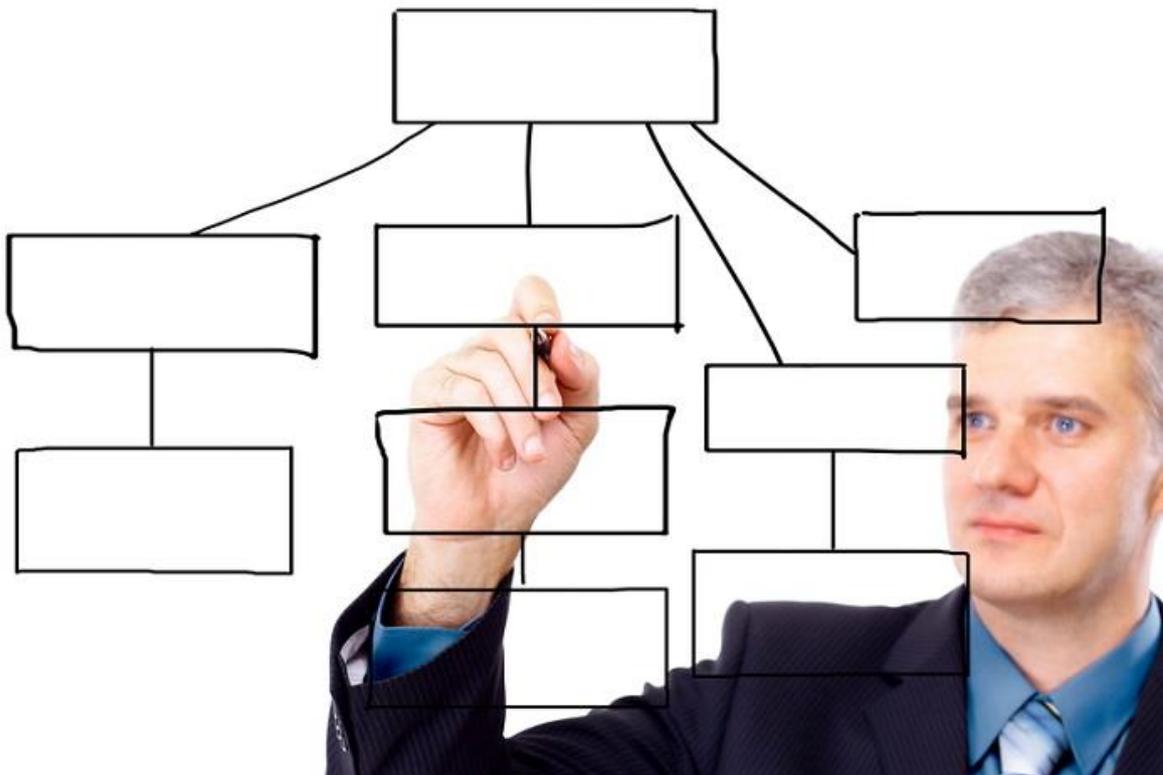
within budget. As a starting point, it's valuable for everyone involved to have a basic understanding of what requirements are, and how to manage them.

## Let's Start with the Basics

---

A requirement is a condition or capability needed by a stakeholder to solve a problem or achieve an objective. A condition or capability that must be met or possessed by a system or system. Component to satisfy a contract, standard, specification, or other formally imposed documents.

A requirement can be expressed with text, sketches, detailed mockups or models, whatever information best communicates to an engineer what to build, and to a QA manager what to test. Depending on your development process, you might use different terminology to capture requirements.



High-level requirements are sometimes referred to simply as **needs** or **goals**. Within software development practices, requirements might be referred to as "use-cases", "features" or "functional requirements". Even more specifically within agile development methodologies, requirements are often captured as **epics** and **stories**.

Regardless of what your team calls them or what process you use; requirements are essential to the development of all products. Without clearly defining requirements you could produce an incomplete or defective product. Throughout the process there can be many people involved in defining requirements.

A stakeholder might request a feature that describes how the product will provide value in solving a problem. A designer might define a requirement based on how the final product should look or perform from a usability or user interface standpoint.

A business analyst might create a system requirement that adheres to specific technical or organizational constraints. For today's sophisticated products and software applications being built, it often takes hundreds or thousands of requirements to sufficiently define the scope of a project or a release. Thus, it's imperative that the team be able to access, collaborate, update, and test each requirement through to completion, as requirements naturally change and evolve over time during the development process.

Now that we've defined the value of requirements management at a high-level, let's go deeper into the four fundamentals that every team member and stakeholder can benefit from understanding:

- Planning good requirements: "What the heck are we building?"
- Collaboration and buy-in: "Just approve the spec, already!"
- Traceability & change management: "Wait, do the developers know that changed?"
- Quality assurance: "Hello, did anyone test this thing?"

