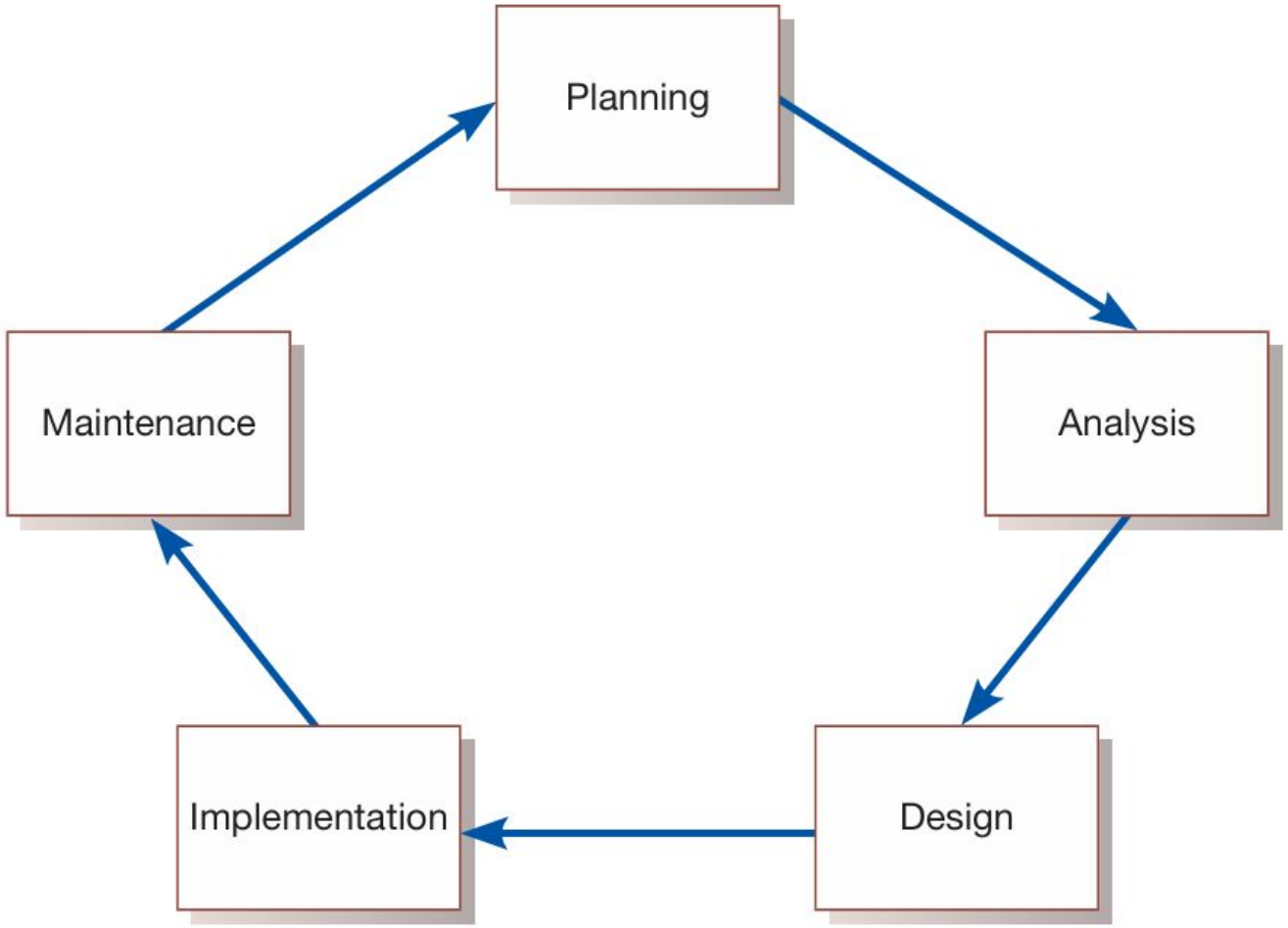
System Anaylsis

Ibrahim Elsemman

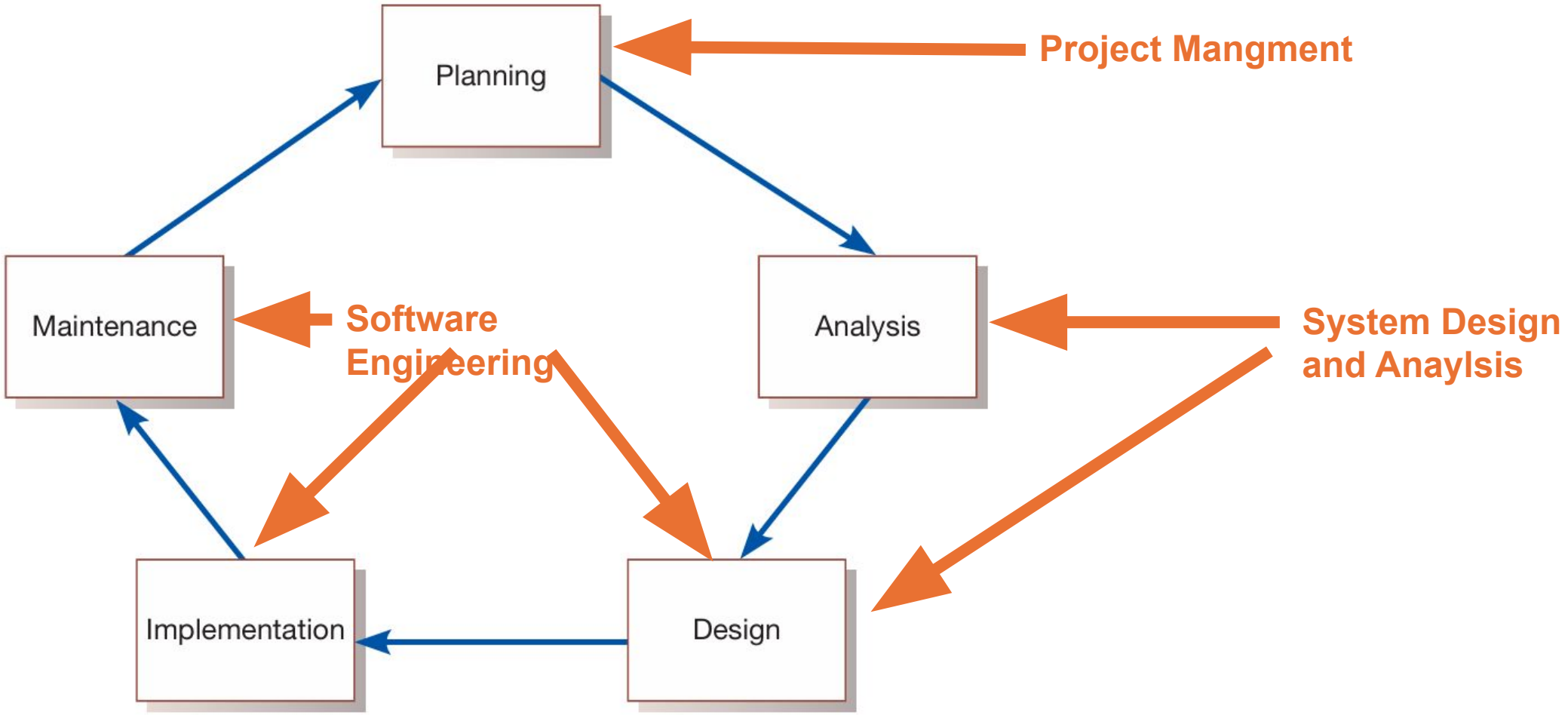
Four Important Courses to Make Enterprise software

- Project Mangment
- System Design and Anaylsis
