SOFTWARE PROCESS MODELS

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.

Generic software process models

- The waterfall model
  - Separate and distinct phases of specification and development.
- Evolutionary development
  - Specification, development and validation are interleaved.
- Component-based software engineering
  - The system is assembled from existing components.
- Formal systems development
  - A mathematical system model is formally transformed to an implementation
- Reuse-based development
  - The system is assembled from existing components

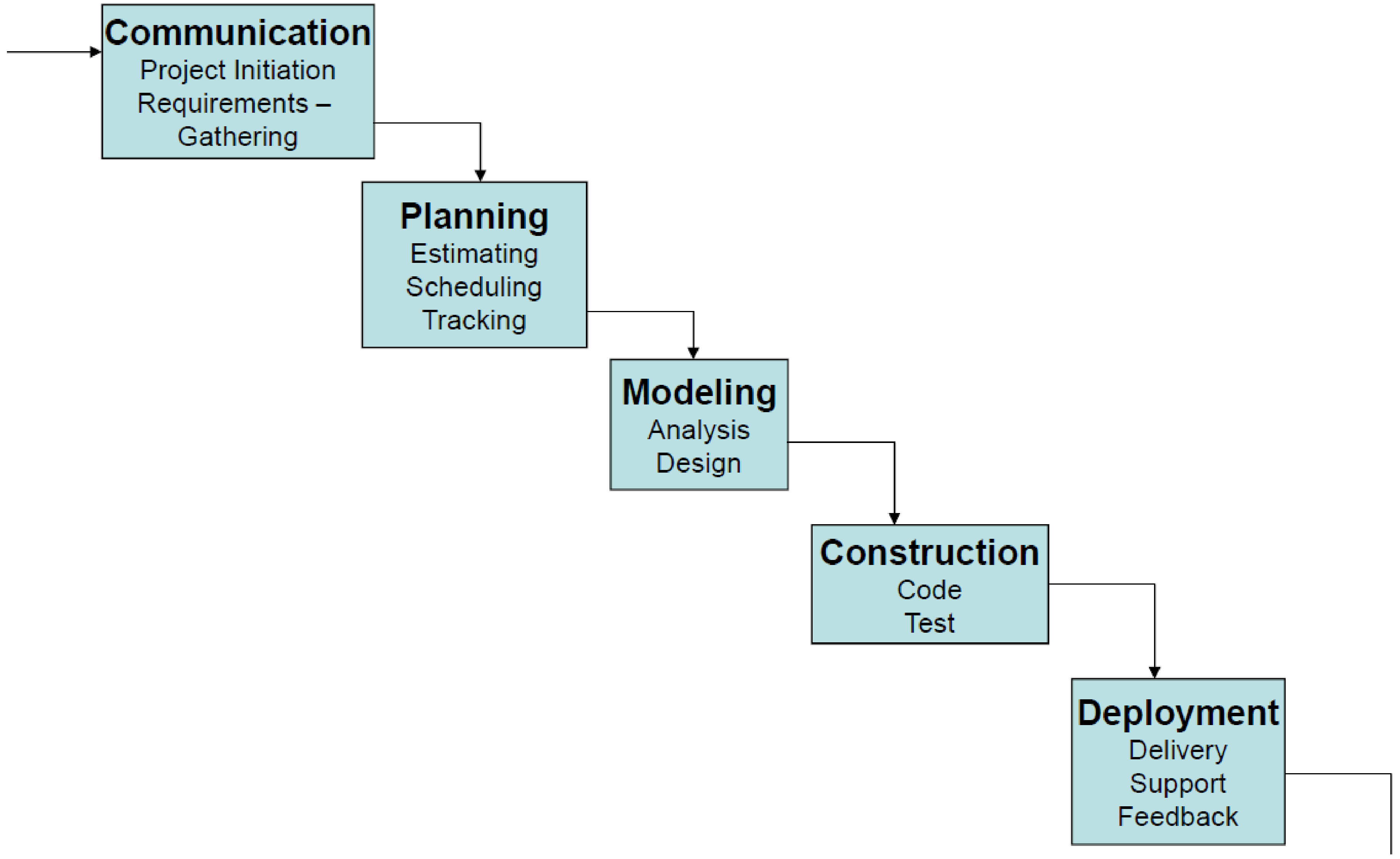
Waterfall model

- It is also called classical life cycle model.
- This situation is sometimes encountered when well defined adaptations or enhancements to an exciting system must be made.
- When this model is applied:-
  - Real projects rarely follow the sequential flow that the model proposes.
  - Indirectly, the linear model can accommodate iteration.
  - Often difficult for the customer to state all requirements explicitly.

Waterfall model (Contd….)

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

Waterfall Model (contd….)



Communication: Because s/w is a largest part of the system, work begins by establishing requirements for all system elements and then allocating some subset of these requirements to s/w. This system view is essential when s/w must interface with other elements such as hardware, people, and databases.

Analysis: The requirements gathering process is intensified and focused specially on s/w. To understand the nature of the programs to be built. The s/w engineer must understand the information domain for the s/w as well as required function, performance, and interfacing. Requirements both s/w and h/w are documented and reviewed with the customer.

Modeling: Design is concerned with identifying s/w components, h/w components and relationship between them. And specifying architecture….etc.