Does everyone know what we're building and why? That's the value of requirements management.

## **Collaboration & Buy-In from Stakeholders**

---

Is everyone in the loop? Do we have approval on the requirements to move forward? These questions come up during development cycles. It would be great if everyone could agree on requirements, but for large projects with many stakeholders, this does not usually happen. Trying to get everyone in agreement can cause decisions to be delayed, or worse, not made at all. Gaining consensus on every decision is not always easy.

In practice, you don't necessarily want "consensus," you want "buy-in" from the group and approval from those in control so you can move the project forward. With consensus, you are trying to get everyone to compromise and agree on the decision. With buy-in, you are trying to get people to back the best solution, make a smart decision and do what is necessary to move forward.



You don't need everyone to agree that the decision is the best. You need everyone to support the decision. Team collaboration can help in receiving support on decisions and in planning good requirements.

Collaborative teams work hard to make sure everyone has a stake in projects and provides feedback. Collaborative teams continuously share ideas, typically have better communication and tend to support decisions made because there is a shared sense of commitment and understanding of the goals of the project.

It's when developers, testers, or other stakeholders feel "out of the loop" that communication issues arise, people get frustrated and projects get delayed. Once everyone has bought-in to the scope of work, it is imperative for requirements to be clear and well documented. Keeping track of all the requirements is where things get tricky.

Imagine having a to-do list a mile long that involves collaborating with multiple people to complete. How would you keep all those items straight? How would you track how one change to an item would affect the rest of the project? This is where traceability and change management add value.

# 12. Planning Good Requirements

So, what makes a good requirement? A good requirement should be valuable and actionable; it should define a need as well as provide a pathway to a solution. Everyone on the team should understand what it means. Requirements vary in complexity.

- A good Requirements Document can be part of a group with high-level requirements broken down into sub-requirements.
- They may also include very detailed specifications that include a set of functional requirements describing the behavior or components of the end-product.
- Good requirements need to be concise and specific, and should answer the question, "what do we need?" Rather than, "how do we fulfil a need?"
- Good requirements ensure that all stakeholders understand their part of the plan; if parts are unclear or misinterpreted the final product could be defective or fail.

Preventing failure or misinterpretation of requirements can be aided by receiving feedback from the team continuously throughout the process as requirements evolve. Continuous collaboration and buy-in with everyone is a key to success.

## Requirement Gathering and Analysis

---

A requirement is a condition or capability needed by a stakeholder to solve a problem or achieve an organizational objective; a condition or capability that must be met or possessed by a system.

*Requirement analysis* in software engineering covers those tasks that go into determining the needs or conditions to meet for a new or altered product taking account of the possible conflicting requirements of various stakeholders, analyzing, documenting, validating and managing software or system requirements.

The requirements should be:

- Documented
- Actionable
- Measurable
- Testable
- Traceable

Requirements should be related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

A Business analyst gathers information through observing the existing systems, studying the existing procedures, discussions with the customers and the end users. The analyst should also have imaginative and creative skills in absence of a working System. Analyzing the gathered requirement to find the missing links is requirement analysis.

## Eliciting Approach

---

To elicit the objectives, ask the business expert, the development manager, and the project sponsor the following questions:

- What business objectives of the company will this project help achieve?
- Why are we doing this project now?
- What will happen if we do it later?
- What if we do not do it at all?
- Who will benefit from this project?
- Do the people who will benefit from it consider it the most important improvement that can possibly be made at this time?
- Should we be doing a different project instead?

Possible objectives might be reducing costs, improving the customer service, simplifying the work flow, replacing obsolete technology, piloting a new technology, and many others. Also, make sure you understand exactly how the proposed project will help accomplish the stated objective.

## Different Types of Requirements

---

The most common types of requirement which a Business analyst is interested would be the following:

### Business Requirements

Business requirements are the critical activities of an enterprise that must be performed to meet the organizational objectives while remaining solution independent. A business requirements document (BRD) details the business solution for a project including the documentation of customer needs and expectations.

### User Requirements

User requirements should specify the specific requirements which the user expects/wants from software to be constructed from the software project. A user requirement should be Verifiable, Clear and concise, Complete, Consistent, Traceable, Viable.

The user requirements document (URD) or user requirements specification is a document usually used in software engineering that specifies what the user expects the software to be able to do.

### System Requirements

System requirements deal with defining software resources requirements and pre-requisites that needs to be installed on a computer to provide optimal functioning of an application.

## Functional Requirements

Functional requirements capture and specify specific intended behavior of the system being developed. They define things such as system calculations, data manipulation and processing, user interface and interaction with the application, and other specific functionality that show how user requirements are satisfied. Assign a unique ID number to each requirement.

