Process models

Dr. Raju Bhukya

Assistant Professor

Dept of CSE, NITW

Introduction to software process

- Six phases in every Software development life cycle
- Requirement gathering and analysis
- Design
- Implementation or coding
- Testing
- Deployment
- Maintenance

- **Requirement gathering and analysis:** Meetings with managers, stake holders and users are held in order to determine the requirements like; Who is going to use the system? How will they use the system? What data should be input into the system? What data should be output by the system?
- **Design:** System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.
- **Implementation / Coding:** On receiving system design documents, the work is divided in modules/units and actual coding is started.
- **Testing:** After the code is developed it is tested against the requirements to make sure that the product is actually solving the needs.
- **Deployment:** After successful testing the product is delivered / deployed to the customer for their use.
- **Maintenance:** Once when the customers starts using the developed system then the actual problems comes up and needs to be solved from time to time.

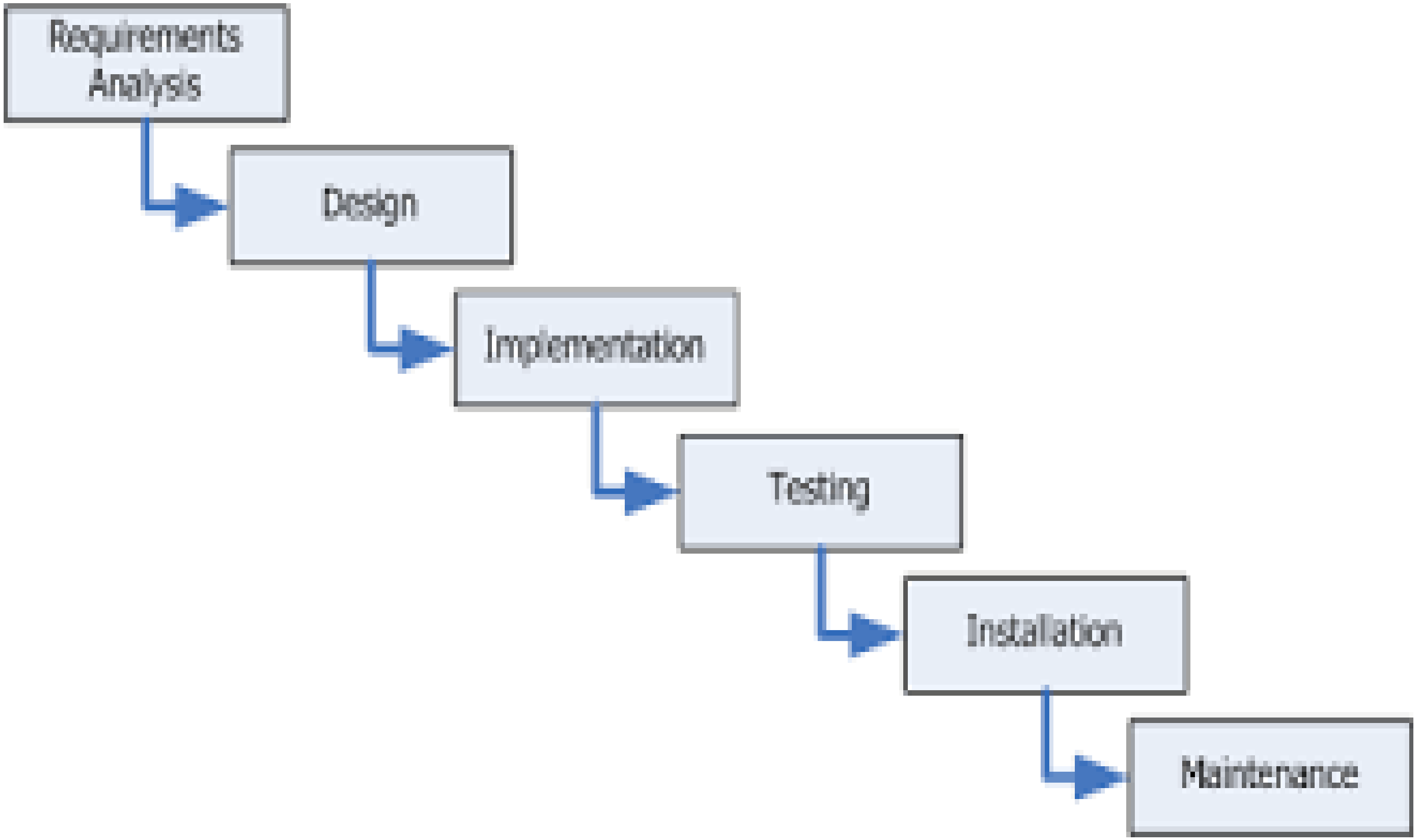
Process models

- Waterfall/sequential process model
- Prototyping model
- Incremental model
- Rapid application development
- Spiral process model
- Win-win process model
- V-Process model

Software process models

- Sequential model executes activities in sequence
- Iterative model repeats activates before preceding next.
- Evolutionary model flow executes activities in circular manner.
- Parallel process flow executes one or more activities in parallel.

