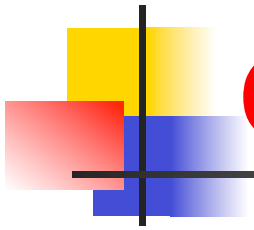
Systems Development Life Cycle Models

Waterfall & RUP Models

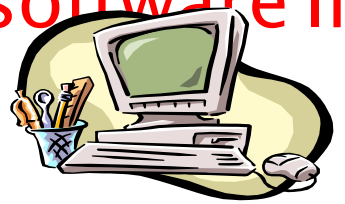


Objectives

- To know the features of the following systems development life cycle models:
  - Waterfall
  - Rational Unified Process
- To know the advantages and disadvantages of each model

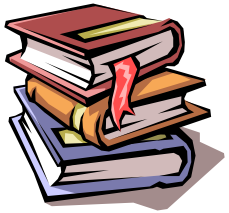
Life-Cycle of Software

- Software is like humans.
- It has a life cycle.
- Software in a system is conceptualized first.
- It becomes obsolescent at the end.
- The period in between is called the software life cycle

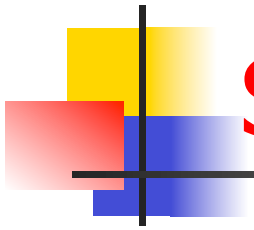


software

&

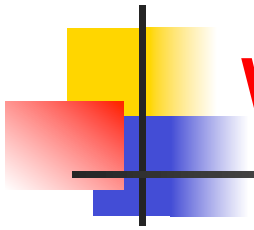


associated documentation



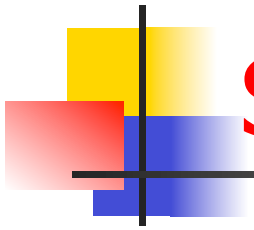
Software Development Models

- There are different software development life cycle models that propose different sequences of activities:
  - Waterfall Model
  - Incremental
  - Prototyping
  - Spiral
  - Rational Unified Process (RUP)
  - Etc.



Why use life cycle model?

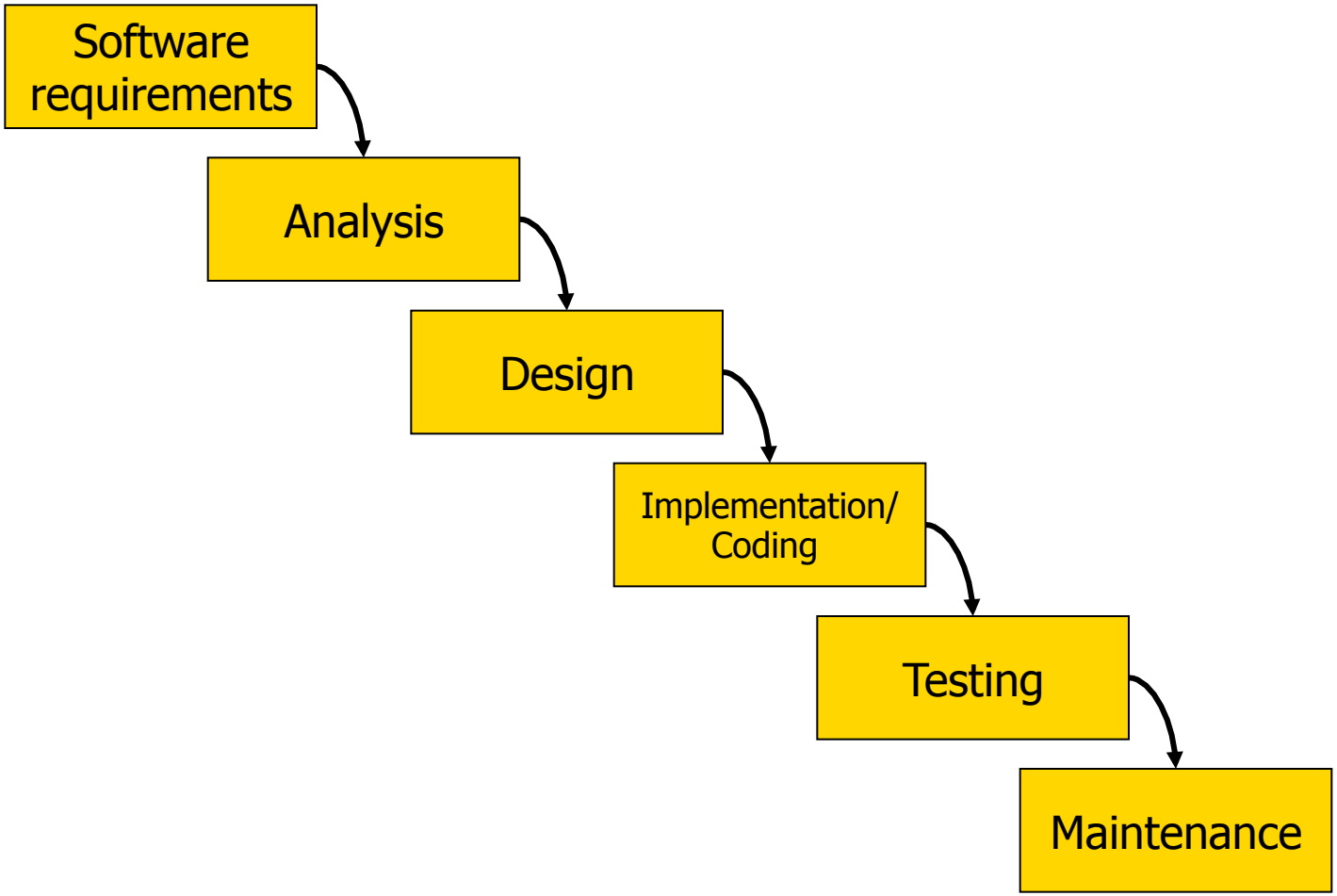
- Life cycle model breaks down the development process into phases or stages.
- This is because software development is complex.
- Breaking down the development process makes it easier to manage.
- Each phase can be performed in various ways.