- Software Engineering
- User Experience (UX)

Systems development life cycle



Systems development life cycle

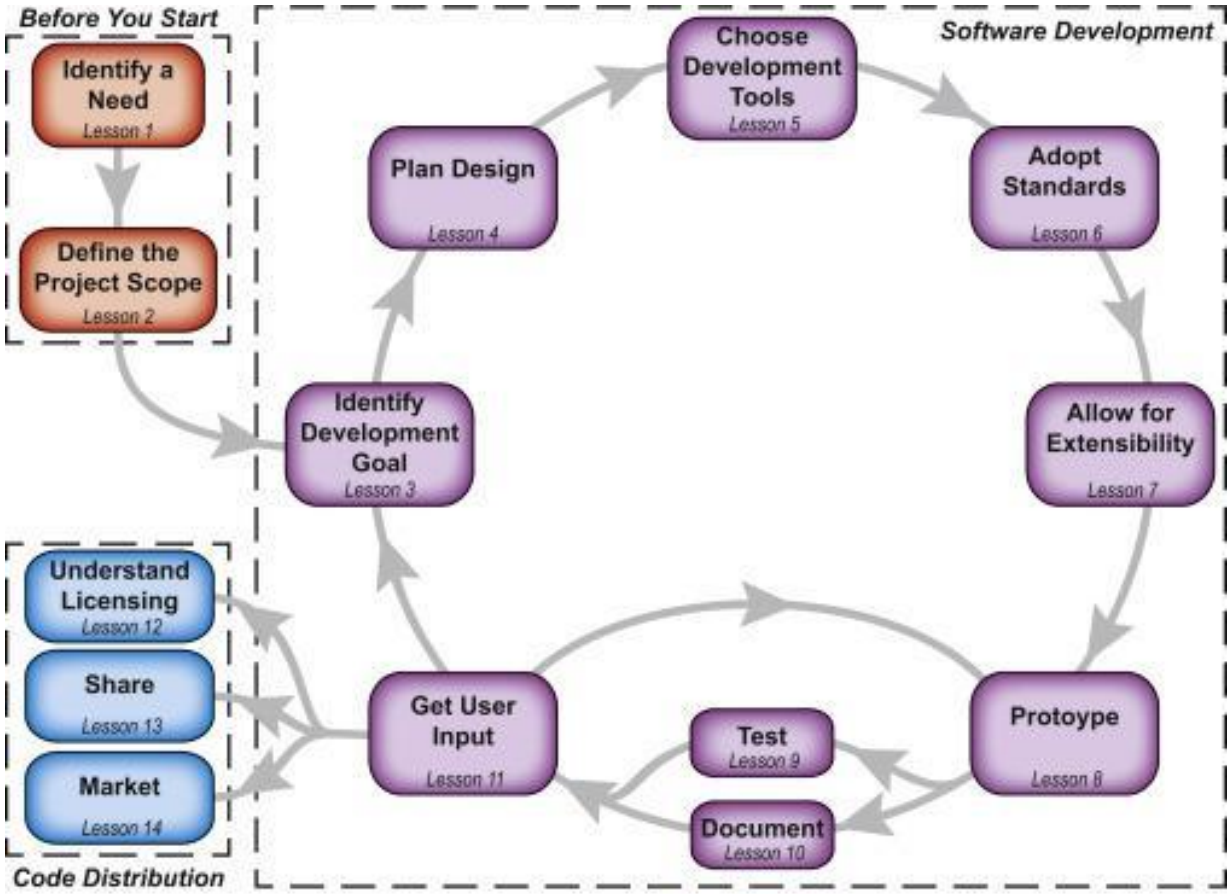


Do we really need project managment in software developlment?