Design consists of

1. Architectural design
2. Detailed Design.

Architectural design involves identifying s/w components, decomposing and decoupling them into processing modules, conceptual data structures, and specifying the relationship between them.

Detail design is concerned with the details of **"how-to":**

- How to package the processing modules?
- How to implement the processing algorithms, data structures, and their connections among modules and data structures

Construction:

Code: The design must be translated into a machine readable form. If design is performed in a detailed manner, coding can be accomplished mechanistically.

Test: Once code has been generated, program testing begins. The testing process focuses on the logical internals of the s/w. And it ensuring that all statements have been tested on the functional externals.

Maintenance: S/w will undoubtedly undergo change after it is released to the customer. Change will occur because errors have been encountered, because the s/w must be adapted to accommodated changes in its external environment.

Advantages:

- It enforces discipline.
- Useful in project planning
- Shows phase containment errors.

Disadvantages:

- Inflexible partitioning of the project into distinct stages
- This makes it difficult to respond to changing customer requirements
- Therefore, this model is only appropriate when the requirements are well-understood

Disadvantages (contd…)

- Sequential approach to software development
- Oldest paradigm for software engineering.
- Inflexible partitioning of the project into distinct stages makes it difficult to respond to changing customer requirements.
- One phase has to be complete before moving onto the next phase.
- It 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.
- mostly used for large systems engineering projects where a system is developed at several sites.
- Difficulty accommodating the natural uncertainty that exists at the begging of many projects.
- Customers' must have patience.
- If undetected until the working program is reviewed, a major blunder can be disastrous.

Incremental Process Models

1. The Incremental Model

2. The RAD Model

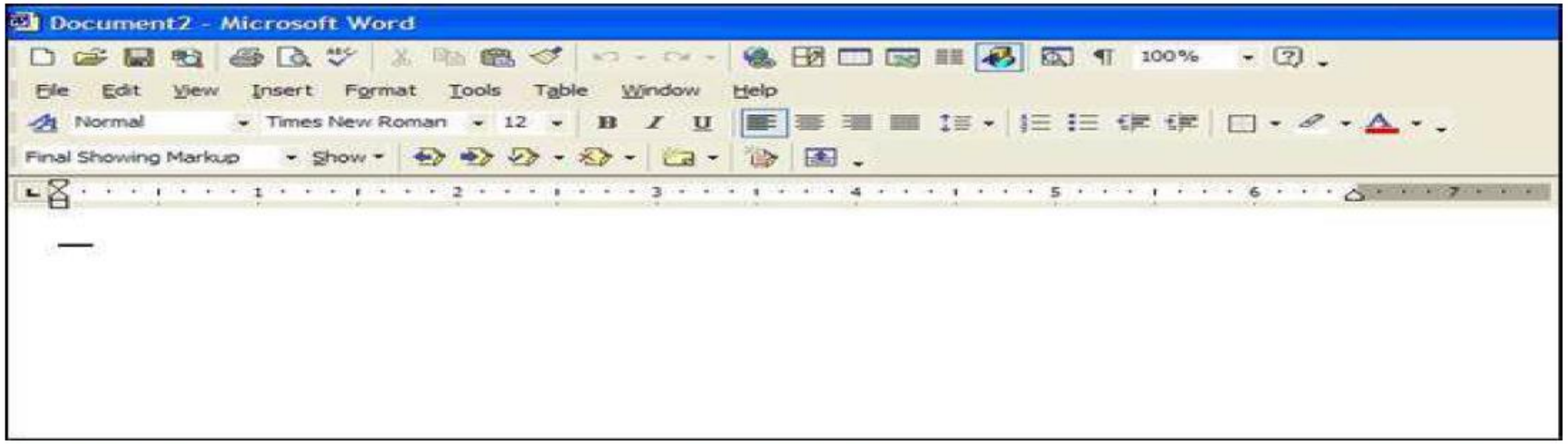
Process iteration

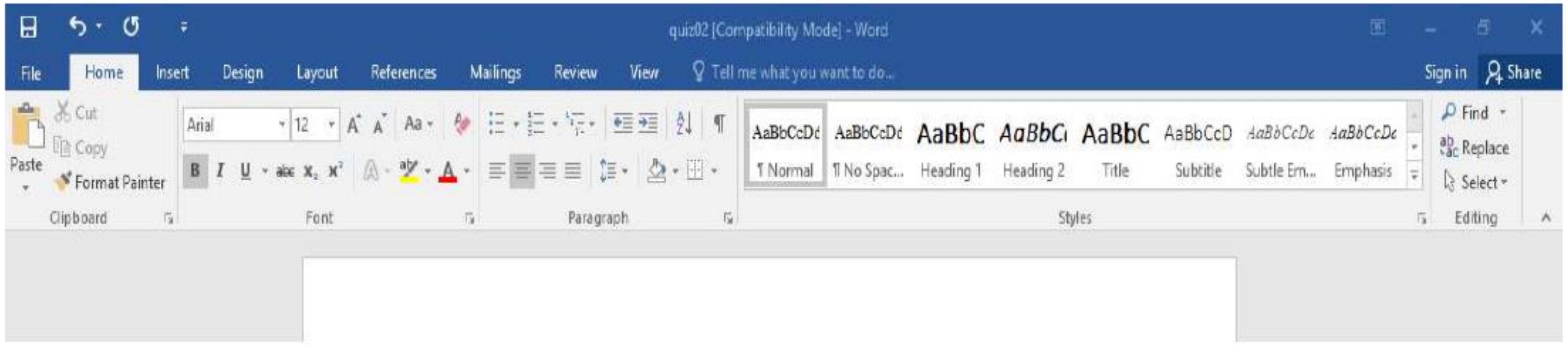
- System requirements ALWAYS evolve in the course of a project so process iteration where earlier stages are reworked is always part of the process for large systems.
- Iteration can be applied to any of the generic process models.
- Two (related) approaches
  - Incremental delivery;
  - Spiral development.

