CS6403 Software Engineering Lecture Notes

UNIT I

SOFTWARE PROCESS AND PROJECT MANAGEMENT

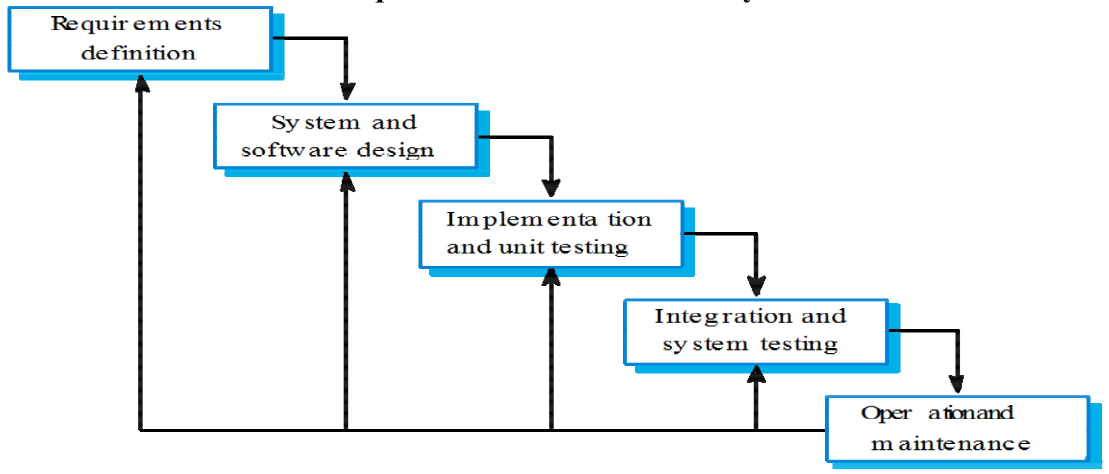
Software engineering paradigm:

- The framework activities will always be applied on every project ... BUT the tasks (and degree of rigor) for each activity will vary based on:
  - the type of project
  - characteristics of the project
  - common sense judgment; concurrence of the project team

The software process:

- A structured set of activities required to develop a software system
  - Specification;
  - Design;
  - Validation;
  - Evolution.
- A software process model is an abstract representation of a process. It presents a description of a process from some particular perspective.

Waterfall model/Linear Sequential Model/classic life cycle :


Requirements definition
System and software design
Implementation and unit testing
Integration and system testing
Operation and maintenance

- **Systems E*ngineering***
  - Software ***as part of larger system, determine requirements for all system elements, allocate requirements to*** software.
- **Software *Requirements Analysis***
  - Develop understanding of problem domain, user needs, function, performance, interfaces, **...**
  - ***Software*** Design
  - ***Multi-step process to determine architecture, interfaces, data structures, functional detail. Produces (high-level) form that can be checked for quality, conformance before coding.***
- ***Coding***
  - Produce machine readable and executable form, match HW, OS and design needs**.**
- ***Testing***

  - Confirm that components, subsystems and complete products meet requirements, specifications and quality, find and fix defects.
- ***Maintenance***
  - Incrementally, ***evolve software*** to fix defects, add features, adapt to new condition. Often 80% of effort spent here*!*

Waterfall model phases:
- Requirements analysis and definition
- System and software design
- Implementation and unit testing
- Integration and system testing
- Operation and maintenance
- The main drawback of the waterfall model is the difficulty of accommodating change after the process is underway. One phase has to be complete before moving onto the next phase.
- Each phase terminates only when the documents are complete and approved by the SQA group.
- Maintenance begins when the client reports an error after having accepted the product. It could also begin due to a change in requirements after the client has accepted the product

Waterfall model: Advantages:
- Disciplined approach
- Careful checking by the Software Quality Assurance Group at the end of each phase.
- Testing in each phase.
- Documentation available at the end of each phase.

Waterfall model problems:
- It is difficult to respond to changing customer requirements.
- Therefore, this model is only appropriate when the requirements are well-understood and changes will be fairly limited during the design process.
- Few business systems have stable requirements.
- The waterfall model is mostly used for large systems engineering projects where a system is developed at several sites.
- The customer must have patience. A working version of the program will not be available until late in the project time-span
- Feedback from one phase to another might be too late and hence expensive.

