SOFTWARE PROJECT MANAGEMENT

Project management

Refers to the coordination of people, resources, and schedules that are necessary to successfully produce a software product that is of high quality and delivered within the specific time period and budget.

Who does it?

A project manager is usually an experienced member of the team who essentially works as the administrative leader of the team.

Software project management complexities

Management of software projects is much more complex than management of many other types of projects. The main factors contributing to the complexity of managing a software project are the following:

Invisibility: Software remains invisible until its development is complete and it is operational. Anything that is invisible, is difficult to manage and control. The best that can do perhaps is to monitor the milestones that have been completed by the development team and the documents that have been produced.

Changeability: Because the software part of any system is easier to change as compared to the hardware part, the software part is the one that gets most frequently changed.
These changes usually arise from changes to the hardware or software (e.g. operating system, other applications), or just because the client changes his mind.

Complexity: Even moderate-sized software has many parts (functions) that interact with each other in many ways. Due to this complexity, many types of risks are associated with its development.

Uniqueness: Every software project is usually associated with many unique features or situations. This makes every project much different from the others. Due to the uniqueness of the software projects, a project manager faces many issues that are not familiar in the past.

Teamwork: Each member has to typically interact, review, and interface with several other members, adding another dimension to the complexity of software projects.

The Management Spectrum

Effective software project management focuses on the four Ps: people, product, process, and project.

The People

Every organization needs to continually improve its ability to attract, develop, motivate, and organize the work needed to accomplish its strategic business objectives.

The Product

Before a project can be planned, product objectives and scope should be established, alternative solutions should be considered, and technical and management constraints should be identified.

The Process

A software process provides the framework from which a comprehensive plan for software development can be established.

The Project

To avoid project failure, a software project manager and the software engineers must avoid a set of common warning signs, understand the critical success factors that lead to good project management, and develop a common-sense approach for planning, monitoring, and controlling the project.

Software Scope and Feasibility

The first software project management activity is the determination of software scope.

The scope is defined by answering the following questions:

Context. How does the software to be built fit into a larger system, product, or business context, and what constraints are imposed as a result of the context?

Information objectives. What customer-visible data objects are produced as output from the software? What data objects are required for input?

Function and performance. What function does the software perform to transform input data into output? Are any special performance characteristics to be addressed?

Once the scope has been identified, it is reasonable to ask: "Can we build software to meet this scope? Is the project feasible?".

Many times, software engineers rush past these questions to be in a project that is doomed from the beginning. Software engineers must try to determine if the system can be created using available technology, budget, time, and other resources.

Resources

Once the scope is defined, the required resources are estimated to build software.

Figure 4.1 depicts the three major categories of software engineering resources: people, reusable software components, and the development environment. Each resource is specified with four characteristics: a description of the resource, a statement of availability, the time when the resource will be required, and the duration of time that the resource will be applied.

Human Resources

Both organizational positions (e.g., manager, software engineer) and specialties (e.g., telecommunications, database, e-commerce) are specified.

For small projects, a single individual may perform all software engineering tasks. For larger projects, the software team may be geographically dispersed across a number of different locations.

The number of people required for a software project can be determined only after an estimate of development effort (e.g., person-months) is made.