## Non-Functional Requirements

Non-functional requirement is the one which specifies criteria that can be used to judge the operation of a system rather than specific behaviors. System architecture speaks on the plan for implementing non-functional requirements.

Non-functional requirements speak on how the system should look like or it can be mentioned like "system shall be". Non-functional requirements are called as qualities of the system.

## Transition Requirements

Transition Requirements describe capabilities that the solution must fulfill in order to facilitate transition from the current state of the enterprise to a desired future state, but that will not be needed once that transition is complete.

They are differentiated from other requirements types, because they are always temporary in nature and because they cannot be developed until both an existing and new solution is defined. They typically cover data conversion from existing systems, skill gaps that must be addressed, and other related changes to reach the desired future state. They are developed and defined through solution assessment and validation.

## Traceability & Change Management

---

Requirements traceability is a way to organize, document and keep track of all your requirements from initial idea generation through to the testing phase.

The requirements trace ability matrix (RTM) provides a method for tracking the functional requirements and their implementation through the development process. Each requirement is included in the matrix along with its associated section number.

As the project progresses, the RIM is updated to reflect each requirement's status. When the product is ready for system testing, the matrix lists each requirement, what product component addresses it, and what test verifies that it is correctly implemented.

Include columns for each of the following in the RTM:

- Requirement description
- Requirement reference in FRD
- Verification Method
- Requirement reference in Test Plan

**Example:** Connecting the dots to identify the relationships between items within your project. It is a connector of common downstream flow.

## Idea Requirements Design Test Business Objectives

---

You should be able to trace each of your requirements back to its original business objective.

By tracing requirements, you are able to identify the ripple effect changes have, see if a requirement has been completed and whether it's being tested properly. Traceability and change management provides managers peace of mind and the visibility needed to anticipate issues and ensure continuous quality.

## Quality Assurance

---

Getting requirements delivered right the first time can mean better quality, faster development cycles and higher customer satisfaction with the product. Requirements management not only helps you get it right, but also helps your team save money and many headaches throughout the development process.

Concise, specific requirements can help you detect and fix problems early, rather than later when it is much more expensive to fix. In addition, it can cost up to **100 times** more to correct a defect later in the development process after it's been coded, than it is to correct early on while a requirement.

By integrating requirements management into your quality assurance process, you can help your team increase efficiency and eliminate rework. Moreover, rework is where most of the cost issues occur.

In other words, development teams are wasting majority of their budgets on efforts that are not performed correctly the first time. For example, a developer codes a feature based on an old specification document, only to learn later, that the requirements for that feature changed. These types of issues can be avoided with effective requirements management best practices.

In summary, requirements management can sound like a complex discipline, but when you boil it down to a simple concept – it's about helping teams answer the question, "Does everyone understand what we're building and why?" From the business analysts, product managers and project leaders to the developers, QA managers and testers, along with the stakeholders and customers involved – so often the root cause of project failure is a misunderstanding of the scope of the project.

When everyone is collaborating, and has full context and visibility to the discussions, decisions and changes involved with the requirements throughout the lifecycle of the project, that is when success happens consistently and you maintain continuous quality. In addition, the process is smoother with less friction and frustration along the way for everyone involved.

**Note:** Research has shown that project teams can eliminate 50-80% of project defects by effectively managing requirements. According to the Carnegie Mellon software engineering institute, "60-80 percent of the cost of software development is in rework."

## Obtaining Requirements Signoff

---

Requirements signoff formalizes agreement by project stakeholders that the content and presentation of the requirements, as documented, are accurate and complete. Formal

agreement reduces the risk that, during or subsequent to implementation, a stakeholder will introduce a new (previously unencountered) requirement.

Obtaining requirements signoff typically involves a face-to-face final review of requirements, as documented, with each project stakeholder. At the end each review, the stakeholder is asked to formally approve the reviewed requirements document. This approval may be recorded either physically or electronically.

Obtaining requirements signoff is typically the final task within Requirements Communication. The Business Analyst will require the output from the Formal Requirements Review(s), including accommodation of any comments or objections which were raised during the review process.

# 13. Business Modelling

A Business Model can be defined as a representation of a business or solution that often include a graphic component along with supporting text and relationships to other components. For example, if we have to understand a company's business model, then we would like to study the following areas like

- Core values of the company
- What it serves?
- What is sets apart?
- Its key resources
- Major relationships
- Its delivery channels

With the help of modelling techniques, we can create a complete description of existing and proposed organizational structures, processes, and information used by the enterprise.

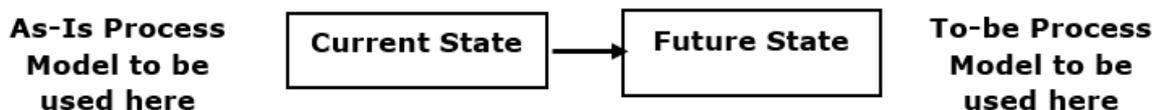
Business Model is a structured model, just like a blueprint for the final product to be developed. It gives structure and dynamics for planning. It also provides the foundation for the final product.