Incremental delivery

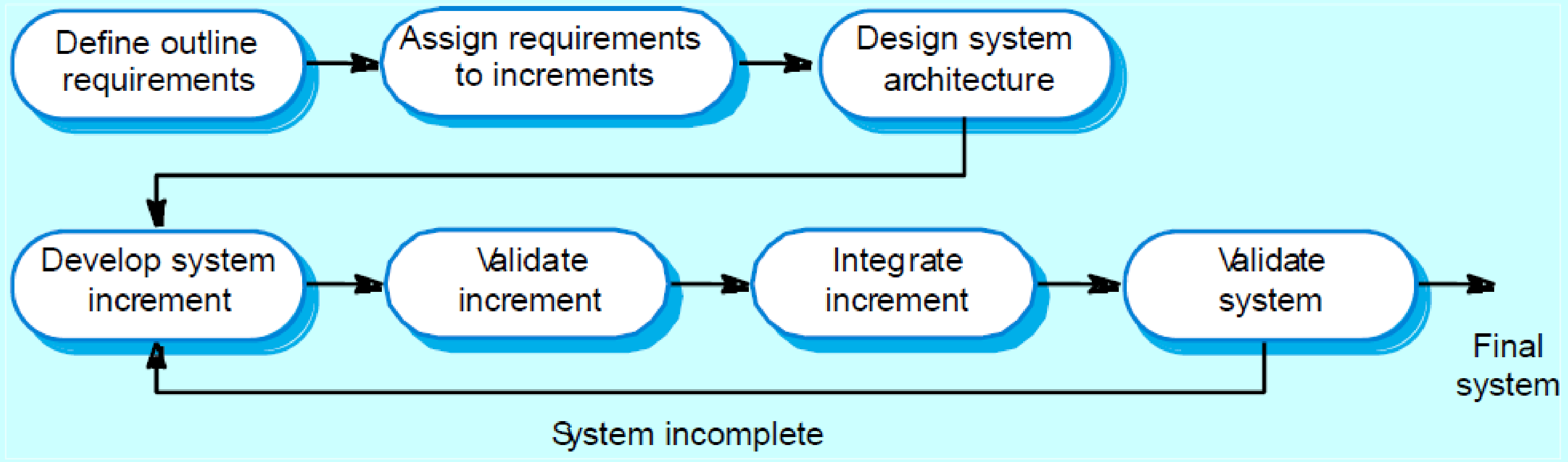
- This model combines elements of the waterfall model applied in an iterative fashion.
- This model applies linear sequences in a spread out fashion as calendar time progresses.
- **Example:-**
  - Word processing software developed using the incremental paradigm might deliver basic file management, editing & document production functions in the 1st increment;
  - more sophisticated editing & document production capabilities in the 2nd increment;
  - spelling & grammar checking in the 3rd increment &
  - advanced page layout capability in the 4th increment
- 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.