- A survey of hundreds of corporate software development projects indicated that five out of six software projects are considered unsuccessful [Johnson 95], and approximately a third of software projects are canceled. The remaining projects delivered software at almost double the expected budget and time to develop as originally planned.

William J. Brown et al, AntiPatterns Refactoring Software, Architectures, and Projects in Crisis

Software Development Process



Visual Description of How the 14 steps during the Software Development Process

What Is a Project?

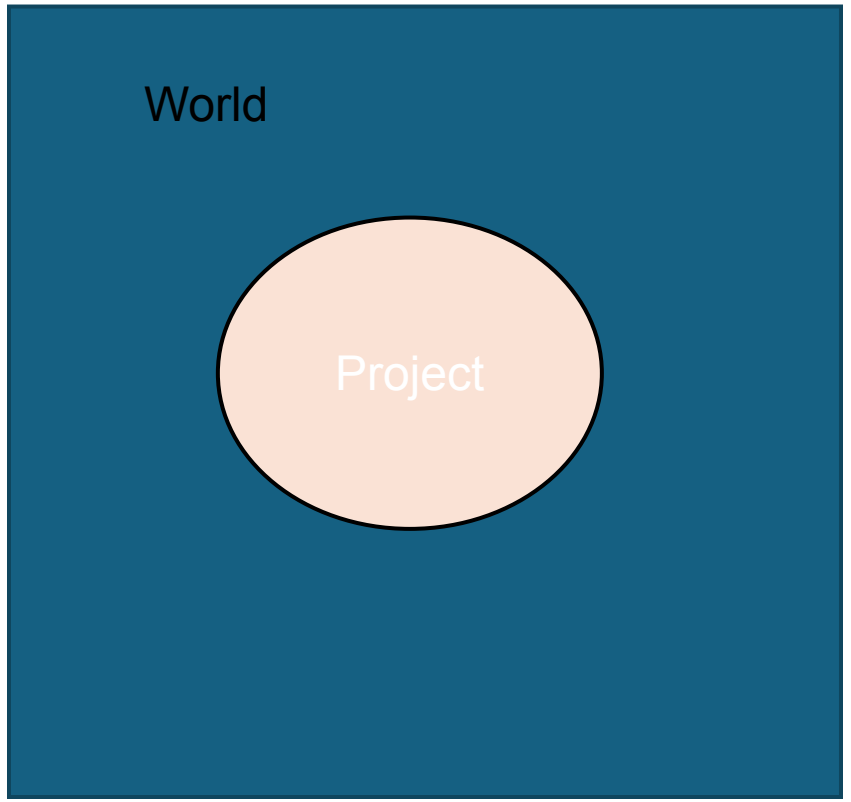
- *A **project** is a temporary endeavor undertaken to create a **unique product, service, or result**.*

Defining the Project Scope

- You should know that:
- My project do these, but not do these

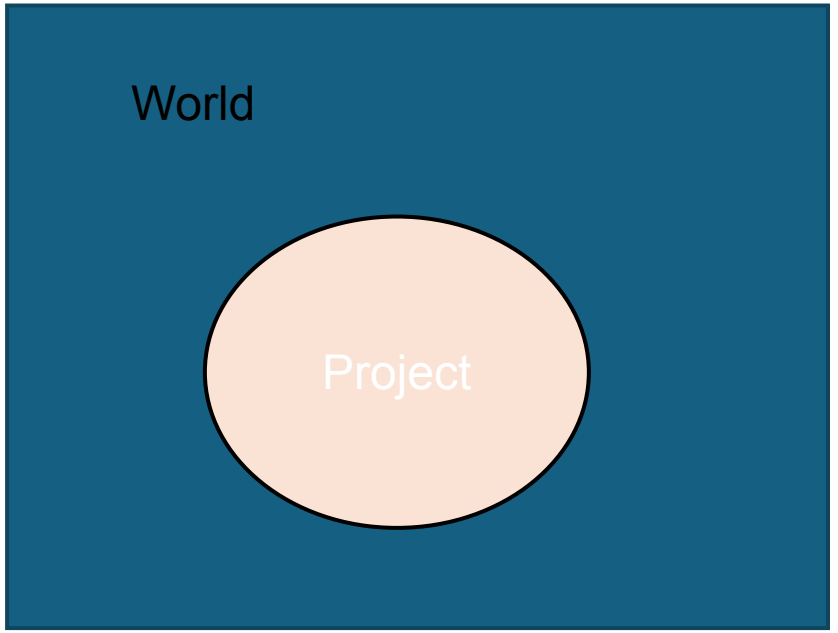
Project Scope Checklist

1. Project objective
2. Deliverables
3. Milestones
4. Technical requirements
5. Limits and exclusions
6. Reviews with customer