Waterfall process model



Waterfall model

- Easy to understand
- Milestones well understood
- Good for the management control
- Used when requirements are well understood
- Used when the risk factor is low
- Work flow is in sequential faction
- Works well on mature projects and weak teams

When to use waterfall model

- Requirements are well known
- Product definition is stable
- Technology is well understood
- When the requirements are stable
- Changes in the requirements are limited

Difficulties in waterfall model

- All requirements must be known well in advance
- Integration may be a major problem
- Doesn't support iteration
- Changes can cause confusion
- Difficult to state all requirements explicitly
- Don’t expect accurate requirements early in project
- Difficult to integrate risk management

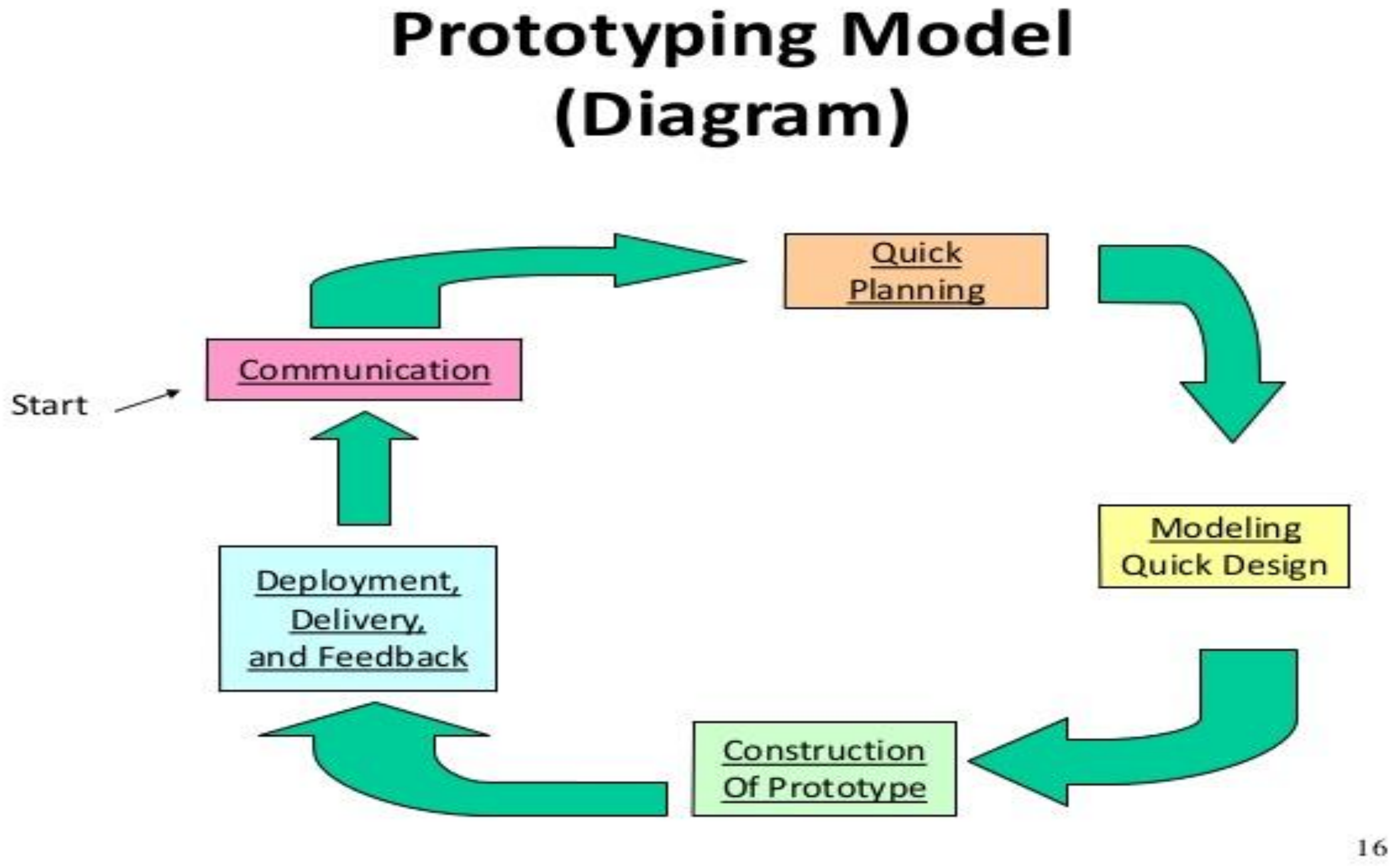
Contd..

- Difficult and expensive to make the changes
- Cost of requirements change is more
- Phase cannot start until previous completes
- No guidance to handle changes in the model
- Long wait before a final product
- Mostly used in large systems

Contd..