The Prototyping Models:
- Often, a customer defines a set of general objectives for software but does not identify detailed input, processing, or output requirements.
- In other cases, the developer may be unsure of the efficiency of an algorithm, the adaptability of an operating system, or the form that human –machine interaction should take
- In this case prototyping paradigm may offer the best approach
- Requirements gathering
- Quick design
- Prototype building
- Prototype evaluation by customers
- Prototype may be refined

- Prototype thrown away and software developed using formal process{ it is used to define the requirement} Prototyping

Strengths:

- Requirements can be set earlier and more reliably
- Customer sees results very quickly.
- Customer is educated in what is possible helping to refine requirements.
- Requirements can be communicated more clearly and completely
- Between developers and clients Requirements and design options can be investigated quickly and Cheaply

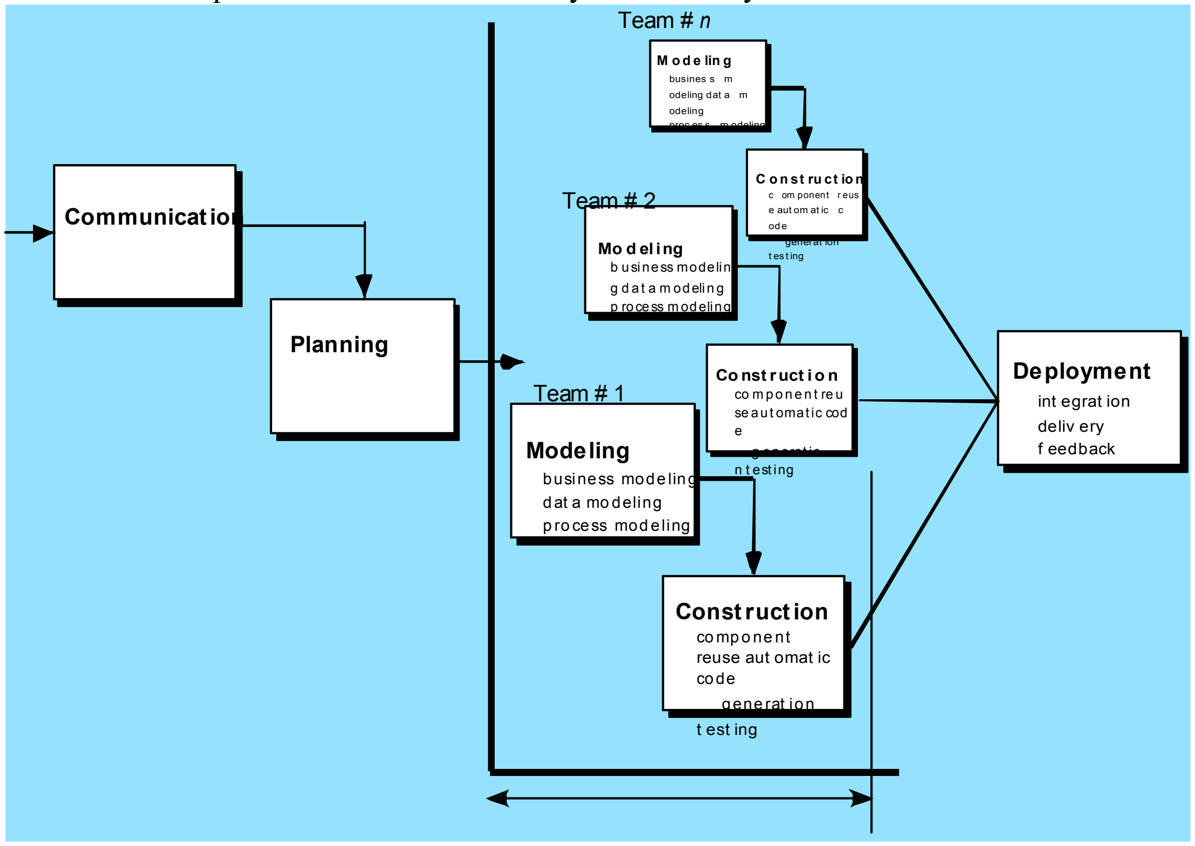
Weaknesses:

– Requires a rapid prototyping tool and expertise in using it–a cost for the development organisation

– Smoke and mirrors - looks like a working version, but it is not.