Defining the Project Scope

- You should know that:
- My project do these, but not do these



Project Scope Checklist

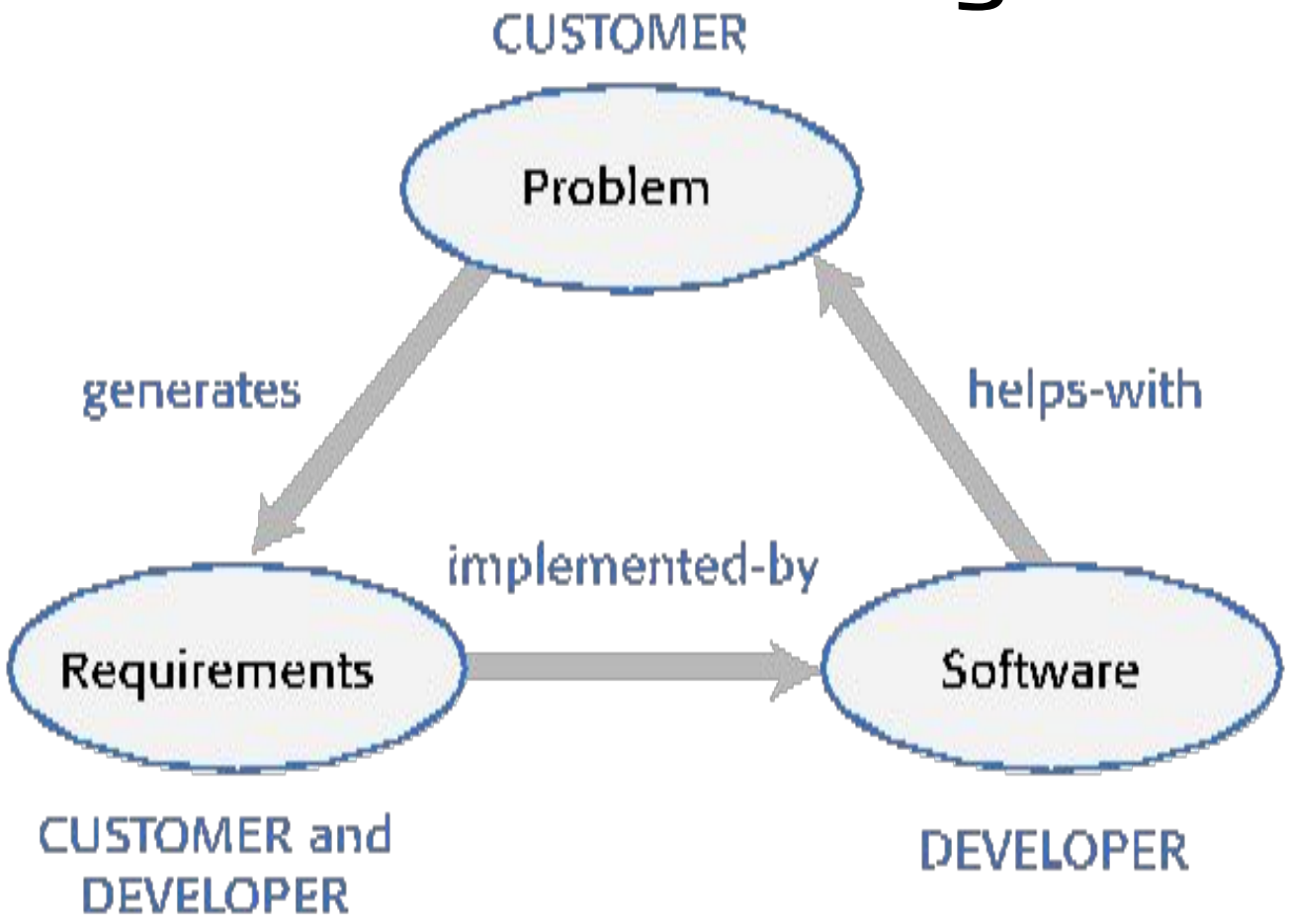
1. Project objective
2. Deliverables
3. Milestones
4. Technical requirements
5. Limits and exclusions
6. Reviews with customer

First Step in Software Engineering

Software products

- Software products are generic software systems that provide functionality that is useful to a range of customers.
- Many different types of products are available from large-scale business systems (e.g. MS Excel) through personal products (e.g. Evernote) to simple mobile phone apps and games (e.g. Suduko).
- Software product engineering methods and techniques have evolved from software engineering techniques that support the development of one-off, custom software systems.
- Custom software systems are still important for large businesses, government and public bodies. They are developed in dedicated software projects.