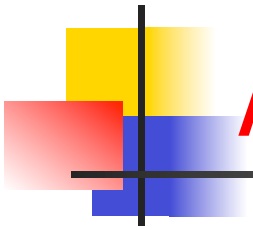


Software Development Models

- Waterfall Model
- Incremental
- Prototyping
- Spiral
- Rational Unified Process (RUP)
- Etc.

Waterfall Model – Phases

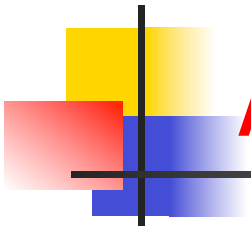




Analysis Phase

1. Requirement Analysis:

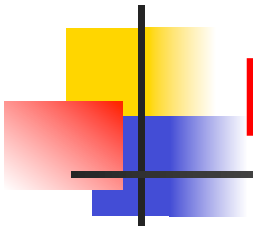
- Phase involves getting the 'user requirements' from the customer, i.e. what is the required purpose of the software.
- Requirements should be clearly defined before a solution can be formulated.
- Important to have a clear understanding of the problem to be solved.
- Should examine the user requirements statement carefully to determine what information is given and what information needs to be clarified.



Analysis Phase

2. Specification:

- Also referred to as *System Analysis*
- Involves detailing the required functions of the software.
- Problem must be clearly understood.
- Information needed to solved the problem (inputs) and the required outputs are identified, as well as the relationship between inputs and outputs.

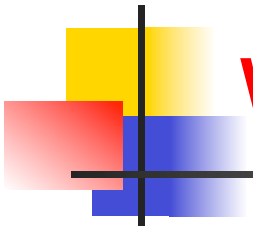


Design Phase

- Involves creating descriptions of a software system in terms of how it is to carry out the functions.
- A list of steps called an algorithm to meet the user requirements is developed and verified.

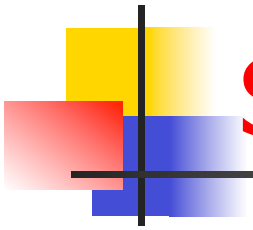
Implementation / Testing / Maintenance Phase

- **Implementation / Coding**
- Process of converting the design into the desired programming language.
- **Testing**
- Test the programs to see that they are correct and that they meet the user requirements
- **Maintenance**
- Occurs after the software is released
- Consists of modifications to software due to:
  - error reports
  - making modifications to the program to incorporate additional features (due to new or changed requirements)



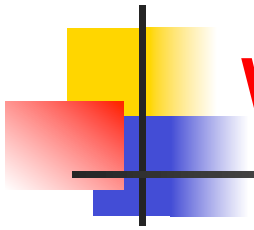
Waterfall Model – Characteristics

- Place considerable emphasis on a careful analysis before the system is actually built.
- Try to identify and tie down the user's requirements as early as possible
- Transition from one activity to the next typically requires:
  - completion of a defined work product
  - formal evaluation and acceptance of the work product
- Tends to be a document-driven process.