- Major drawback is difficulty of accommodating change
- More appropriate when requirements are well understood
- Long waiting before starting the next phase

Prototyping



Prototyping Model (Diagram)

16

Prototyping

- Prototyping enables us to explore the problem
- Enables us to explore the solution of the problem
- Enables us to communicate designs
- It enables us to classify the requirements
- Problem can be detected earlier
- Design is of higher quality
- Resulting system is easier to maintain

Prototyping

- Prototyping is an information-gathering technique
- Prototypes are useful in seeking user reactions, suggestions, innovations, and revision plans
- Prototyping may be used as an alternative to the systems development life cycle

Initial User Reactions

- Reactions must be gathered from users
- There are three types
  - User suggestions
  - Innovations
  - Revision plans

Prototyping As an Alternative to the Systems Life Cycle

- Two main problems with the SDLC
  - Extended time required to go through the development life cycle
  - User requirements change over time
  - Prototyping may be used as an alternative

Prototype Disadvantages

- Managing the prototyping process is difficult because of its rapid, iterative nature
- Requires feedback on the prototype
- Incomplete prototypes may be regarded as complete systems

Prototype Evaluation

- Analysts must work and evaluate users' reactions to the prototype
- Three ways the user is involved
  - Experimenting with the prototype
  - Giving open reactions to the prototype
    - Use a prototype evaluation form
  - Suggesting additions to and/or deletions from the prototype

Risk can be detected in early stage

- System is easier to use
- Problem is detected earlier
- Prototyping requires experienced engineers
- Users can provide feedback during the development
- Prototype can be produced in weeks
- Users becomes more positive about implementing the system

Detection of errors

- Prototyping enables early detection of the errors
- Used to give a concise impression of the system
- In rapid development prototyping is used.
- System is evaluated by end users & gives feedback
- Developers further refine's prototype

Changes are easy to handle

- Developer demonstrates prototype , user evaluates & gives improvement
- Cycle continues until both are satisfied
- Customers can see the requirements as they are gathered
- Developers can learn from the customers
- A more accurate end product
- Unexpected requirements are accommodated

Customers fell as actual system

- Allows flexible design and development
- Interactions with prototype stimulates awareness of additional information
- Rapid software development to validate requirements
- It is a rapid development of a system
- Helps the customers and developers to understand requirements of the system

Risk reduction activity

- Can be considered as risk reduction activity
- Prototyping reduces requirement risk
- Misunderstanding between developers and users exposed
- Missing and confusing services may be identified
- Working system is available early in the process

Quality

- Prototype improves design quality
- System usability can be improved
- Can reduce overall development effort
- Prototype may be screen scratch, cartoon screen, power point etc.,
- User can see, hold, interact more easily
- Prototyping is an information gathering technique
- Used in reactions, suggestions, innovations, revisions
- Prototyping may be used as alternative to the system development
- Reactions must be gathered from the users

When to use prototyping model

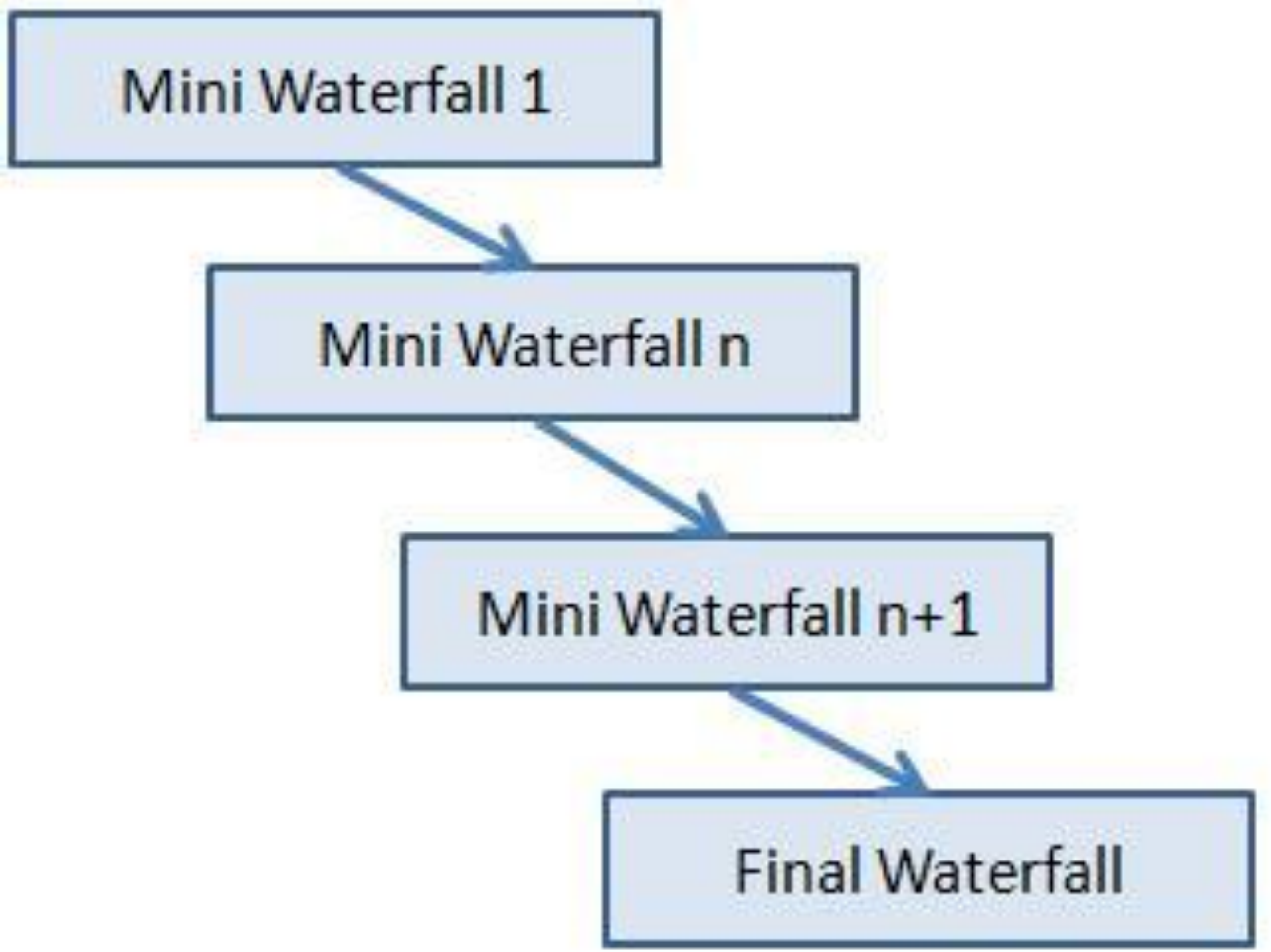
- When requirements are not clear
- When customer and developer are in confusion about the problem
- When the system is too complex
- When the requirements are not stable
- When there is no clarity in requirements
- When the risk is more
- When customer is changing the requirements