MS-WORD 97vs 2016

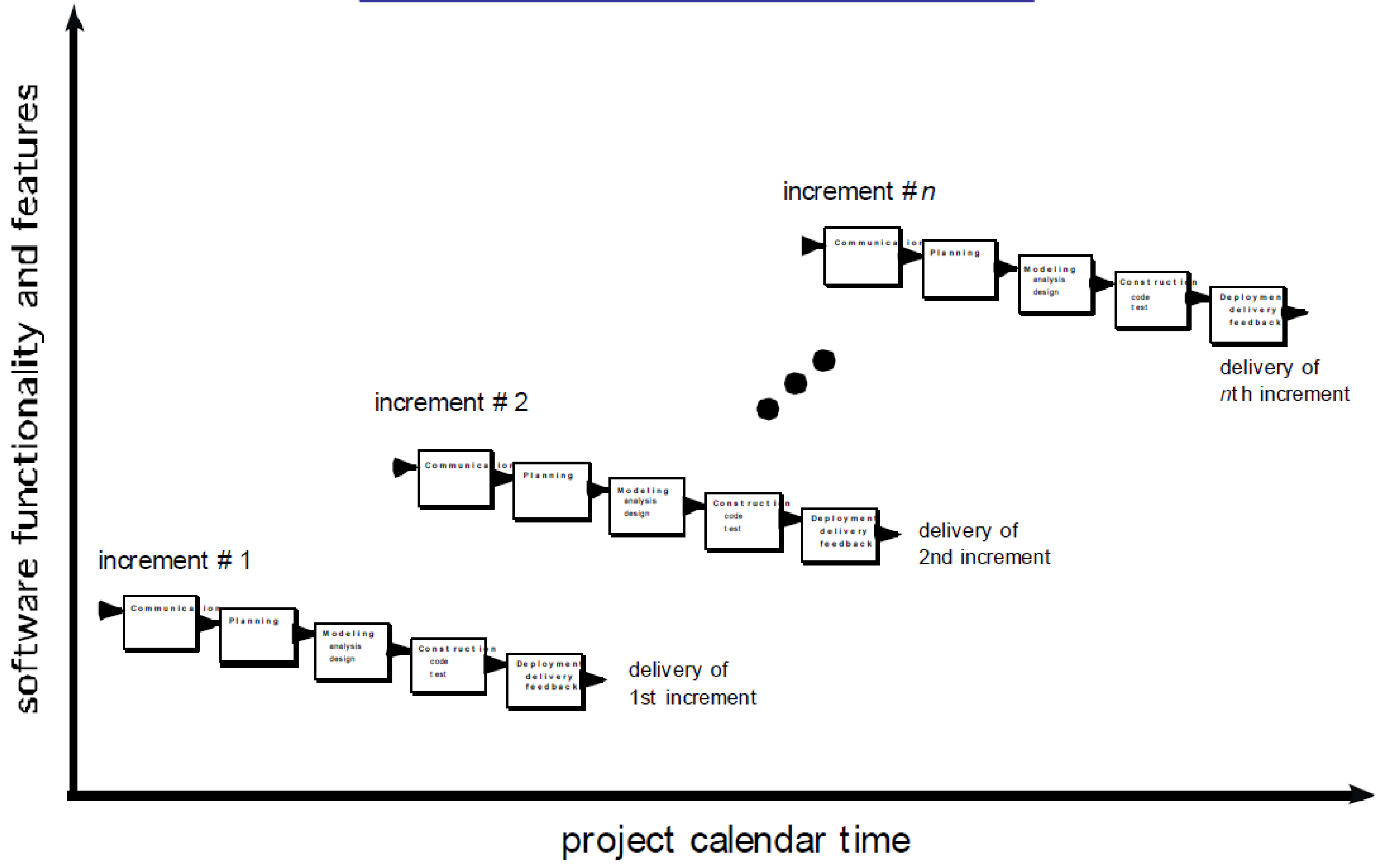

Document2 - Microsoft Word
File Edit View Insert Format Tools Table Window Help
Normal
Times New Roman
12
100%
Final Showing Markup
Show



Incremental development



Incremental Model



Incremental Model (Contd….)

- The 1st increment is often a **core product**.
- The **basic requirements** are addressed but many supplementary features (some known/unknown) remain undelivered.
- The core product is used by the customer.
- As a result of use/ evaluation, a plan is developed for the next increments.
- This process is repeated until the complete product is produced
- This model is particularly useful when staffing is unavailable for complete implementation by the business deadline has been established for the project.
- Increments can be planned to manage technical risks.

Advantages

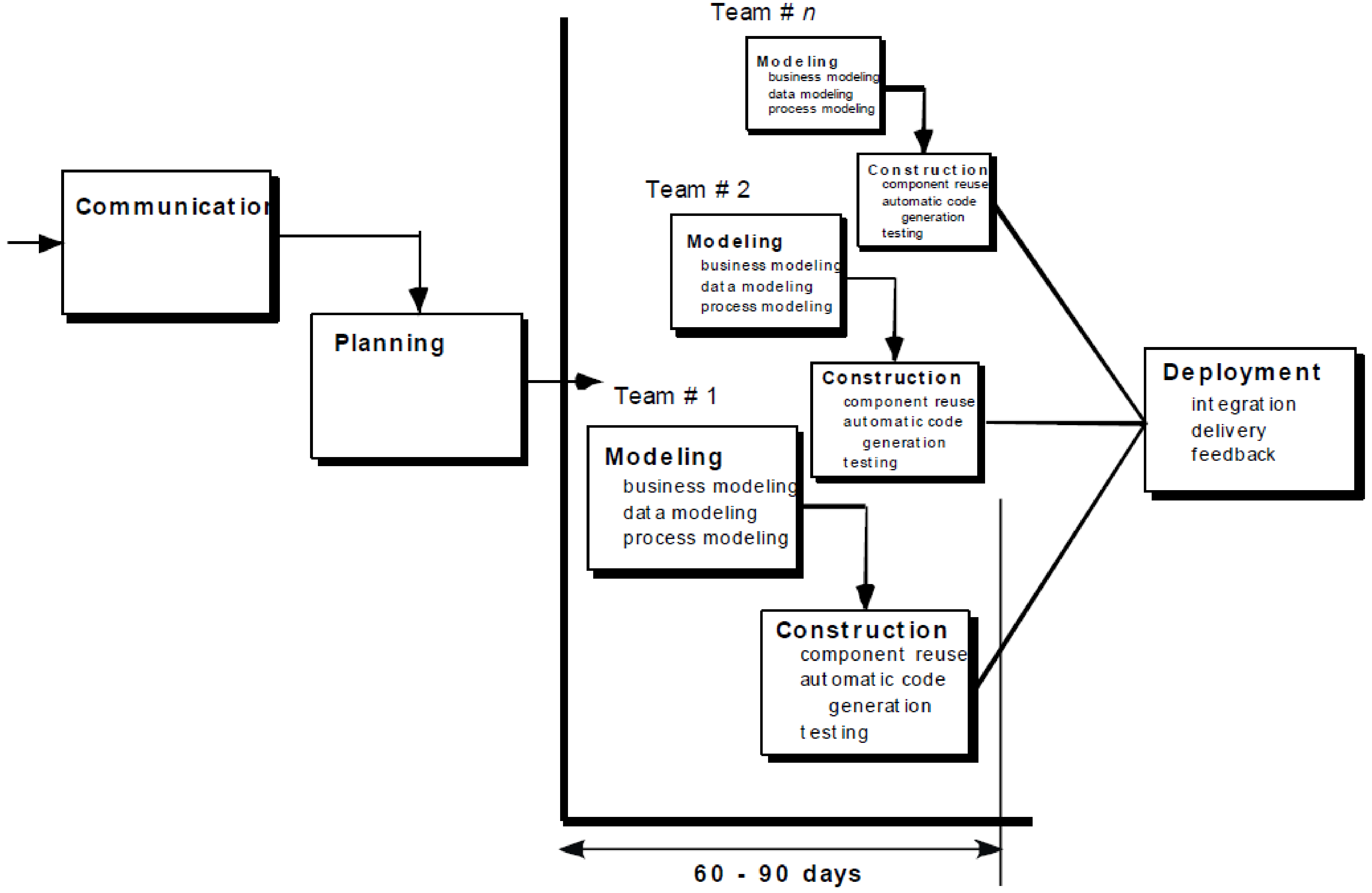
- Customer value can be delivered with each increment so system functionality is available earlier.
- Early increments act as a prototype to help draw requirements for later increments.
- Lower risk of overall project failure.
- The highest priority system services tend to receive the most testing.