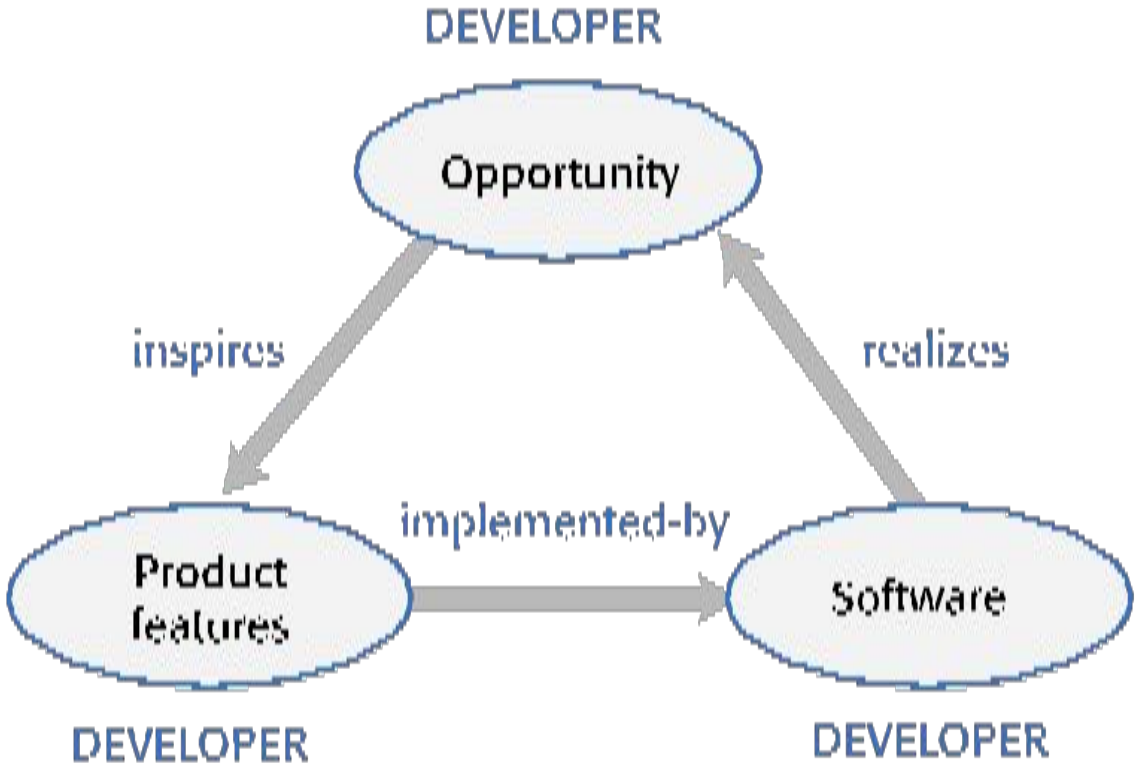
Project-based software engineering



Project-based software engineering

- The starting point for the software development is a set of ‘software requirements’ that are owned by an external client and which set out what they want a software system to do to support their business processes.
- The software is developed by a software company (the contractor) who design and implement a system that delivers functionality to meet the requirements.
- The customer may change the requirements at any time in response to business changes (they usually do). The contractor must change the software to reflect these requirements changes.
- Custom software usually has a long-lifetime (10 years or more) and it must be supported over that lifetime.

Product software engineering



Product software engineering

- The starting point for product development is a business opportunity that is identified by individuals or a company. They develop a software product to take advantage of this opportunity and sell this to customers.
- The company who identified the opportunity design and implement a set of software features that realize the opportunity and that will be useful to customers.
- The software development company are responsible for deciding on the development timescale, what features to include and when the product should change.
- Rapid delivery of software products is essential to capture the market for that type of product.