The RAD Model:

- Rapid Application Development is a linear sequential software development process model that emphasizes an extremely short development cycle
- Rapid application achieved by using a component based construction approach
- If requirements are well understood and project scope is constrained the RAD process enables a development team to create a –fully functional system‖


Communication
Planning
Team # n
Modeling
business modeling data modeling process modeling
Construction
component reuse automatic code generation testing
Team # 2
Modeling
business modeling data modeling process modeling
Construction
component reuse automatic code generation testing
Team # 1
Modeling
business modeling
data modeling
process modeling
Construction
component reuse automatic code generation testing
Deployment
integration
delivery
feedback

6 0 - 9 0 d ays

RAD phases :

- Business modeling
- Data modeling
- Process modeling

- Application generation
- Testing and turnover

Business modeling:

- What information drives the business process?
- What information is generated?
- Who generates it?

Data Modeling:

- The information flow defined as part of the business modeling phase is refined into a set of data objects that are needed to support the business.
- The characteristics (called attributes) of each object are identified and the relationships between these objects are defined

Process modeling:

- *The data modeling phase are transformed to achieve the information flow necessary to implement a business function.*
- *Processing descriptions are created for adding , modifying, deleting, or retrieving a data object*

Application generation:

- *RAD assumes the use of 4 generation techniques.*
- *Rather than creating software using conventional 3 generation programming languages, the RAD process works to reuse existing program components (when possible) or created reusable components (when necessary)*

Testing and Turnover:

- *Since the RAD process emphasizes reuse, many of the program components have already been testing.*
- *This reduces over all testing time.*
- *However, new components must be tested and all interfaces must be fully exercised*

Advantages &Disadvantages of RAD:

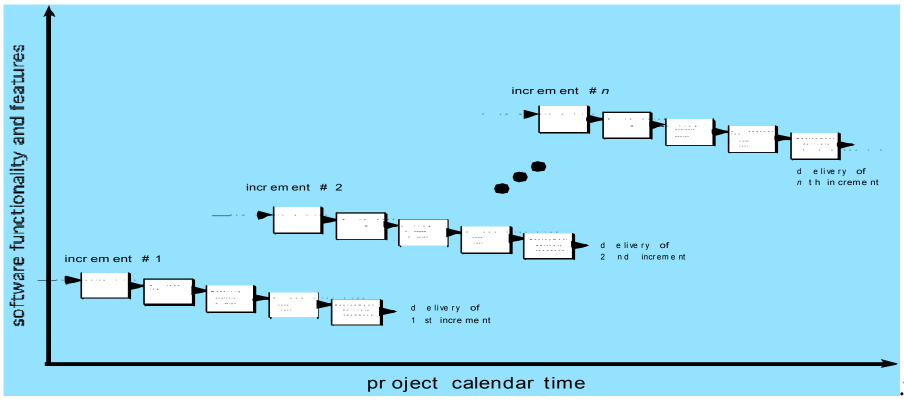
Advantages

- *Extremely short **development** time.*
- *Uses component-based construction and emphasises reuse and code generation*

Disadvantages

- ***Large human resource requirements (to create all of the teams).***
- *Requires strong commitment between developers and customers for "**rapid-**fire" activities.*
- ***High performance requirements maybe can't be met (requires tuning the components).***