## Purpose of Business Modelling

---

Business modelling is used to design current and future state of an enterprise. This model is used by the Business Analyst and the stakeholders to ensure that they have an accurate understanding of the current "As-Is" model of the enterprise.

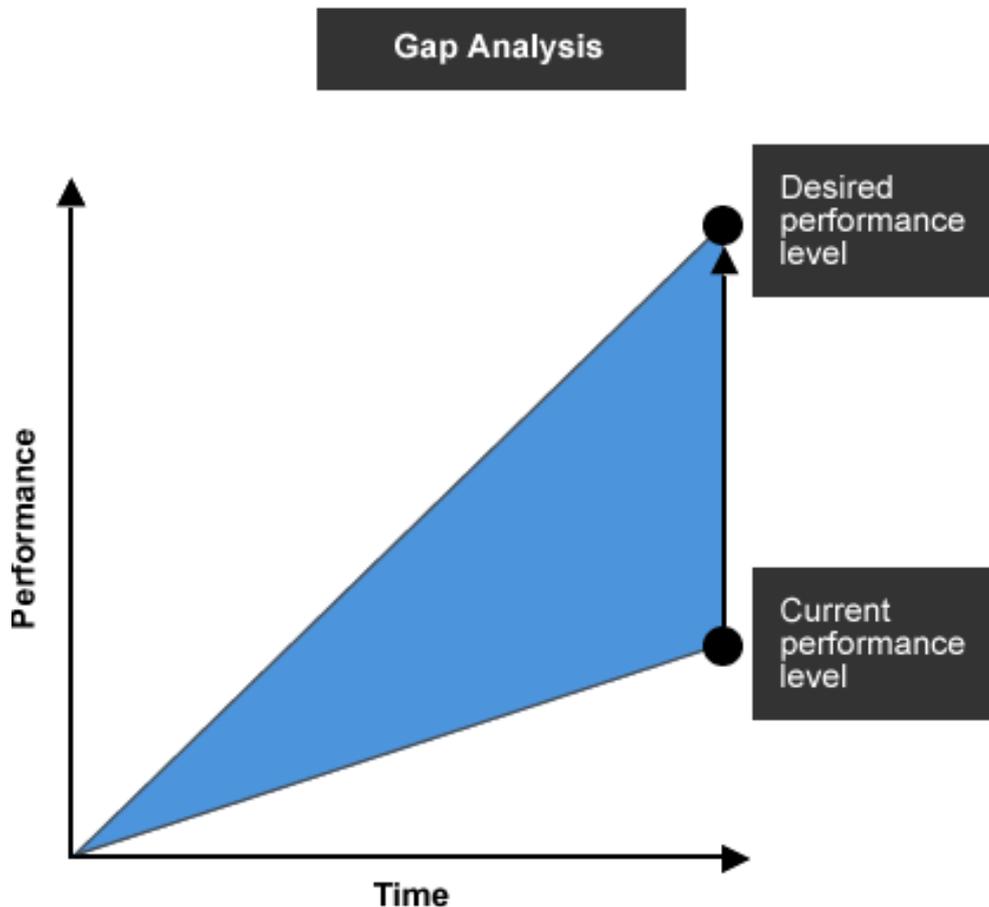
It is used to verify if, stakeholders have a shared understanding of the proposed "To-be of the solution.



Analyzing requirements is a part of business modelling process and it forms the core focus area. Functional Requirements are gathered during the "Current state". These requirements are provided by the stakeholders regarding the business processes, data, and business rules that describe the desired functionality which will be designed in the Future State.

## Performing GAP Analysis

After defining the business needs, the current state (e.g. current business processes, business functions, features of a current system and services/products offered and events that the system must respond to) must be identified to understand how people, processes and technology, structure and architecture are supporting the business by seeking input from IT staff and other related stakeholders including business owners.



A gap analysis is then performed to assess, if there is any gap that prevents from achieving business needs by comparing the identified current state with the desired outcomes.

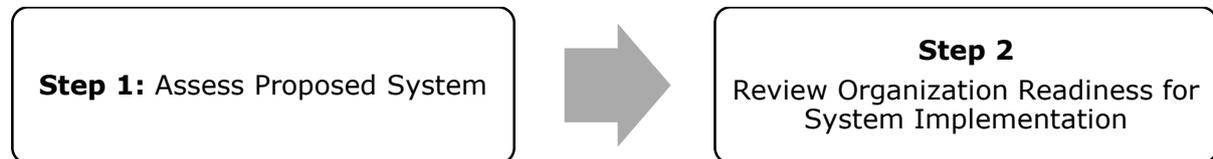
If there is no gap (i.e. the current state is adequate to meet the business needs and desired outcomes), it will probably not be necessary to launch the IT project. Otherwise, the problems/issues required to be addressed in order to bridge the gap should be identified.