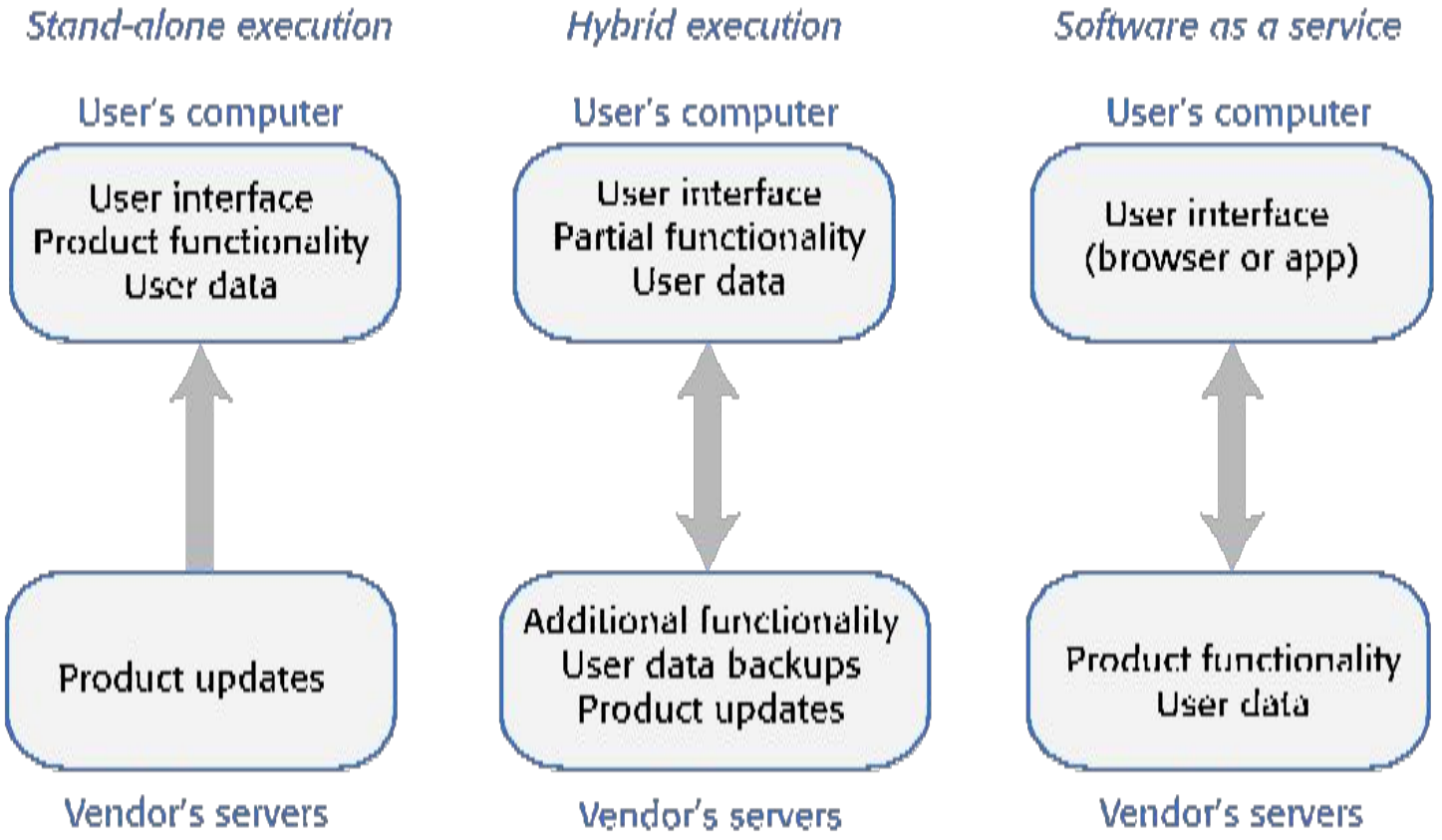
Software product lines and platforms

- ***Software product line***
  A set of software products that share a common core. Each member of the product line includes customer-specific adaptations and additions. Software product lines may be used to implement a custom system for a customer with specific needs that can't be met by a generic product.
- ***Platform***
  A software (or software+hardware) product that includes functionality so that new applications can be built on it. An example of a platform that you probably use is Facebook. It provides an extensive set of product functionality but also provides support for creating 'Facebook apps'. These add new features that may be used by a business or a Facebook interest group.

Software execution models

- ***Stand-alone*** The software executes entirely on the customer's computers.
- ***Hybrid*** Part of the software's functionality is implemented on the customer's computer but some features are implemented on the product developer's servers.
- ***Software service*** All of the product's features are implemented on the developer's servers and the customer accesses these through a browser or a mobile app.

Software execution models



Comparable software development

- The key feature of product development is that there is no external customer that generates requirements and pays for the software. This is also true for other types of software development:
  - ***Student projects*** Individuals or student groups develop software as part of their course. Given an assignment, they decide what features to include in the software.
  - ***Research software*** Researchers develop software to help them answer questions that are relevant to their research.
  - ***Internal tool development*** Software developers may develop tools to support their work - in essence, these are internal products that are not intended for customer release.

The product vision

- The starting point for software product development is a ‘product vision’.
- Product visions are simple statements that define the essence of the product to be developed.
- The product vision should answer three fundamental questions:
  - What is the product to be developed?
  - Who are the target customers and users?
  - Why should customers buy this product?

Moore's vision template

- FOR (target customer)
- WHO (statement of the need or opportunity)
- The (PRODUCT NAME) is a (product category)
- THAT (key benefit, compelling reason to buy)
- UNLIKE (primary competitive alternative)
- OUR PRODUCT (statement of primary differentiation)

Vision template example

“FOR a mid-sized company's marketing and sales departments WHO need basic CRM functionality, THE CRM-Innovator is a Web-based service THAT provides sales tracking, lead generation, and sales representative support features that improve customer relationships at critical touch points. UNLIKE other services or package software products, OUR product provides very capable services at a moderate cost.”

Table 1.2 Information sources for developing a product vision

- ***Domain experience***
  The product developers may work in a particular area (say marketing and sales) and understand the software support that they need. They may be frustrated by the deficiencies in the software they use and see opportunities for an improved system.

- ***Product experience***
  Users of existing software (such as word processing software) may see simpler and better ways of providing comparable functionality and propose a new system that implements this. New products can take advantage of recent technological developments such as speech interfaces.

- ***Customer experience***
  The software developers may have extensive discussions with prospective customers of the product to understand the problems that they face, constraints, such as interoperability, that limit their flexibility to buy new software, and the critical attributes of the software that they need.

- ***Prototyping and playing around***
  Developers may have an idea for software but need to develop a better understanding of that idea and what might be involved in developing it into a product. They may develop a prototype system as an experiment and ‘play around’ with ideas and variations using that prototype system as a platform.