The Incremental Model

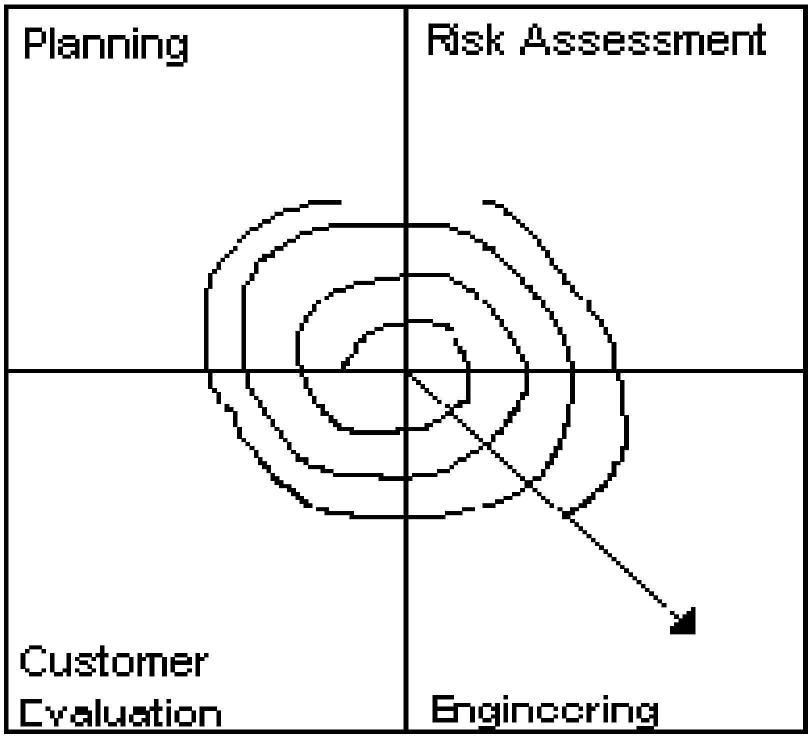

software functionality and features
increment # n
delivery of nth increment
increment # 2
delivery of 2nd increment
increment # 1
delivery of 1st increment
project calendar time

The Incremental development

- *Combination of linear + prototype*
- *Rather than deliver the system as a single delivery, the development and delivery is broken down into increments with each increment delivering part of the required functionality*
- User requirements are prioritised and the highest priority requirements are included in early increments
- Once the development of an increment is started, the requirements are frozen though requirements for later increments can continue to evolve

  Incremental development advantages:
- The customer is able to do some useful work after release
- Lower risk of overall project failure
- The highest priority system services tend to receive the most testing

Spiral Model:



Spiral model sectors:

- Customer communication

  Tasks required to establish effective communication between developer and customer
- Planning

  The tasks required to define recourses, timelines, and project is reviewed and the next phase of the spiral is planned
- Risk analysis
  - Risks are assessed and activities put in place to reduce the key
- Risks engineering
  - Tasks required to build one or more representations of the application

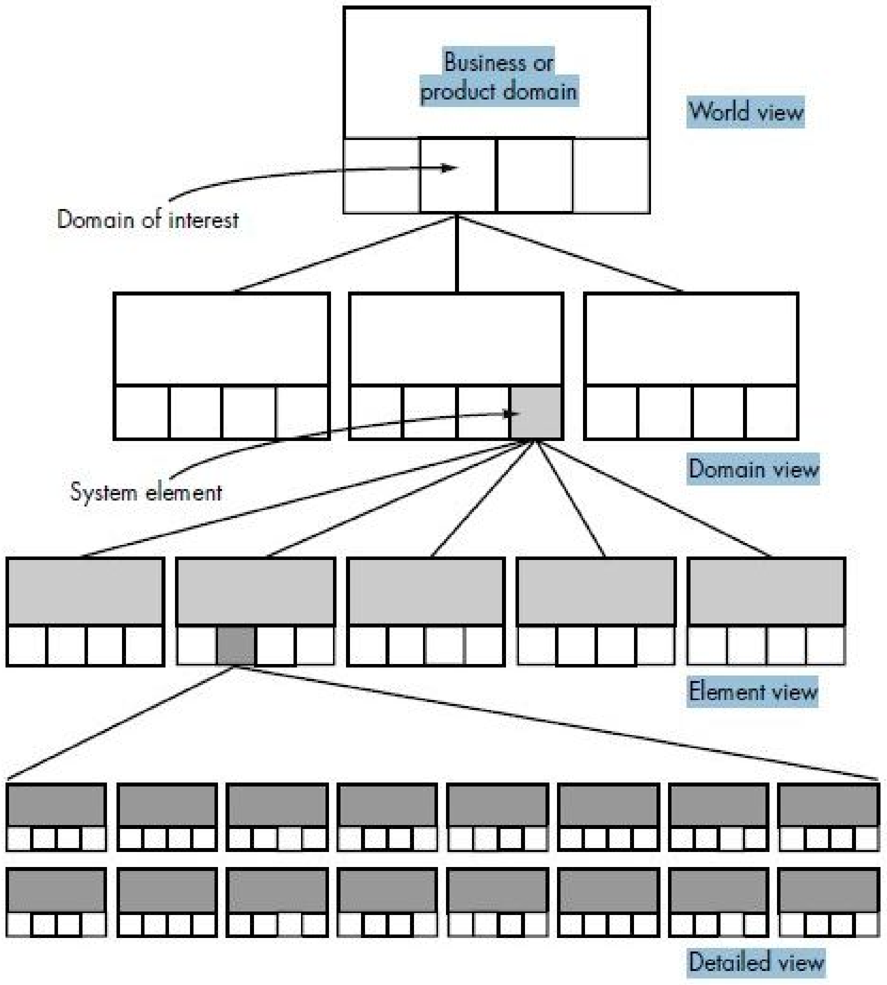
- Construction & release
  - Tasks required to construct, test, install and provide user support (e.g documentation and training)
- Customer evaluation
  - Customer feedback collected every stage