Techniques such as SWOT (Strengths, Weaknesses, Opportunities and Threats) Analysis and document analysis can be used.

## To Assess Proposed System

---

BA should assist the IT project team in assessing the proposed IT system to ensure that it meets the business needs and maximizes the values delivered to stakeholders. BA should also review the organization readiness for supporting the transition to the proposed IT system to ensure a smooth System Implementation.



BA should help the IT project team to determine whether the proposed system option and the high-level system design could meet the business needs and deliver enough business value to justify the investment. If there are more than one system options, BA should work with the IT staff to help to identify the pros and cons of each option and select the option that delivers the greatest business value.

## Guiding Principles for Business Modelling

---

The primary role of business modelling is mostly during inception stage and elaboration stages of project and it fades during the construction and transitioning stage. It is mostly to do with analytical aspects of business combined with technical mapping of the application or software solution.

- **Domain and User variation:** Developing a business model will frequently reveal areas of disagreement or confusion between stakeholders. The Business Analyst will need to document the following variations in the as-is model.
- **Multiple work units perform the same function:** Document the variances in the AS-IS model. This may be different divisions or geographies.
- **Multiples users perform the same work:** Different stakeholders may do similar work differently. The variation may be the result of different skill sets and approaches of different business units or the result of differing needs of external stakeholders serviced by the enterprise. Document the variances in the AS-IS model.
- **Resolution Mechanism:** The Business Analyst should document whether the To-Be solution will accommodate the inconsistencies in the current business model or whether the solution will require standardization. Stakeholders need to determine which approach to follow. The To-Be model will reflect their decision.

## Example of BA role in Modelling ERP Systems

---

A Business analyst is supposed to define a standard business process and set up into an ERP system which is of key importance for efficient implementation. It is also the duty of a BA to define the language of the developers in understandable language before the implementation and then, utilize best practices and map them based on the system capabilities.

A requirement to the system is the GAAP fit analysis, which has to balance between:

- The need for the technical changes, which are the enhancements in order to achieve identity with the existing practice.
- Effective changes, which are related to re-engineering of existing business processes to allow for implementation of the standard functionality and application of process models.

## Functional Business Analyst

---

Domain expertise is generally acquired over a period by being in the “business” of doing things. For example,

- A **banking associate** gains knowledge of various types of accounts that a customer (individual and business) can operate along with detailed business process flow.
- An **insurance sales representative** can understand the various stages involved in procuring of an Insurance policy.
- A **marketing analyst** has more chances of understanding the key stakeholders and business processes involved in a Customer Relationship Management system.
- A Business Analyst involved in **capital markets** project is supposed to have subject matter expertise and strong knowledge of Equities, Fixed Income and Derivatives. Also, he is expected to have handled back office, front office, practical exposure in applying risk management models.
- A **Healthcare Business Analyst** is required to have basic understanding of US Healthcare Financial and Utilization metrics, Technical experience and understanding of EDI 837/835/834, HIPAA guidelines, ICD codification – 9/10 and CPT codes, LOINC, SNOMED knowledge.

Some business analysts acquire domain knowledge by testing business applications and working with the business users. They create a conducive learning environment through their interpersonal and analytical skills. In some cases, they supplement their domain knowledge with a few domain certifications offered by AICPCU/IIA and LOMA in the field of Insurance and financial services. There are other institutes that offer certification in other domains.

## Other Major Activities

---

Following a thorough examination of current business processes, you can offer highly professional assistance in identifying the optimal approach of modelling the system.

- Organizing the preparation of a formalized and uniform description of business processes in a manner ensuring efficient automation in the system.
- Assistance to your teams in filling out standard questionnaires for the relevant system as may be furnished by the developers.
- Participation in working meetings requirements towards the developers are defined.

- Check and control as to whether the requirements set by you have been properly “reproduced” and recorded in the documents describing the future model in the system (Blueprints).
- Preparation of data and assisting for prototyping the system.
- Assistance in preparation of data for migration of lists and balances in the format required by the system.
- Review of the set-up prototype for compliance with the requirements defined by the business process owners.
- Acting as a support resource to your IT teams in preparing data and actual performance of functional and integration tests in the system.