The RAD Model

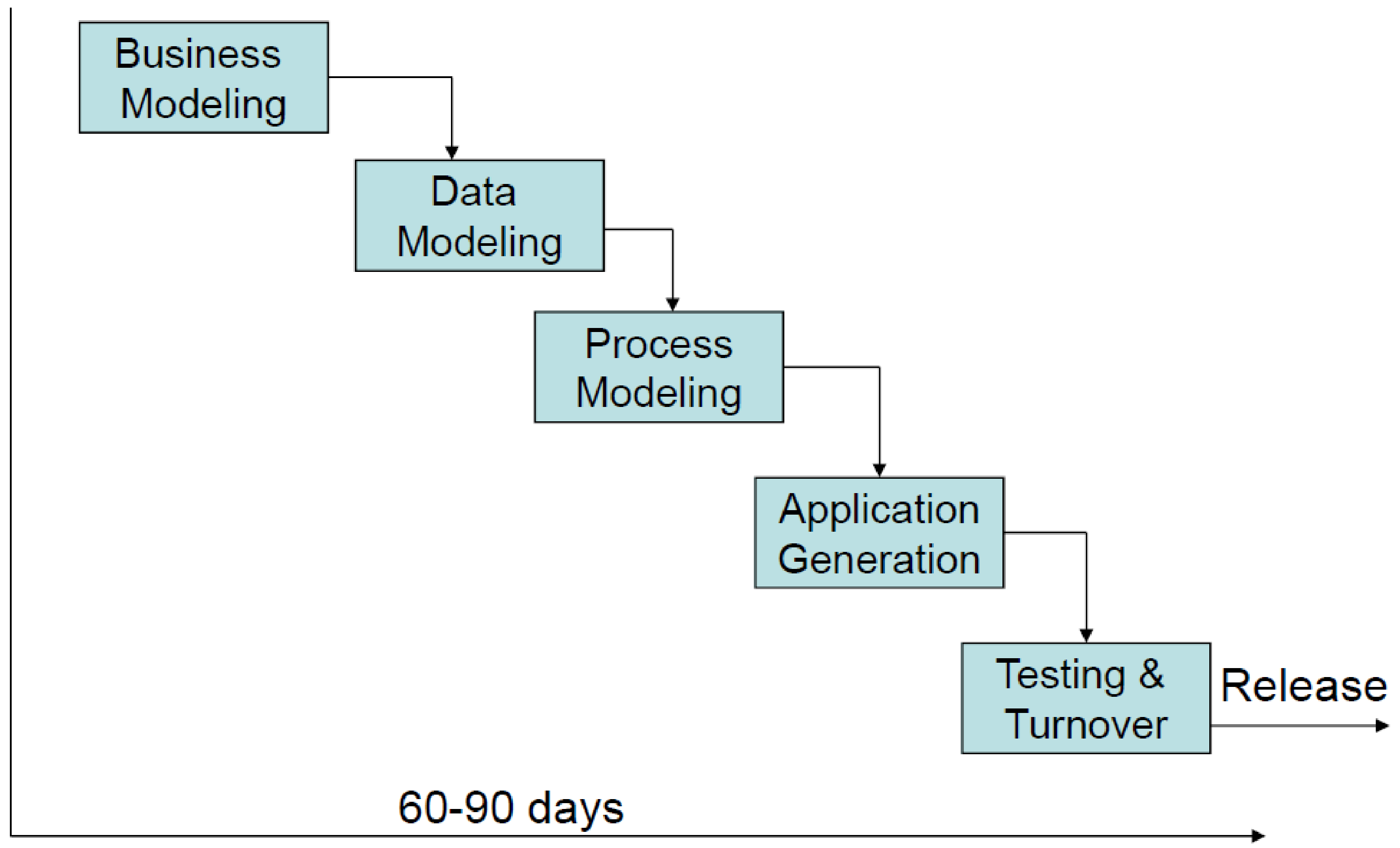
(Rapid Application Development)

- It is an incremental software process model that emphasizes a short development cycle.
- It is a **"high speed"** adaptation of the water fall model in which rapid development is achieved by using a component based construction.
- If requirements are well understood & project scope is meaningless constrained, it enables a development team to create a **"fully functional system"** within a very short time period (e.g. 60 to 90 days).