Advantages and disadvantages

- Advantages of prototyping
  - Reduced time and costs
  - Improved and increased user involvement

- Disadvantages of prototyping
  - Insufficient analysis
  - User confusion of prototype and finished system
  - Developer misunderstanding of user objectives
  - Developer attachment to prototype
  - Excessive development time of the prototype
  - Expense of implementing prototyping

Incremental process model



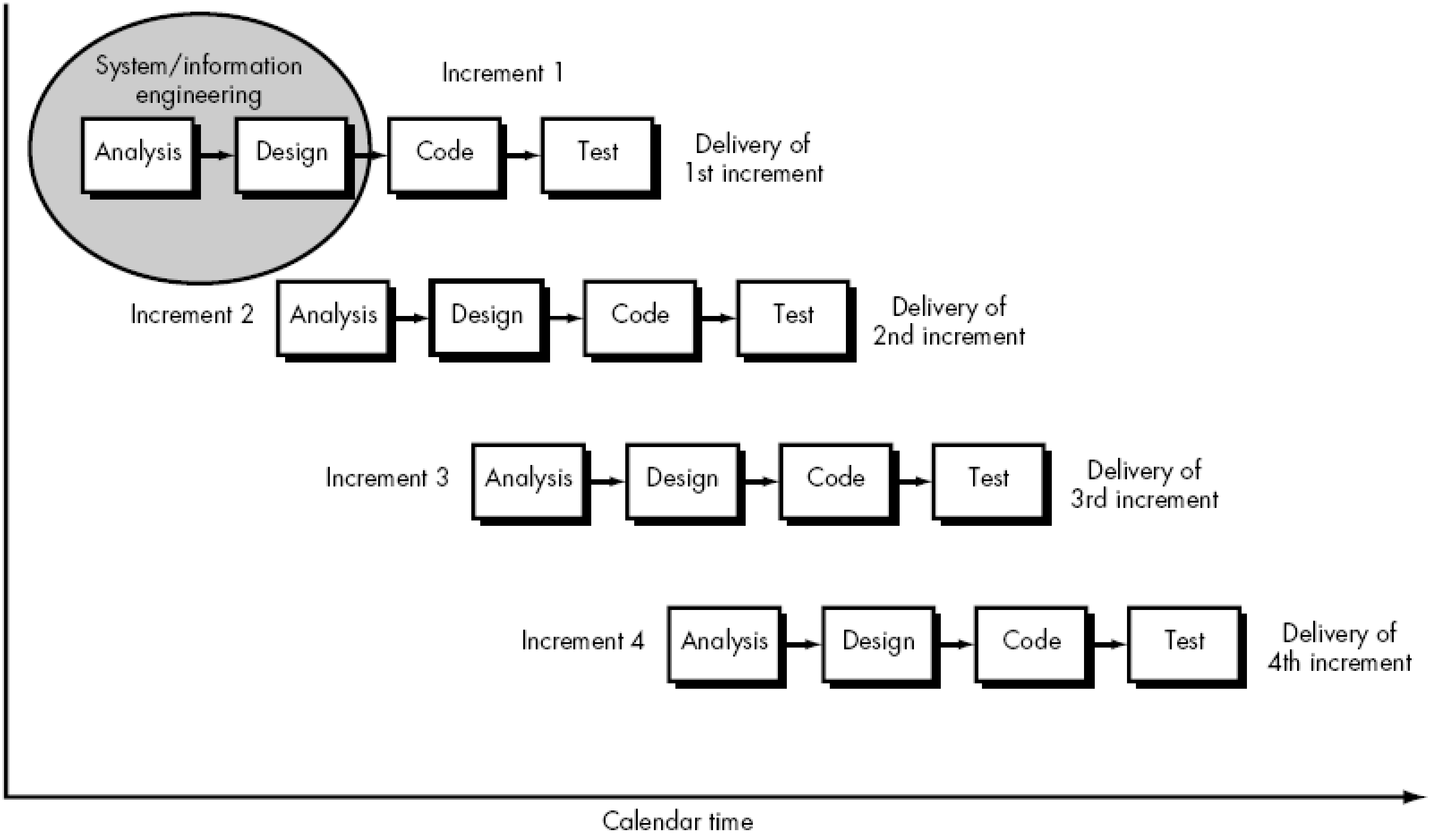
Mini Waterfall 1

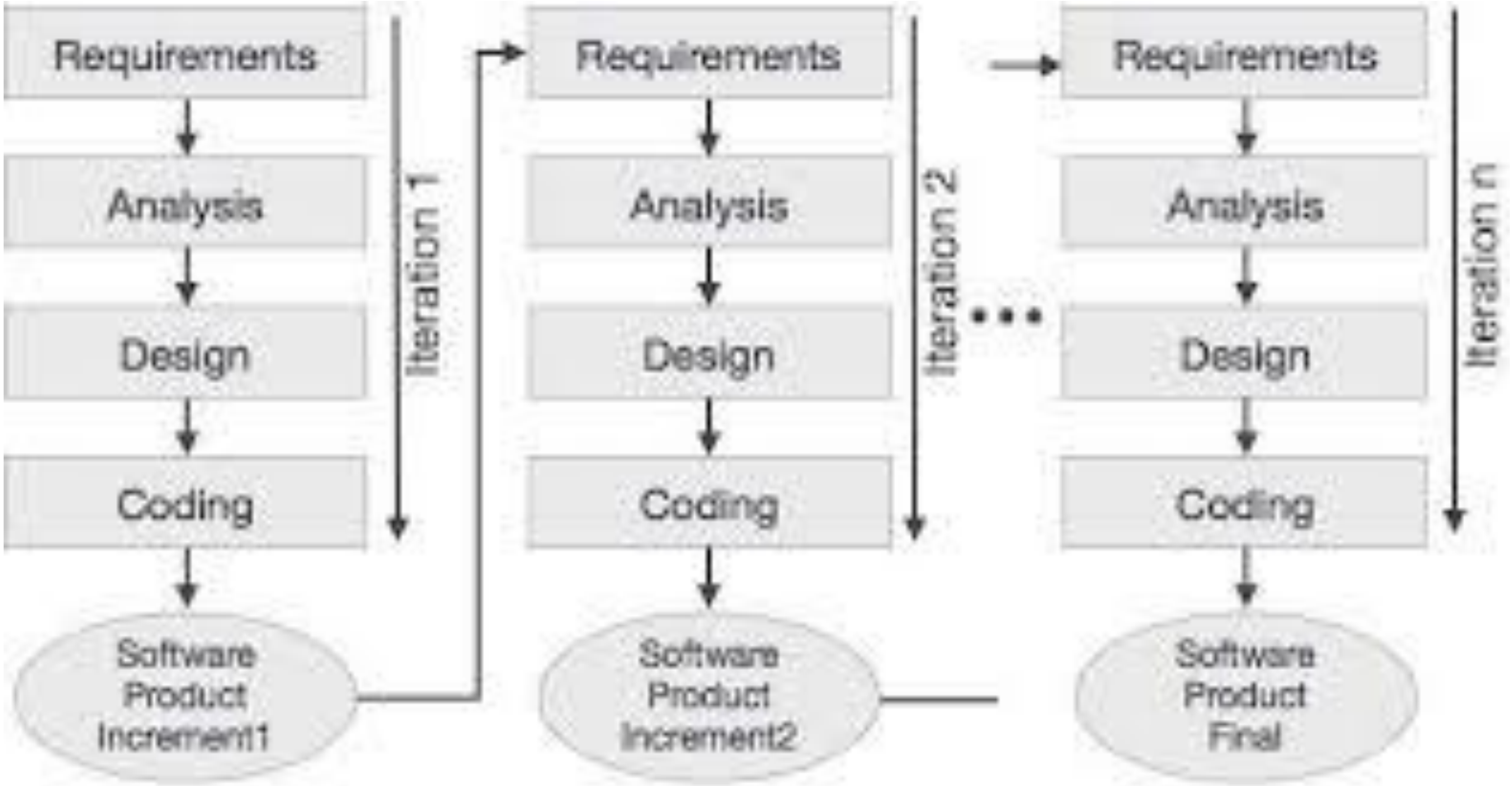
Mini Waterfall n

Mini Waterfall n+1

Final Waterfall

Incremental Process model




Requirements
Analysis
Design
Coding
Software Product Increment1
Iteration 1
Requirements
Analysis
Design
Coding
Software Product Increment2
Iteration 2
Requirements
Analysis
Design
Coding
Software Product Final
Iteration n

Incremental process model

- Partial implementation of the system
- Slowly add the increased functionality
- Incremental model gives priority to the requirements
- Each subsequent release system adds previous release
- Early release starts with small, subsystem, later adds the functionality

Contd..

- Delivers main system in initial release & adds up.
- Rather deliver as a single, system broken into increments
- User requirements are prioritized and delivered
- Used when multiple independent deliveries are identified
- Focus on operational product with each increment

Contd..

- Final product is built as separate prototypes
- At the end prototypes are merged into final product
- Consider whole project as set of mini projects
- Each mini project extends analysis, requirements, design, coding, testing
- Each iteration can be viewed as small but uses waterfall model

Contd..

- During each iteration selects portion of s/w product
- Particular increment performs waterfall model
- Divide the project into builds
- Operational product is done in weeks
- Smaller capital, rapid return on investment
- Required skilled developers
- User id closely involved in directing next steps

Contd..

- Allows the s/w to improve by giving enough time
- Allows system to run much sooner
- Delivery as a single, system broken into increments
- Final product is build as separate prototypes and prototype merged into final model
- Considers whole project as set of increments

Contd..