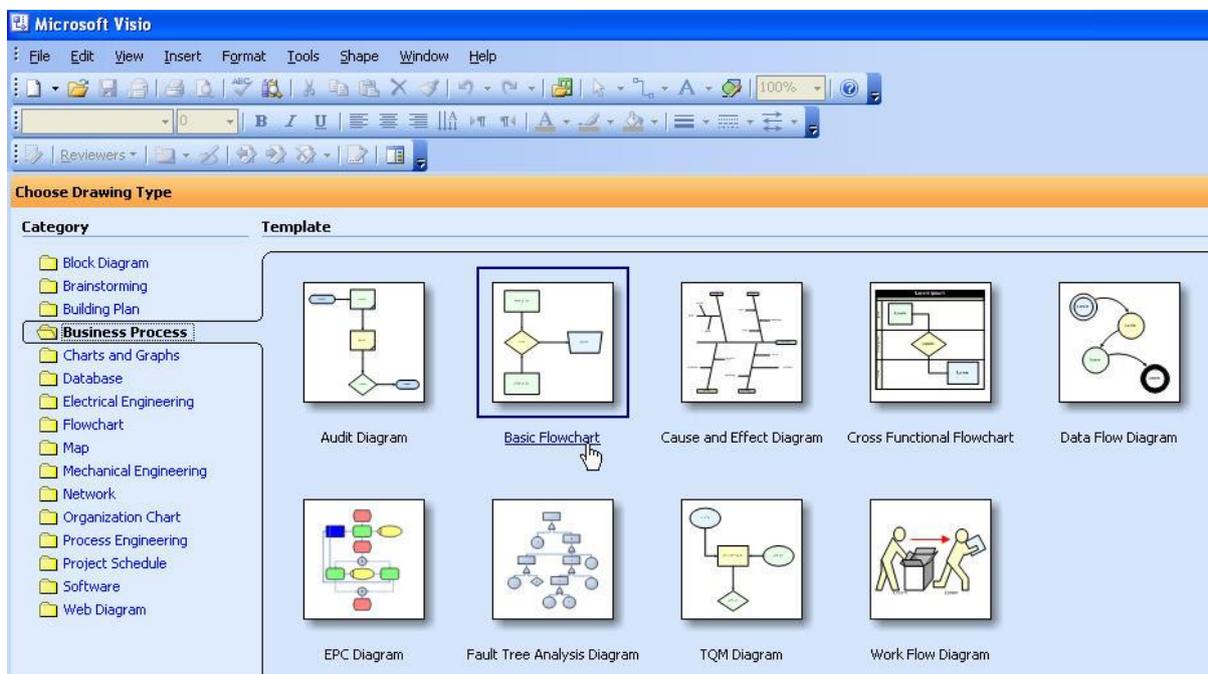
In the next section, we will discuss briefly about some of the popular Business Modelling Tools used by large organizations in IT environments.

## Tool 1: Microsoft Visio

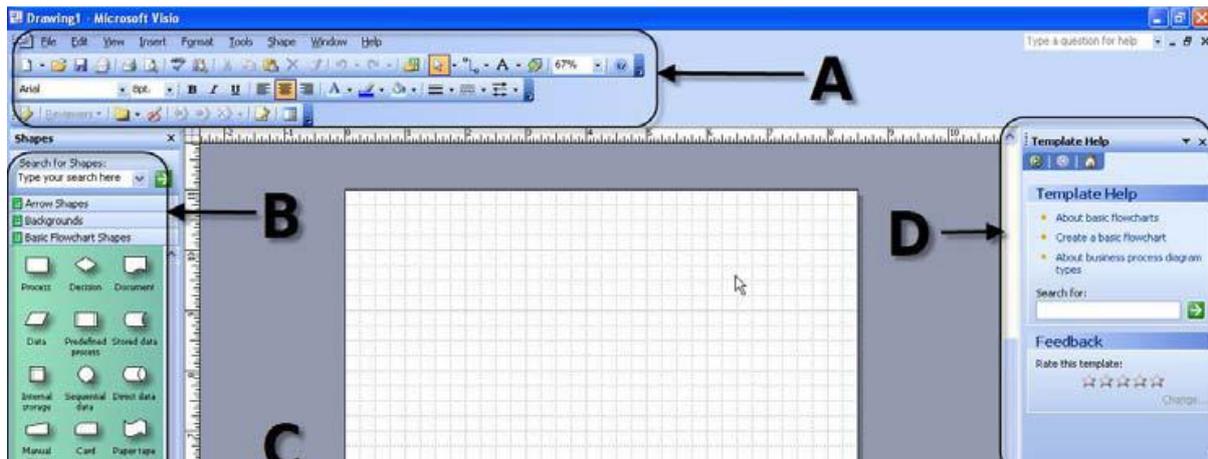
MS-Visio is a drawing and diagramming software that helps transform concepts into a visual representation. Visio provides you with pre-defined shapes, symbols, backgrounds, and borders. Just drag and drop elements into your diagram to create a professional communication tool.

**Step 1:** To open a new Visio drawing, go to the Start Menu and select Programs → Visio.

**Step 2:** Move your cursor over “Business Process” and select “Basic Flowchart”



The following screenshot shows the major sections of MS-Visio application.