The RAD Model (contd….)



The RAD Model (contd….)



The RAD Model (contd….)

- **Business Modeling:**
  - It describes the information flow among business functions are modeled. Like
    - What information drives the business process?
    - What information is generated?
    - Who generates it?
    - Where does the information go?
    - Who process it?
- **Data Modeling:**
  - The information flow defined as a part of the business modeling phase is refined into a set of data objects are needed to support the business. The characteristics are of each object is identified and the relationships between these objects defined.

- **Process Modeling:**

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

- **Application Generation:**

  - RAD forces the use of **fourth generation techniques**, rather than creating a s/w using conventional third generation programming languages.
  - The RAD process works to reuse existing program components to create reusable components.
  - In all cases it uses automated tools to construct the s/w.

- **Testing & Turnover:**

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

Disadvantages

- For large, It requires sufficient human resources to create a right number of RAD teams.
- It fails, if customers & developers are not committed for rapid-fire-activities.
- It may problematic if not properly modularized.
- It may not work, if high performance is achieved through tuning interfaces to system components.
- It may not appropriate in case of more technical risks

Evolutionary Models

Types:

- **Prototyping Model**
- **Spiral Model**
- **Concurrent Development Model**

Spiral development

- It was originally proposed by Boehm.
- It has been developed to include the best features of both the sequential life model and prototyping models. These two models are missing risk analysis. The risk analysis is included in it. It provides the potential for rapid development of incremental versions of the software.
- Using the spiral model, software is developed in a series of incremental releases. During early iterations, the incremental release might be a paper model or prototype. During later iterations, increasingly more complete versions of the engineered system are produced.
- Process is represented as a spiral rather than as a sequence of activities with backtracking.
- Each loop in the spiral represents a phase in the process.
- No fixed phases such as specification or design - loops in the spiral are chosen depending on what is required.
- Risks are explicitly assessed and resolved throughout the process.

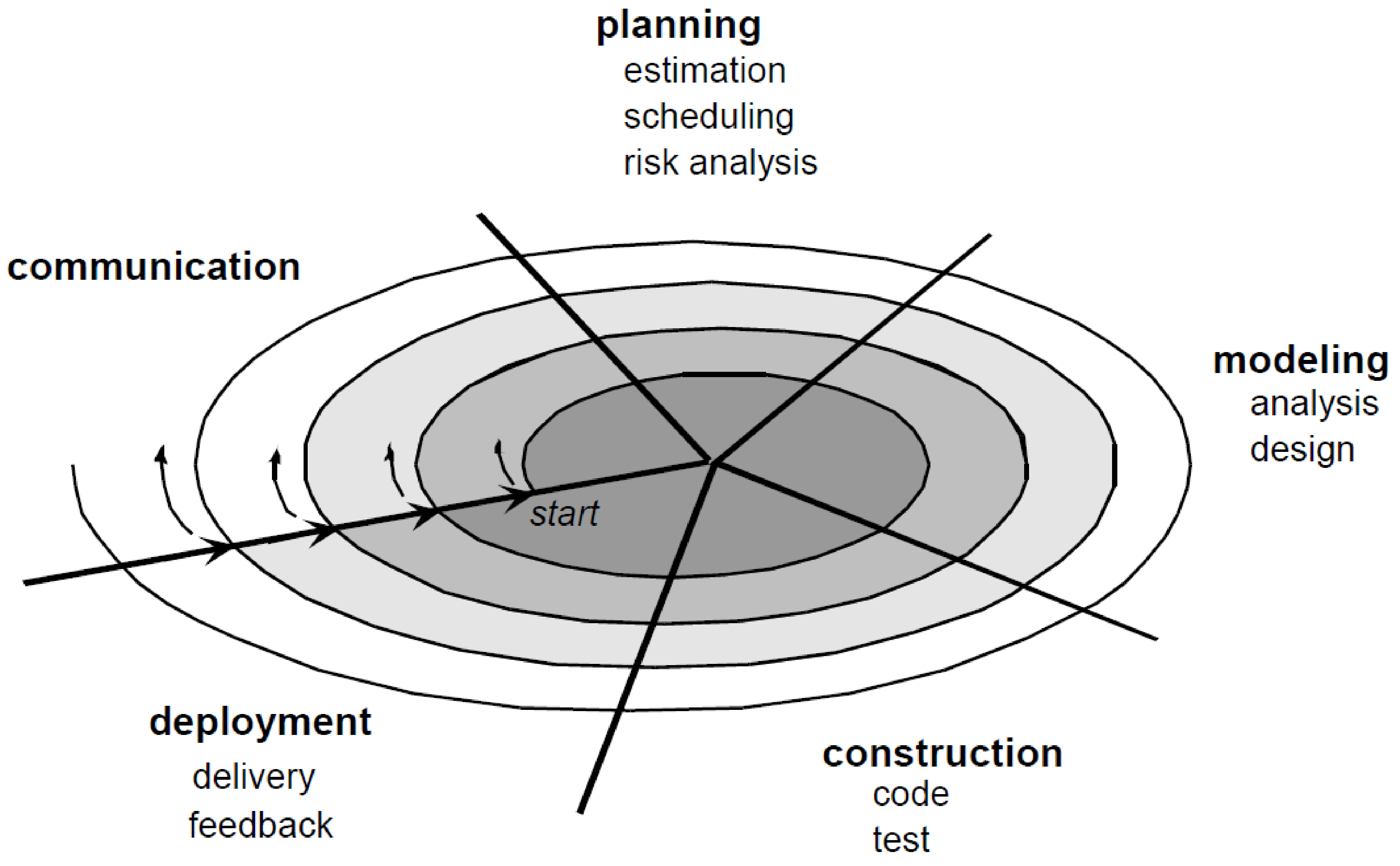
Spiral model (Meta Model)