- Each mini project extends waterfall model phases
- Each iteration can be viewed as small but complete
- Operational quality within weeks
- Smaller capital, rapid return on investment
- Required skilled developers
- User is closely involved in directing next steps

Contd..

- Allows the software to improve by giving enough time
- Allows the system to complete much earlier
- Allows to identify risk at earlier stages
- Each iteration is a risk driven
- Result of single iteration is an increment
- First increment is often a core product
- Feedback is received from the customers and second increment is developed in parellel

Benefits of using process model

- s/w development team can easily understand the project they develop
- Process model helps to find inconsistencies, redundancies in process
- Each release delivers operational product
- Lowers initial delivery cost
- Initial product delivery is faster
- Customers get important functionality early

Lowers project failure rate

- Risk of changing requirements is reduced
- Product is build according to the priority of the customer
- Lower risk of project failure
- Highest priority system is delivered first
- Cost of accommodating changes in requirements is reduced
- Documentation and analysis is much less

Rapid Delivery

- Easier to get customer feedback
- More rapid delivery and development to customers is possible
- Customers can use software in less time
- Useful when staffing is too short
- Multiple opportunities for checking s/w process as correct
- Faults can be detected and corrected early
- Risks can be resolved early

Incremental model Strengths

- Develop major function first.
- Each release delivers an operational product.
- Customer can respond to each built.
- Uses divide & conquer break down strategy.
- Lower initial deliver cost.
- Initial product delivery is faster.
- Customer get important functionality early.
- Risk of changed requirements is reduced.
- Customer value can be delivered with each increment.

Delivery with priority

- Early increments acts like a prototype to help later increments.
- Lower risk of project failure.
- Highest priority system is delivered first.
- Cost of accommodating changing customer requirements reduced.
- Amount of analysis and documentation is much less.
- Easier to get customer feedback in development work.

Contd..

- Customers can comment on models implemented.
- More rapid delivery and development to customers is possible.
- Customers can use the software in less time.
- Useful when staffing is too short for full scale development .
- Multiple opportunities to check the software process as correct.
- Fault can detected and corrected early.

Contd..

- Risk can be resolved early.
- Working version of software is available early.
- Each increment is waterfall mini project.
- Combines the water fall with iterative of prototyping.

Incremental model weaknesses

- Requires good planning and design.
- Requires early definition of a complete system.
- Allow smallest definition of increments.
- Well defined module interfaces are required.
- Systems are poorly structured.
- Process is not visible.
- Managers needs regular deliverables to measure process.

Contd..

- System structure tends to degrade as new increments added.
- Regular changes tend to coccupt its structure.
- Further software changes becomes increasingly difficult and costly.

When to use incremental model

- Risk, finding, schedule, non availability of staff, early release.
- Initial requirements are known and expected to evolve over time.
- Need of basic functionality to the market early.
- Projects with lengthy development schedules.
- Projects with new technology.
- Working version s/w is available early.

Properties of process

- Easy to understand and communication
- Supports process improvement
- Supports process management

Software prototyping

- A prototype is an initial version of a system used to demonstrate concepts and try out design options.
- A prototype can be used in:
  - The requirements engineering process to help with requirements elicitation and validation;
  - In design processes to explore options and develop a UI design;
  - In the testing process to run back-to-back tests.

Rapid application development

- Incremental development process.
- Short development time(less than 3 Months)
- Full functional system can be developed in short time.
- Suitable for a system that can be modularized and each function is developed in less than 3 Months.
- Incremental model that emphasizes a short development cycle.
- Because of rapidly changing business needs.

RAD

- Is a “high-speed” adaptation of the linear sequential model in which rapid development is achieved.
- Incremental software development process model that emphasizes on short development cycle.
- Rapidly changing business environments
- Have to respond to new opportunities and competition.
- Requires rapid development
- Businesses may be willing to accept lower quality software if rapid delivery of essential functionality.

Requirements

- Changing environment, it is often impossible to arrive at a stable, consistent set of system requirements.
- Therefore a waterfall model of development is impractical
- RAD approach delivers within shot period of time.

User requirements are developed

- Solution is designed.
- Solution is prototyped.
- Prototype is reviewed
- Use improvement is provided.
- The process begins.

Characteristics of RAD processes

- The processes of specification, design and implementation are concurrent.
- The system is developed in a series of increments.
- End users evaluate each increment and make proposals for later increments.
- System user interfaces are usually developed using an interactive development system.

Contd..

- Business have to respond new opportunities & competition.
- This requires software with rapid development and delivery.
- Business may accept lower quality if rapid delivery is essential.
- Changing environment, it is impossible to achieve at stable constant set of system requirements.

Contd..

- Water fall is not possible, alternative is RAD.
- System is developed in a series of increments.
- Goals are faster, better and cheaper.
- When business objectives well defined and narrow
- Use of prototyping helps visualize and make adjustments.

Contd..

- Development of task structure encourages parallel project activities.
- RAD uses iterative prototyping.
- Designers review prototyping.
- Customers tries the prototype & evolve their requirements.
- Customers & developers review the product & refine requirements.
- Two to four weeks for requirements planning.

Contd..