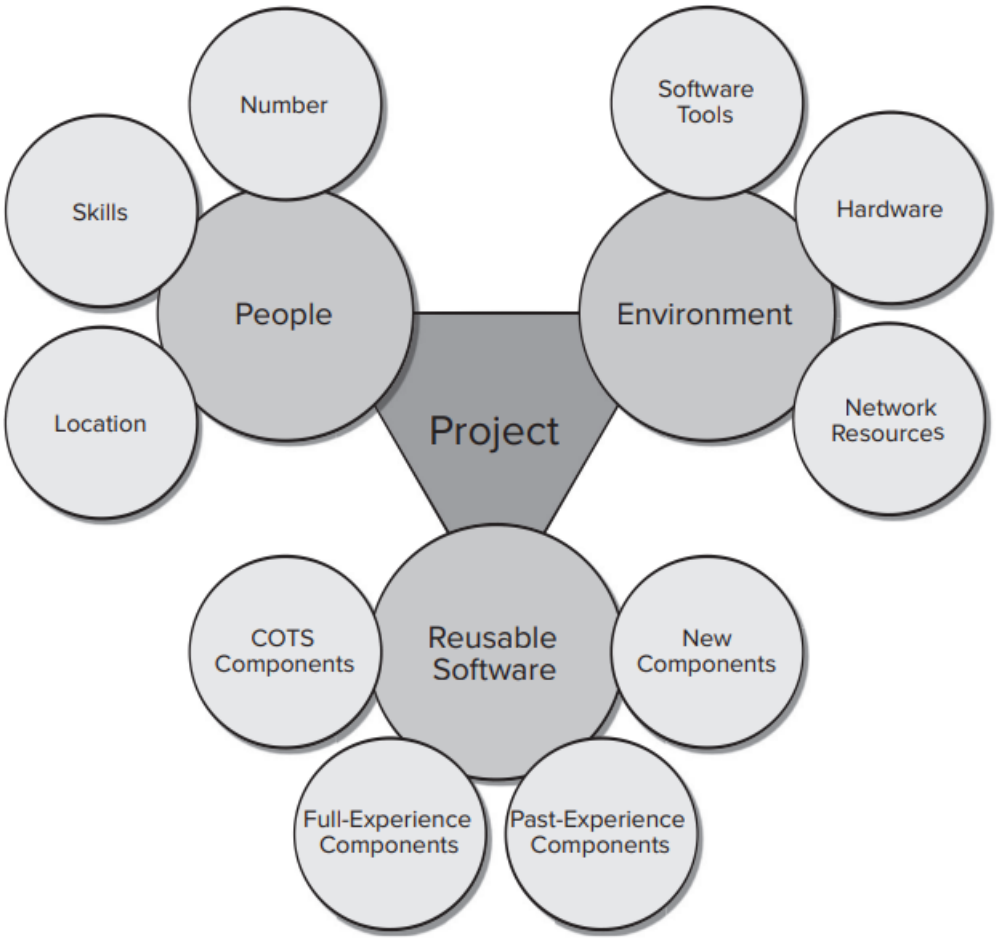

Figure 4.1: Project resources.

Reusable Software Resources

Component-based software engineering (CBSE) emphasizes reusability, that is, the creation and reuse of software building blocks. Such building blocks, often called components, must be cataloged for easy reference, standardized for easy application, and validated for easy integration.
It is important to consider whether it would be less costly to buy an existing software product than to build a custom software product from scratch.

Environmental Resources

The environment that supports a software project often called the software engineering environment (SEE), incorporates hardware and software.
When a system is to be engineered, the software team may require access to hardware elements being developed by other engineering teams. So, the time window required for hardware and software must be determined and verified that these resources will be available.

SOFTWARE RISKS

What is it? A risk is a potential problem for a software project it might happen, or it might not. It's a really good idea to identify it, assess its probability of occurrence, estimate its impact, and establish a contingency plan.
Who does it? Everyone involved in the software process—managers, software engineers, and other stakeholders—participates in risk analysis

and management.

Risk always involves two characteristics:

• Uncertainty:

The risk may or may not happen; that is, there are no 100% probable risks.

• Loss:

If the risk becomes a reality, unwanted consequences or losses will occur.

There are different categories of risks considered:

- Project risks threaten the project plan. That is, if project risks become real, it is likely that the project schedule will slip and that costs will increase. Project risks identify potential budget, schedule, staffing, resource, and requirements problems and their impact on a software project.

- Technical risks threaten the quality and timeliness of the software to be produced. If a technical risk becomes a reality, implementation may become difficult or impossible. Technical risks identify potential design, implementation, interface, and maintenance problems.