Let us now discuss the basic utility of each component:

**A:** the toolbars across the top of the screen are like other Microsoft programs such as Word and PowerPoint. If you have used these programs before, you may notice a few different functionalities, which we will explore later.

Selecting Help Diagram Gallery is a good way to become familiar with the types of drawings and diagrams that can be created in Visio.

**B:** The left side of the screen shows the menus specific to the type of diagram you are creating. In this case, we see:

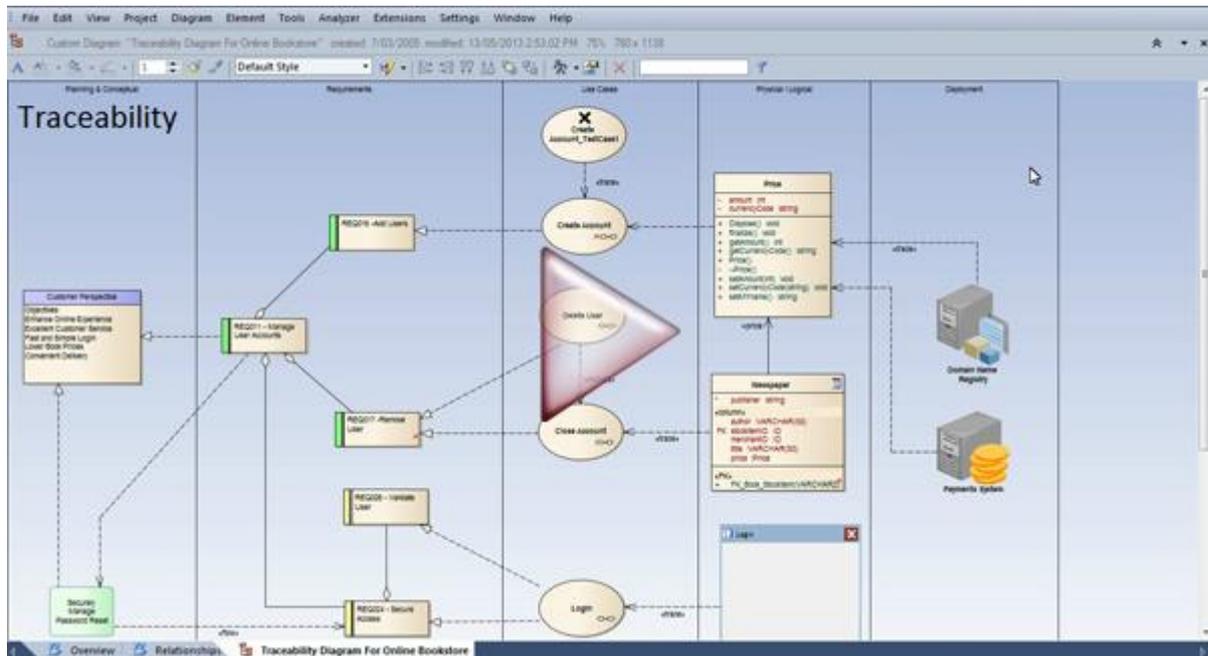
- Arrow Shapes
- Backgrounds
- Basic Flowchart Shapes
- Borders and Titles

**C:** The center of the screen shows the diagram workspace, which includes the actual diagram page as well as some blank space adjacent to the page.

**D:** The right side of the screen shows some help functions. Some people may choose to close this window to increase the area for diagram workspace, and re-open the help functions when necessary.

## Tool 2: Enterprise Architect

Enterprise architect is a visual modeling and design tool based on UML. The platform supports the design and construction of software systems, modeling business processes and modeling industry based domains. It is used by business and organizations to not only model the architecture of their systems. But to process the implementation of these models across the full application development life cycle.



**Sample screenshot from an Enterprise solution**

The intent of Enterprise architect is to determine how an organization can most effectively achieve its current and future objectives.