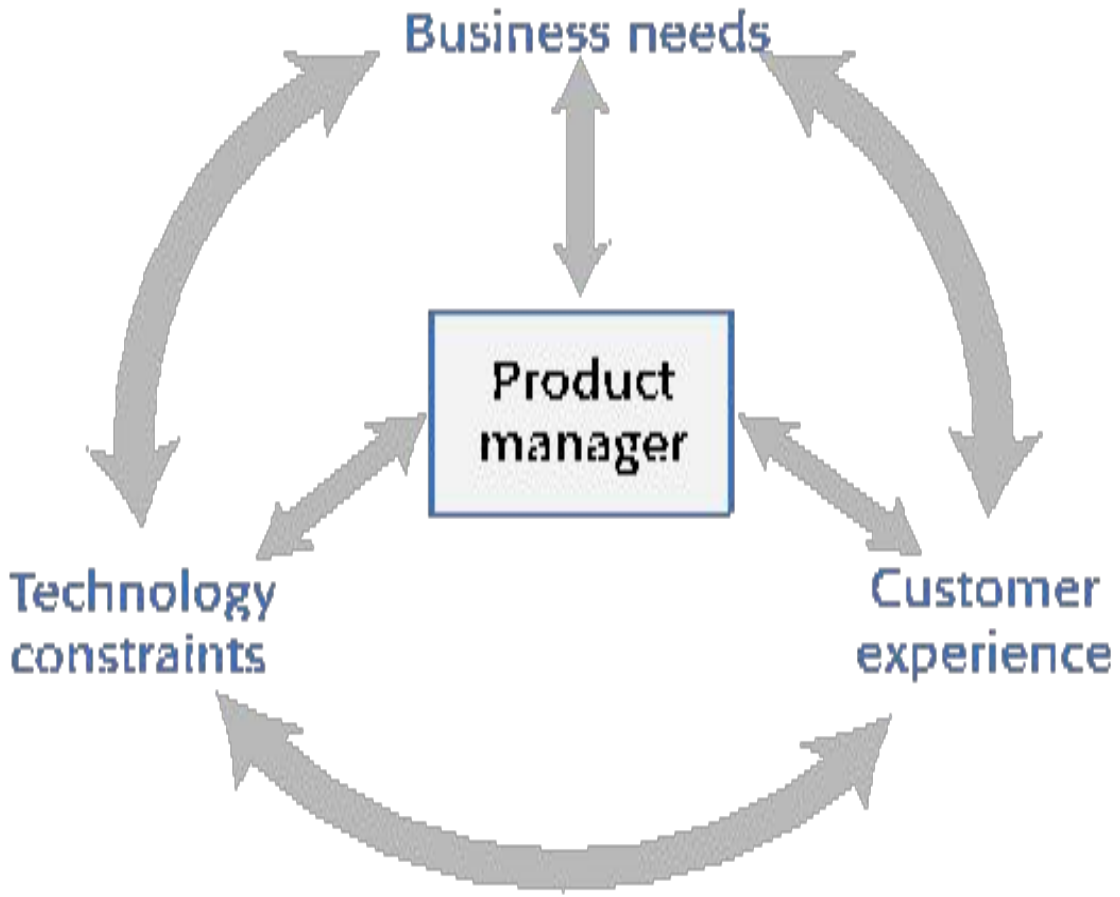
Table 1.3 A vision statement for the iLearn system

- **FOR** teachers and educators **WHO** need a way to help students use web-based learning resources and applications, **THE** iLearn system is an open learning environment **THAT** allows the set of resources used by classes and students to be easily configured for these students and classes by teachers themselves. **UNLIKE** Virtual Learning Environments, such as Moodle, the focus of iLearn is the learning process rather than the administration and management of materials, assessments and coursework. **OUR** product enables teachers to create subject and age-specific environments for their students using any web-based resources, such as videos, simulations and written materials that are appropriate.
- Schools and universities are the target customers for the iLearn system as it will significantly improve the learning experience of students at relatively low cost. It will collect and process learner analytics that will reduce the costs of progress tracking and reporting.

Software product management

- Software product management is a business activity that focuses on the software products developed and sold by the business.
- Product managers (PMs) take overall responsibility for the product and are involved in planning, development and product marketing.
- Product managers are the interface between the organization, its customers and the software development team. They are involved at all stages of a product's lifetime from initial conception through to withdrawal of the product from the market.
- Product managers must look outward to customers and