- Business risks threaten the software ability to work successfully. Like building an excellent system that no one really wants (market risk), and building a product that the sales don't understand how to sell (sales risk).

RISK IMPACT AND PROBABILITY

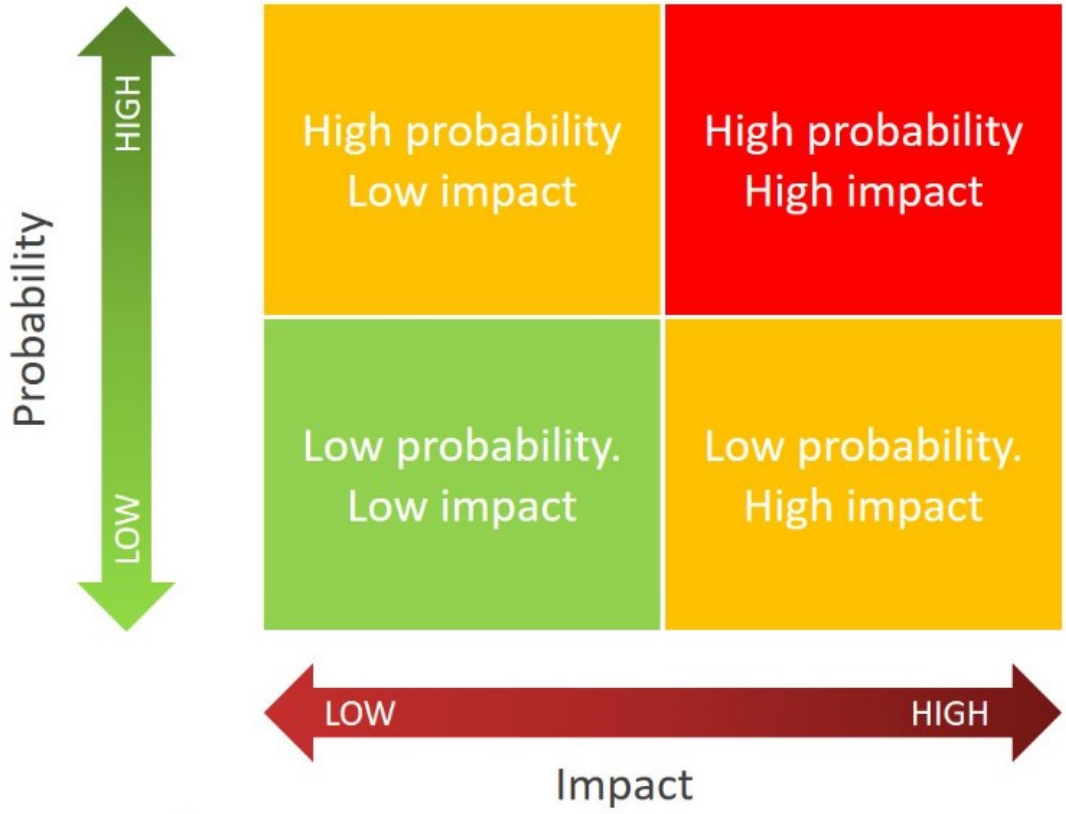

Figure 4.2: Risk impact and probability chart.

Low impact/low probability

Those risks are having a low probability of occurrence, and little impact on risk. So, this risk is not essential to take immediate action. (priority=4, impact value= low).

Low impact/high probability

Those risks are having a high probability of occurrence, and little effect. So, this risk is medium essential to take immediate action.

(priority=3, impact value= medium)

High impact/low probability

Those risks are having a low probability of occurrence, and a high impact of risk. This risk is highly significant to take immediate action.

(priority=2, impact value= significant)

High impact/high probability

Those risks are having a high probability of occurrence, and a high impact risk. This risk is critical and most important to take immediate action.

(priority=1, impact value= critical)