- Prototypes of critical procedures are built.
- Plan for implementing system is prepared.
- Develops work with users, finalize design, build & test prototype.
- Fixed time seams within activities are done.
- Times is decided first accordingly activities are planned.
- Minimal planning and fast prototyping.
- Developing instead of planning.

Contd..

- Lack of pre-planning allows software to be written faster.
- Easier to change the requirements.
- RAD enables faster delivery with quality saving valuable resources.
- With conventional methods there is a long delay before customers gets product.
- Users and analysts meet to identify objectives of the application or system
- Oriented toward solving business problems

Key players in RAD

- Sponsors, requirements planning team,
- users, design team, review board training managers, project managers, construction team.

Key objectives of RAD are

- High speed, high quality & low cost.
- Active user involvement.
- System is tested and reviewed by developers & users.
- Ensures more customers satisfaction.
- End users joins the development of application.

Drawback of RAD

- Requires sufficient human resources.
- Need to create large enough number of RAD teams.
- Used for the large scale projects.
- Developers and customers committed to rapid fire activities.
- Doesn't suits when risk is high.
- Doesn’t suits when the system is not modularized properly.

Drawbacks of RAD Model

- RAD requires sufficient human resources to create the right number of RAD team, for a large project.
- RAD requires strong commitment from developers and customers in much abbreviated time frame.
- A system must be properly modularized.
- RAD is not appropriate when technical risks are high.

When to Use RAD

- RAD is used when
  - The team includes programmers and analysts who are experienced with it
  - There are pressing reasons for speeding up application development
  - The project involves a novel ecommerce application and needs quick results
  - Users are sophisticated and highly engaged with the goals of the company

When to use RAD

- Reasonably well known requirements.
- When users involved through life cycle.
- When project is in time framed.
- Functionality delivered in increments.
- High performance not required.
- Low technical risks.
- When system becomes modularized.
- The system to be completed in 2-3 Months.
- High availability of staffing and budget.

Disadvantages of RAD

- May try and hurry the project too much
- Loosely documented
- May not address pressing business problems
- Potentially steep learning curve for programmers inexperienced with RAD tools

Contd..

- Requires a systematic approach for modularization.
- Requires highly skilled and well trained developers.
- Doesn’t work for all projects.
- With conventional methods, there is a long delay before customers wants results.
- Conventional methods, software development can take too long &
- Customers requirements may change.
- Satisfying the customer at the time of delivery.
- Customers & developers must be committed to rapid fire activities in a short time frame.
- Not applicable for cheaper projects as modeling and code generation is high.

RAD strengths

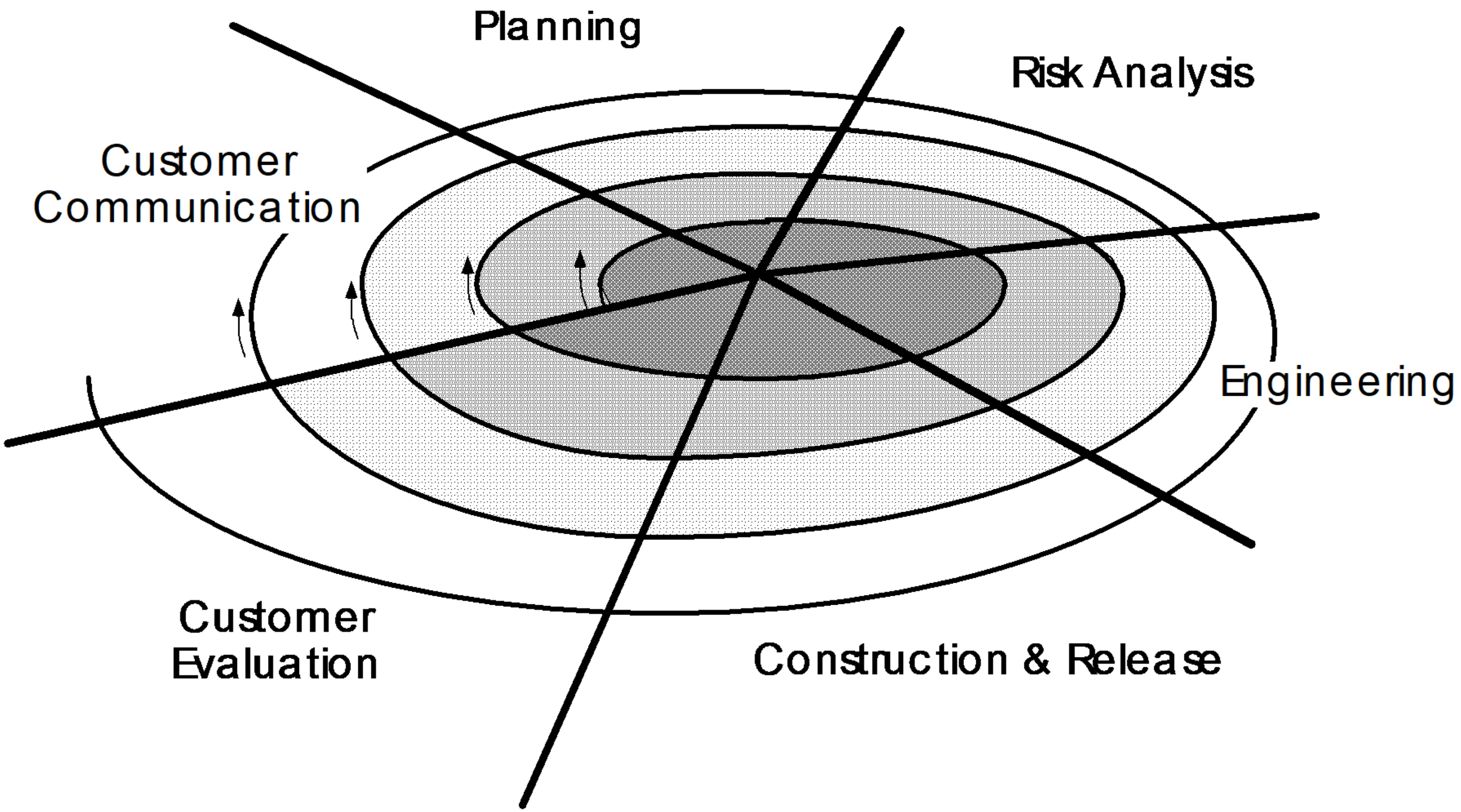
- Reduced cycle time & improved productivity.
- Time box approach over comes cost & schedule risks.
- Customers involved throughout the complete cycle.
- Reduced development time.
- Increases reusability of components.
- Quick initial review covers.
- Encourages customer feedback.
- Some times faster development with quality systems.
- Designed to take advantage of CASE tools.