Strengths of Waterfall

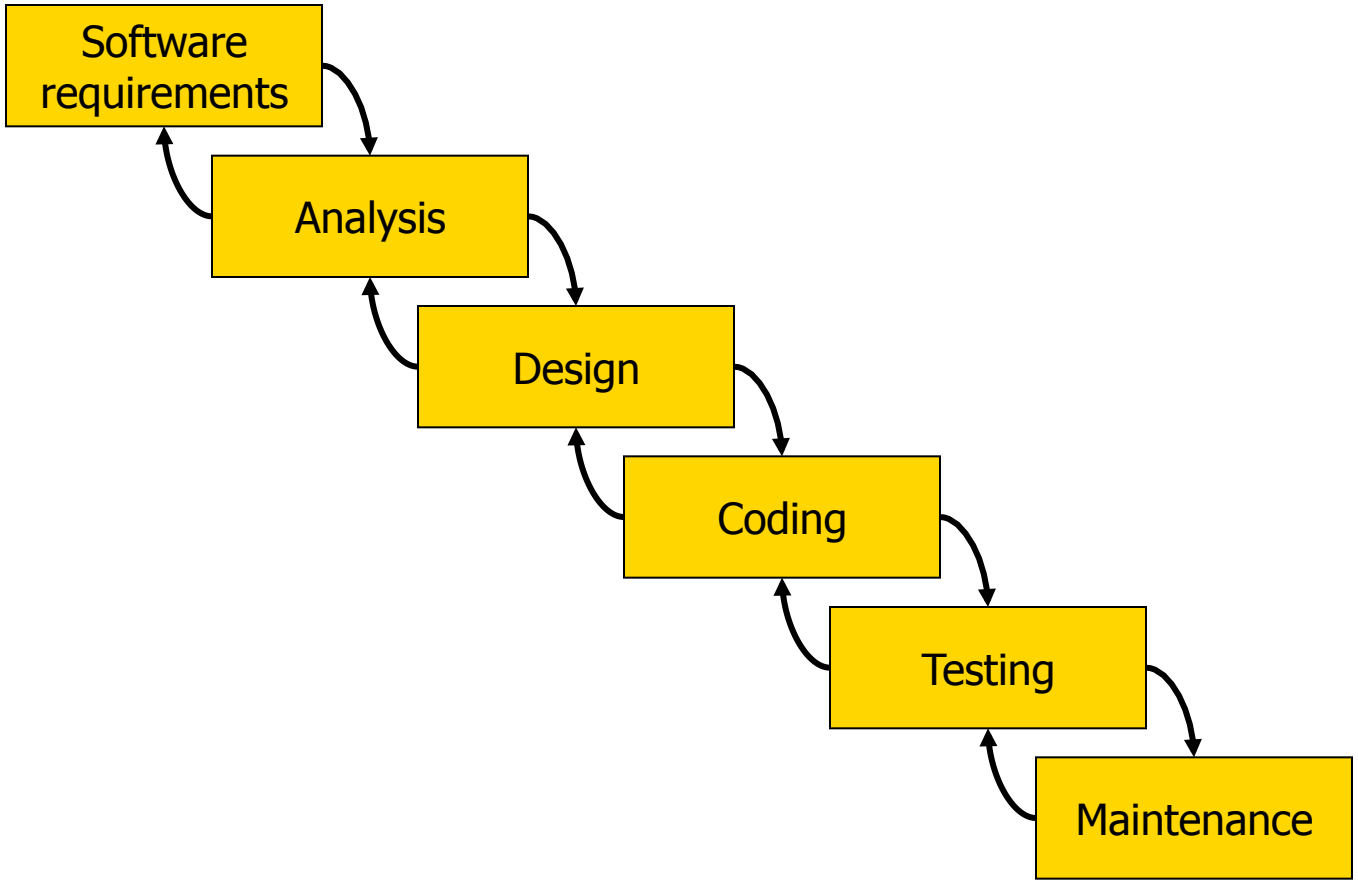
- Encourages periodic review, validation and verification, results in higher performance product, more closely matches the requirements
- Each phase results in document, helps clarify decisions, provides an audit trail, serves as concrete milestone.
- Formal transition from phase to phase results in a progressive "setting" of the product; reduces unnecessary changes.
- Because of the above items, this model is appealing to
  - **project managers** since it **provides them a "sense of control",** easy flow to understand;
  - to **contractors** since often they **get paid for document delivery** (low risk to make considerable money without building the actual product)