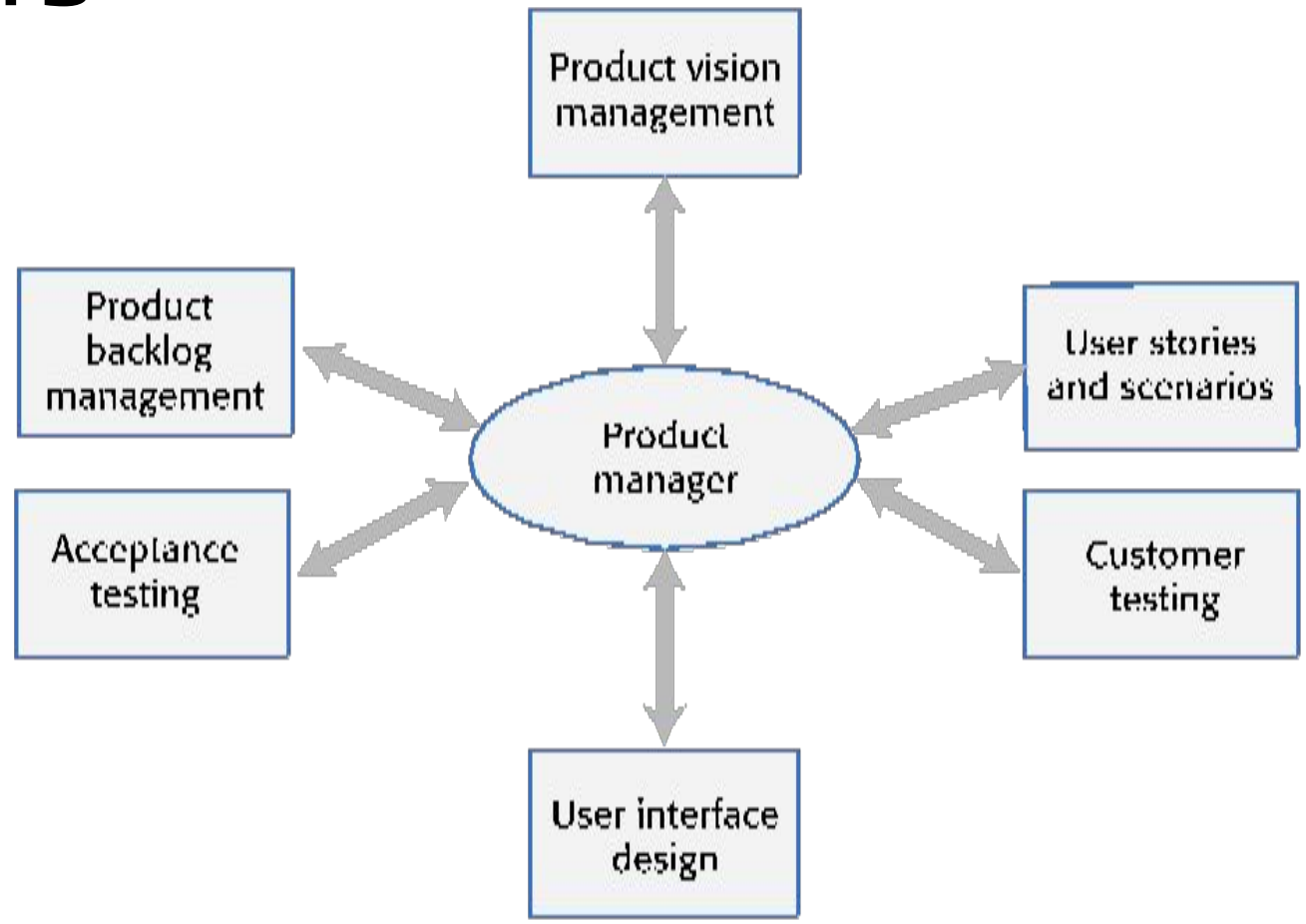
Product management concerns



Product management concerns

- ***Business needs*** PMs have to ensure that the software being developed meets the business goals of the software development company.
- ***Technology constraints*** PMs must make developers aware of technology issues that are important to customers.
- ***Customer experience*** PMs should be in regular contact with customers and potential customers to understand what they are looking for in a product, the types of users and their backgrounds and the ways that the product may be used.

Technical interactions of product managers



Technical interactions of product managers

- Product vision management
  - The product manager may be responsible for helping with the development of the product vision. The should always be responsible for managing the vision, which involves assessing and evaluating proposed changes against the product vision. They should ensure that there is no ‘vision drift’
- Product roadmap development
  - A product roadmap is a plan for the development, release and marketing of the software. The PM should lead roadmap development and should be the ultimate authority in deciding if changes to the roadmap should be made.
- User story and scenario development
  - User stories and scenarios are used to refine a product vision and identify product features. Based on his or her knowledge of customers, the PM should lead the development of stories and scenarios.

Technical interactions of product managers

- Product backlog creation and management
  - The product backlog is a prioritized ‘to-do’ list of what has to be developed. PMs should be involved in creating and refining the backlog and deciding on the priority of product features to be developed.
- Acceptance testing
  - Acceptance testing is the process of verifying that a software release meets the goals set out in the product roadmap and that the product is efficient and reliable. The PM should be involved in developing tests of the product features that reflect how customers use the product.
- Customer testing
  - Customer testing involves taking a release of a product to customers and getting feedback on the product’s features, usability and business. PMs are involved in selecting customers to be involved in the customer testing process and working with them during that process.
- User interface design
  - Product managers should understand user limitations and act as surrogate users in their interactions with the development team. They should evaluate user interface features as they are developed to check that these features are not unnecessarily complex or force users to work in an unnatural way.

Product prototyping

- Product prototyping is the process of developing an early version of a product to test your ideas and to convince yourself and company funders that your product has real market potential.
  - You may be able to write an inspiring product vision, but your potential users can only really relate to your product when they see a working version of your software. They can point out what they like and don't like about it and make suggestions for new features.
  - A prototype may be also used to help identify fundamental software components or services and to test technology.
- Building a prototype should be the first thing that you do when developing a software product. Your aim should be to have a working version of your software that can be used to demonstrate its key features.
- You should always plan to throw-away the prototype after development and to re-implement the software, taking account of issues such as security and reliability.

Two-stage prototyping

- ***Feasibility demonstration*** You create an executable system that demonstrates the new ideas in your product. The aims at this stage are to see if your ideas actually work and to show funders and/or company management the original product features that are better than those in competing products.
- ***Customer demonstration*** You take an existing prototype created to demonstrate feasibility and extend this with your ideas for specific customer features and how these can be realized. Before you develop this type of prototype, you need to do some user studies and have a clearer idea of your potential users and scenarios of use.