Benefits of prototyping

- Improved system usability.
- A closer match to users' real needs.
- Improved design quality.
- Improved maintainability.
- Reduced development effort.

Advantages	Disadvantages
Dramatic time savings the systems development effort	More speed and lower cost may lead to lower overall system quality
Can save time, money and human effort	Danger of misalignment of system developed via RAD with the business due to missing information
Tighter fit between user requirements and system specifications	May have inconsistent internal designs within and across systems
Works especially well where speed of development is important	Possible violation of programming standards related to inconsistent naming conventions and inconsistent documentation
Ability to rapidly change system design as demanded by users	Difficulty with module reuse for future systems
System optimized for users involved in RAD process	Lack of scalability designed into system
Concentrates on essential system elements from user viewpoint	Lack of attention to later systems administration built into system
Strong user stake and ownership of system	High cost of commitment on the part of key user personnel

The Spiral Model



Spiral process model

- It combines the features of the prototyping model and the waterfall model.
- It is favored for large, expensive, and complicated models.

Steps in spiral model

- Define the problem with as much detail as possible
- Interviewing the client and potential users of the system, studying any existing system.
- A preliminary design is created for the new system.
- A first prototype of the new system is constructed from the preliminary design

A second prototype is derived by the following procedure

- Evaluate first prototype for strengths, weaknesses and risks
- Define the requirements of the 2nd prototype
- Plan and design the 2nd prototype
- Construct and test the 2nd prototype

WINWIN Spiral Model

Addresses more on customer communication, the following activities are defined:

1. Identification of the system or subsystem’s key “stakeholder”.
2. Determination of the stakeholders’ “win conditions”.
3. Negotiation of the stakeholders’ win conditions to reconcile them into a set of win-win conditions for all concerned.

Advantages and disadvantages

- Estimates of the budget and schedule become more realistic as work progresses
- Easier to cope with the changes inherent to software development
- Software engineers can start working on the project earlier.
- Estimates of budget and time are harder to judge at the beginning of the project
- Requirements evolve through the process

V process model

- ‘V-model’ is nothing but ‘Verification’ and ‘Validation’ model.
- In ‘V-model’ the developer’s life cycle and tester’s life cycle are mapped to each other.
- In this model testing is done side by side of the development.

V Process model

- Requirements life cycle model just like the waterfall model.
- In this model before development is started, a system test plan is created.
- Test plan focuses on meeting the functionality specified in the requirements gathering.
- High-level design focuses on system architecture and design.
- Integration test plan is created in this phase as well in order to test the pieces of the software systems.
- Low-level design phase creates actual software components.
- Defines the logic for each component of the system.

- Implementation phase is where coding takes place.
- Path of execution continues up the right side of the V as the test plans developed earlier is now put in use.
- Coding: This is at the bottom of the V-Shape model. Module design is converted into code by developers.

Advantages of V-model

- Simple and easy to use.
- Testing activities like planning, test designing happens well before coding. This saves a lot of time.
- Higher chance of success over the waterfall model.
- Defects found at early stage.
- Avoids the downward flow of the defects.
- Works well for small projects where requirements are easily understood.

Disadvantages of V-model

- Software is developed during the implementation phase, so no early prototypes of the software are produced.
- If any changes happen in midway, then the test documents along with requirement documents has to be updated.

When to use the V-model

- The V-shaped model should be used for small to medium sized projects where requirements are fixed.
- The V-Shaped model should be chosen when ample technical resources are available
- When technical expertise is available.
- High confidence of customer is required for choosing the V-Shaped model approach.
- No prototypes are produced.
- High risk involved in meeting customer expectations.

Advantages of Iterative model

- In iterative model we can only create a high-level design of the application before we actually begin to build the product and define the design solution for the entire product.
- In iterative model we are building and improving the product step by step. Hence we can track the defects at early stages. This avoids the downward flow of the defects.
- In iterative model we can get the reliable user feedback.
- In iterative model less time is spent on documenting and more time is given for designing.