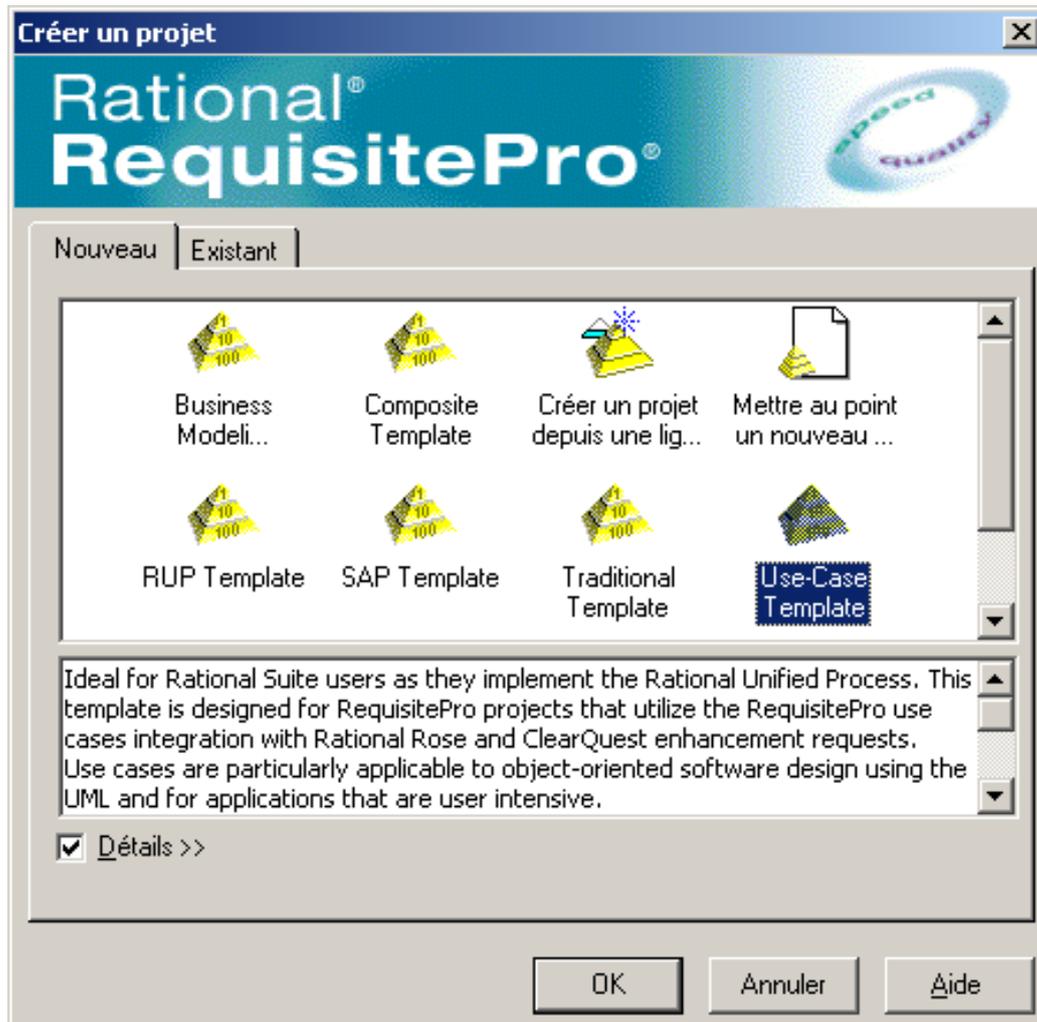
Enterprise architect has four points of view which are as follows:

- **Business perspective:** The Business perspective defines the processes and standards by which the business operates on day to day basis.
- **Application Perspective:** The application perspective defines the interactions among the processes and standards used by the organization.
- **Information Perspective:** This defines and classifies the raw data like document files, databases, images, presentations and spreadsheets that organization requires in order to efficiency operate.
- **Technology Prospective:** This defines the hardware, operating systems, programming and networking solutions used by organization.

## Tool 3: Rational Requisite Pro

The process of eliciting, documenting organizing tracking and changing Requirements and communicating this information across the project teams to ensure that iterative and unanticipated changes are maintained throughout the project life cycle.

Monitoring status and controlling changes to the requirement baseline. The Primary elements are Change control and Traceability.



Requisite Pro is used for the above activities and project administration purposes, the tool is used for querying and searching, Viewing the discussion that were part of the requirement.

In Requisite Pro, the user can work on the requirement document. The document is a MS-Word file created in Reqpro application and integrated with the project database. Requirements created outside Requisite pro can be imported or copied into the document.

In Requisite Pro, we can also work with traceability, here it is a dependency relationship between two requirements. Traceability is a methodical approach to managing change by linking requirements that are related to each other.

Requisite Pro makes it easy to track changes to a requirement throughout the development cycle, so it is not necessary to review all your documents individually to determine which elements need updating. You can view and manage suspect relationships using a Traceability Matrix or a Traceability Tree view.

Requisite Pro projects enable us to create a project framework in which the project artifacts are organized and managed. In each project the following are included

- General project information
- Packages
- General document information
- Document types
- Requirement types
- Requirement attributes
- Attribute values
- Cross-project traceability

Requisite Pro allows multiple user to access the same project documents and database simultaneously hence the project security aspect is to very crucial. Security prevents the system use, potential harm, or data loss from unauthorized user access to a project document.

It is recommended that the security is enabled for all RequisitePro projects. Doing so ensures that all changes to the project are associated with the proper username of the Individual who made the change, thereby ensuring that you have a complete audit trail for all changes.