Spiral Model Advantages:

- Focuses attention on reuse options.
- Focuses attention on early error elimination.
- Puts quality objectives up front.
- Integrates development and maintenance.
- Provides a framework for hardware/software Development.

System Engineering

- Software engineering occurs as a consequence of a process called system engineering.
- Instead of concentrating solely on software, system engineering focuses on a variety of elements, analyzing, designing, and organizing those elements into a system that can be a product, a service, or a technology for the transformation of information or control.



- The system engineering process usually begins with a –world view.‖ That is, the entire business or product domain is examined to ensure that the proper business or technology context can be established.
- The world view is refined to focus more fully on specific domain of interest. Within a specific domain, the need for targeted system elements (e.g., data, software, hardware, people) is analyzed. Finally, the analysis, design, and construction of a targeted system element is initiated.
- At the top of the hierarchy, a very broad context is established and, at the bottom, detailed technical activities, performed by the relevant engineering discipline (e.g., hardware or software engineering), are conducted.
- Stated in a slightly more formal manner, the world view (WV) is composed of a set of domains (*Di*), which can each be a system or system of systems in its own right.

  $$WV = \{D1, D2, D3, \ldots, Dn\}$$
- Each domain is composed of specific elements (*Ej*) each of which serves some role in accomplishing the objective and goals of the domain or component:

  $$Di = \{E1, E2, E3, \ldots, Em\}$$
- Finally, each element is implemented by specifying the technical components (*Ck*) that achieve the necessary function for an element:

  $$Ej = \{C1, C2, C3, \ldots, Ck\}$$

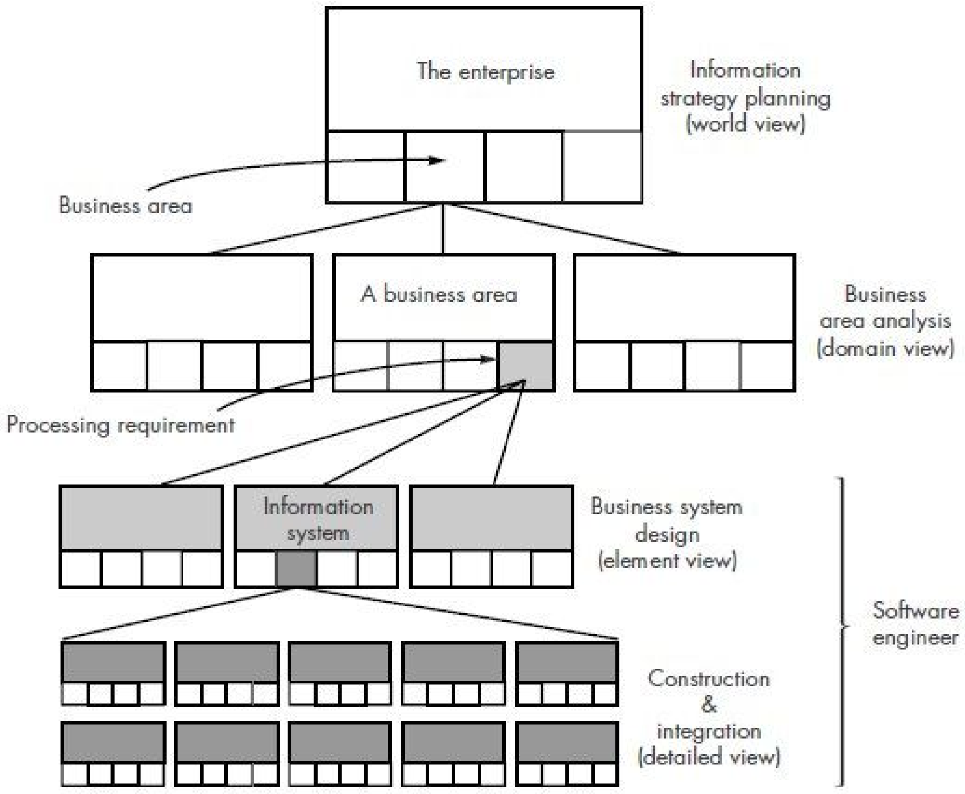
Computer Based System

- computer-based system as A set or arrangement of elements that are organized to accomplish some predefined goal by processing information.
- The goal may be to support some business function or to develop a product that can be sold to generate business revenue.
- To accomplish the goal, a computer-based system makes use of a variety of system elements:
  1. **Software.** Computer programs, data structures, and related documentation that serve to effect the logical method, procedure, or control that is required.
  2. **Hardware.** Electronic devices that provide computing capability, the interconnectivity devices (e.g., network switches, telecommunications devices) that enable the flow of data, and electromechanical devices (e.g., sensors, motors, pumps) that provide external world function.
  3. **People.** Users and operators of hardware and software.
  4. **Database.** A large, organized collection of information that is accessed via software.
  5. **Documentation.** Descriptive information (e.g., hardcopy manuals, on-line help files, Web sites) that portrays the use and/or operation of the system.
  6. **Procedures.** The steps that define the specific use of each system element or the procedural context in which the system resides.
- The elements combine in a variety of ways to transform information. For example, a marketing department transforms raw sales data into a profile of the typical purchaser of a product; a robot transforms a command file containing specific instructions into a set of control signals that cause some specific physical action.
- Creating an information system to assist the marketing department and control software to support the robot both require system engineering.

- One complicating characteristic of computer-based systems is that the elements constituting one system may also represent one macro element of a still larger system. The macro element is a computer-based system that is one part of a larger computer-based system.
- As an example, we consider a "factory automation system" that is essentially a hierarchy of systems. At the lowest level of the hierarchy we have a numerical control machine, robots, and data entry devices.
- Each is a computerbased system in its own right. The elements of the numerical control machine include electronic and electromechanical hardware (e.g., processor and memory, motors, sensors), software (for communications, machine control, interpolation), people (the machine operator), a database (the stored NC program), documentation, and procedures.
- A similar decomposition could be applied to the robot and data entry device. Each is a computer-based system.
- At the next level in the hierarchy, a manufacturing cell is defined. The manufacturing cell is a computer-based system that may have elements of its own (e.g., computers, mechanical fixtures) and also integrates the macro elements that we have called numerical control machine, robot, and data entry device.

Business Process Engineering Overview

- The goal of business process engineering (BPE) is to define architectures that will enable a business to use information effectively.
- When taking a world view of a company‘s information technology needs, there is little doubt that system engineering is required. Not only is the specification of the appropriate computing architecture required, but the software architecture that populates the –unique configuration of heterogeneous computing resources‖ must be developed.
- Business process engineering is one approach for creating an overall plan for implementing the computing architecture .
- Three different architectures must be analyzed and designed within the context of business objectives and goals:
  - data architecture
  - applications architecture
  - technology infrastructure
- The *data architecture* provides a framework for the information needs of a business or business function. The individual building blocks of the architecture are the data objects that are used by the business. A data object contains a set of attributes that define some aspect, quality, characteristic, or descriptor of the data that are being described.
- The *application architecture* encompasses those elements of a system that transform objects within the data architecture for some business purpose. In the context of this book, we consider the application architecture to be the system of programs (software) that performs this transformation. However, in a broader context, the application architecture might incorporate the role of people (who are information transformers and users) and business procedures that have not been automated.
- The *technology infrastructure* provides the foundation for the data and application architectures. The infrastructure encompasses the hardware and software that are used to support the application and data. This includes computers, operating systems, networks, telecommunication links, storage technologies, and the architecture (e.g., client/server) that has been designed to implement these technologies.