- A spiral model is divided into a number of framework activities, also called ***task regions***. Typically, there are between three and six task regions. Figure shows a spiral model that contains 6 task regions:
- **Customer communication**— tasks required to establish effective communication between developer and customer.
- **Planning**— tasks required to define resources, timelines, and other project related information.
- **Risk analysis**— tasks required to assess both technical and management risks.
- **Engineering**— tasks required to build one or more representations of the application.
- **Construction and release**— tasks required to construct, test, install, and provide user support (e.g., documentation and training).
- **Customer evaluation**— tasks required to obtain customer feedback based on evaluation of the software representations created during the engineering stage and implemented during the installation stage.

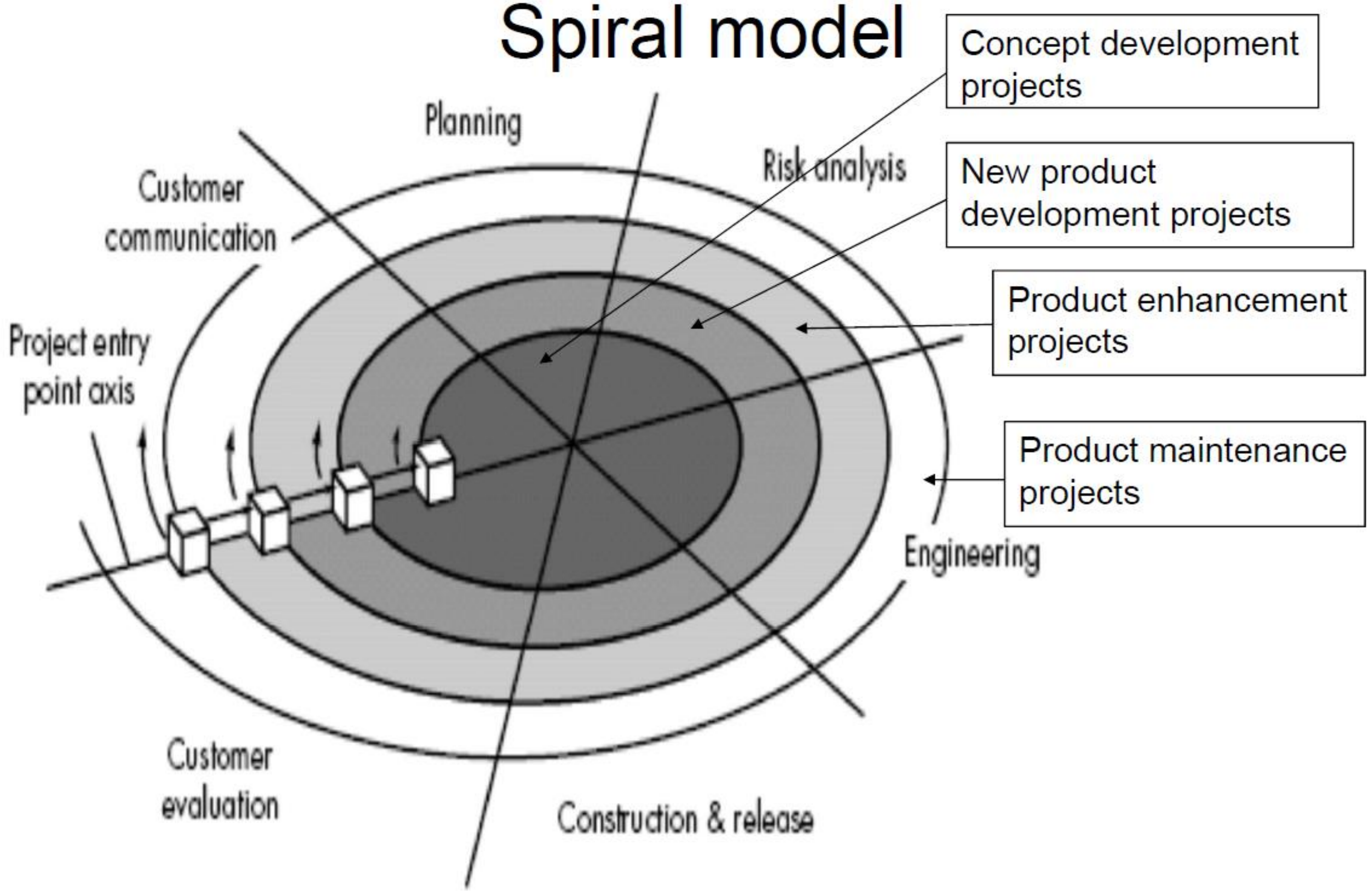
Spiral model (contd….)

- Objective setting
  - Specific objectives for the phase are identified.
- Risk assessment and reduction
  - Risks are assessed and activities put in place to reduce the key risks.
- Development and validation
  - A development model for the system is chosen which can be any of the generic models.
- Planning
  - The project is reviewed and the next phase of the spiral is planned.

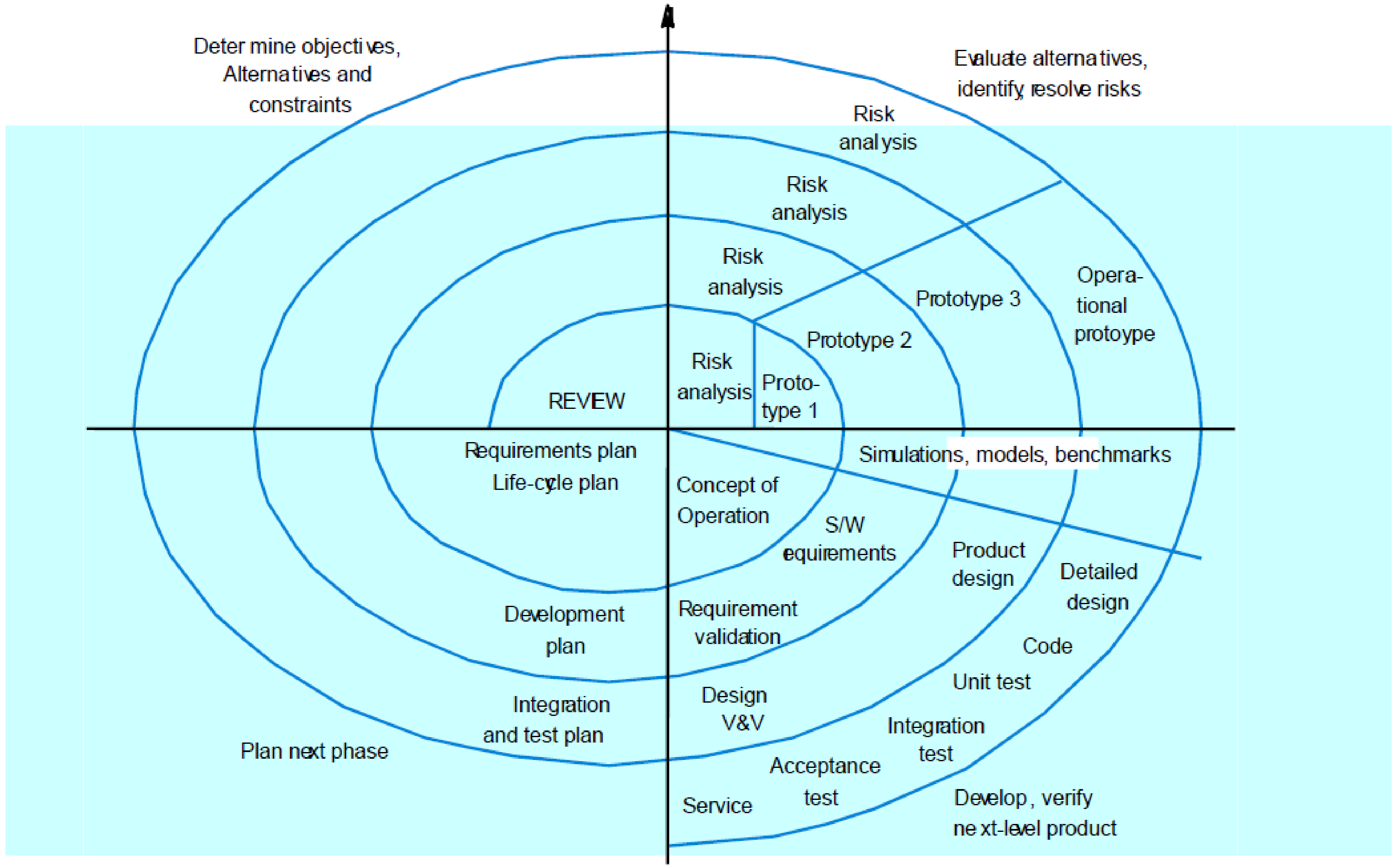
Spiral Model (contd….)



Spiral model



Spiral Model (contd….)



Advantages:

- Easy and quick to identify customer requirements
- Customers can validate the prototype at the earlier stage and provide their inputs and feedback
- It is a realistic approach to the development of large scale systems and s/w.
- Good to deal with the following cases:
  - Customer can not provide the detailed requirements
  - Very complicated system-user interactions
  - Use new technologies, hardware and algorithms
  - Develop new domain application systems

Disadvantages:

- It is not a solution.
- Difficult to convince customers.
- Demands considerable risk assessment expertise and depends on this expertise for success.
- If a major risk is not uncovered and managed, problems will undoubtedly occur.
- Finally the model has not been used as widely used as the waterfall , prototyping models.