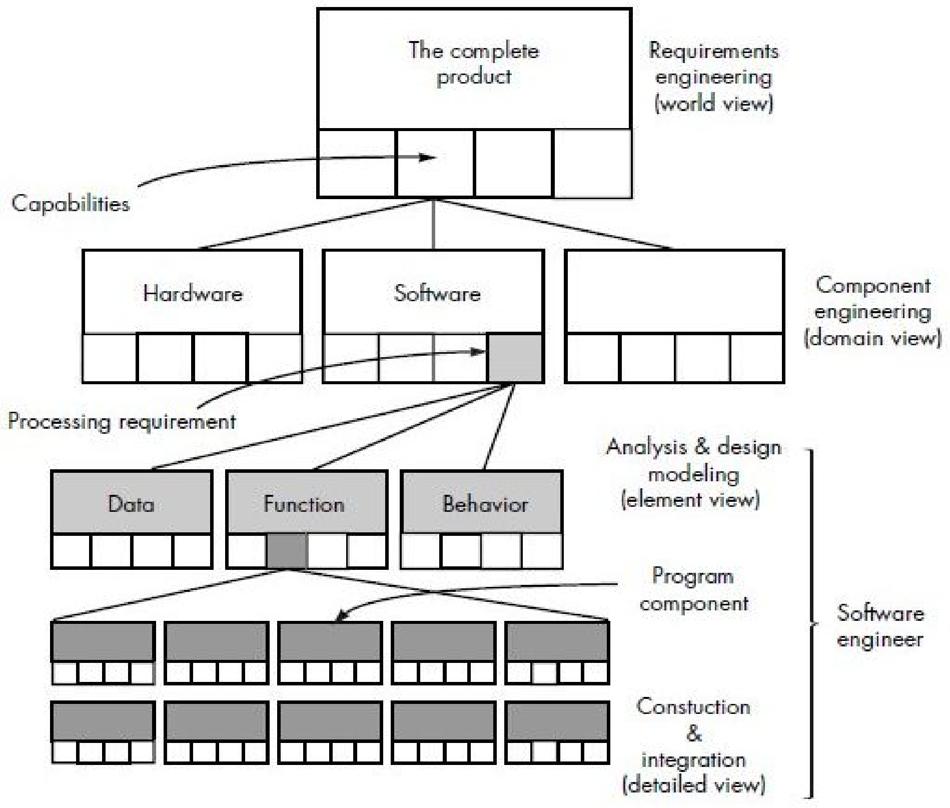

The enterprise
Information strategy planning (world view)
Business area
A business area
Business area analysis (domain view)
Processing requirement
Information system
Business system design (element view)
Software engineer
Construction & integration (detailed view)

- The final BPE step—*construction and integration* focuses on implementation detail. The architecture and infrastructure are implemented by constructing an appropriate database and internal data structures, by building applications using software components, and by selecting appropriate elements of a technology infrastructure to support the design created during BSD. Each of these system components must then be integrated to form a complete information system or application.
- The integration activity also places the new information system into the business area context, performing all user training and logistics support to achieve a smooth transition.

Product Engineering Overview

- The goal of product engineering is to translate the customer‘s desire for a set of defined capabilities into a working product. To achieve this goal, product engineering—like business process engineering—must derive architecture and infrastructure.
- The architecture encompasses four distinct system components: software, hardware, data (and databases), and people. A support infrastructure is established and includes the technology required to tie the components together and the information (e.g., documents,CD-ROM, video) that is used to support the components.
- The world view is achieved through requirements engineering. The overall requirements of the product are elicited from the customer. These requirements encompass information and control needs, product function and behavior, overall product performance, design and interfacing constraints, and other special needs.
- Once these requirements are known, the job of requirements engineering is to allocate function and behavior to each of the four components noted earlier. Once allocation has occurred, system component engineering commences.

- System component engineering is actually a set of concurrent activities that address each of the system components separately: software engineering, hardware engineering, human engineering, and database engineering.


The complete product
Requirements engineering (world view)
Capabilities
Hardware
Software
Component engineering (domain view)
Processing requirement
Analysis & design modeling (element view)
Data
Function
Behavior
Program component
Software engineer
Constuction & integration (detailed view)

- Each of these engineering disciplines takes a domain-specific view, but it is important to note that the engineering disciplines must establish and maintain active communication with one another. Part of the role of requirements engineering is to establish the interfacing mechanisms that will enable this to happen.
- The element view for product engineering is the engineering discipline itself applied to the allocated component. For software engineering, this means analysis and design modeling activities (covered in detail in later chapters) and construction and integration activities that encompass code generation, testing, and support steps.
- The analysis step models allocated requirements into representations of data, function, and behavior. Design maps the analysis model into data, architectural, interface, and soft ware component-level designs.