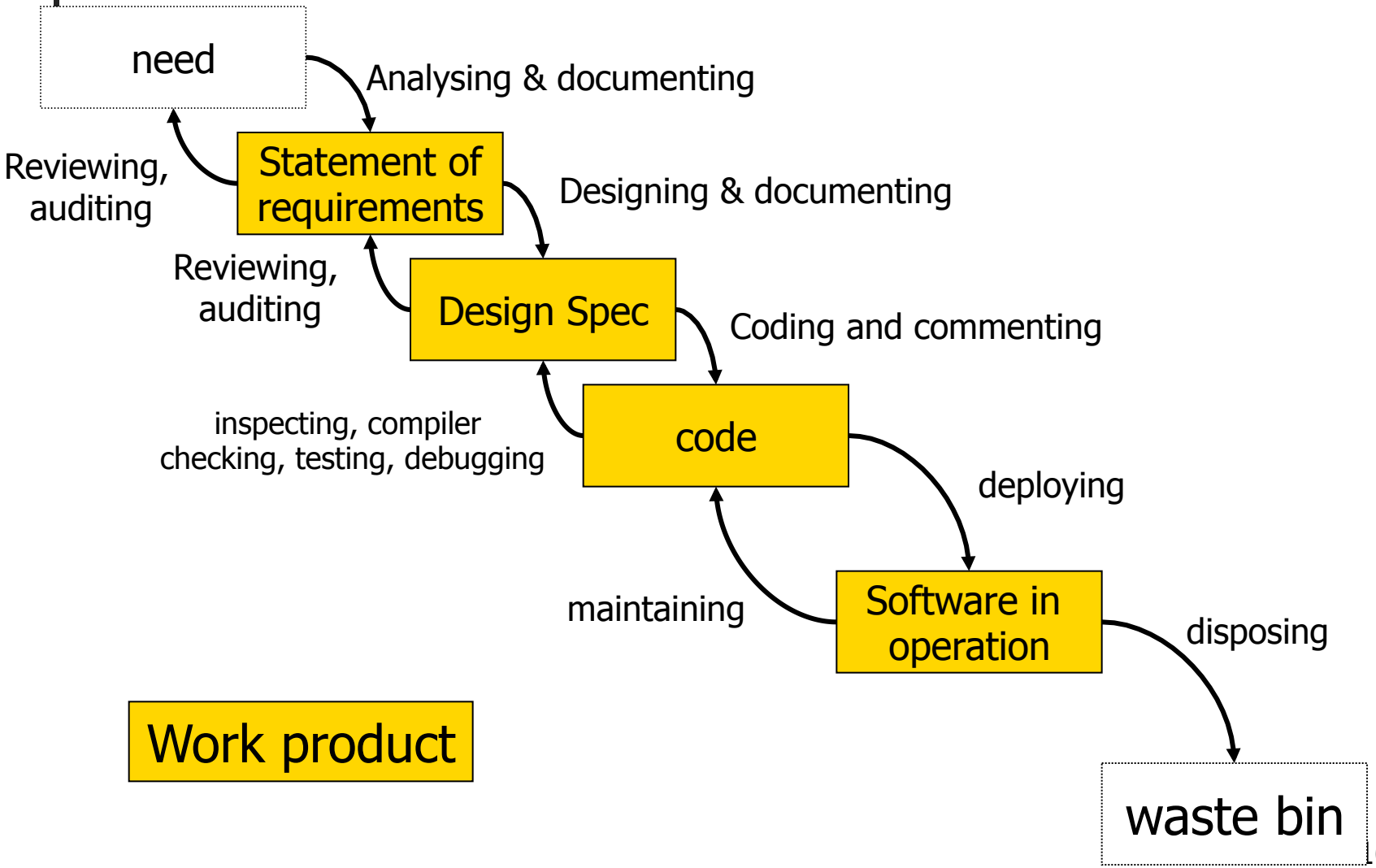
Weaknesses of Waterfall

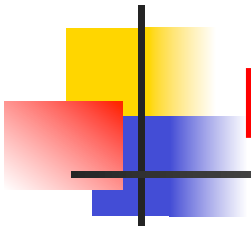
- inherent assumption: possible to get requirements, design complete and correct on first pass
  - true for some projects
  - for most projects, difficult to correctly state or visualize requirements before system designed or implemented
  - first designs almost always sub-optimal
- when necessary to revisit completed phases, normally large administrative overhead, inertia
  - reduces ability to make course corrections when early decisions turn out to be inappropriate

Modified Waterfall Model



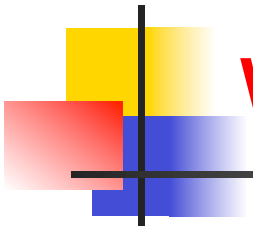
Revised Waterfall Model





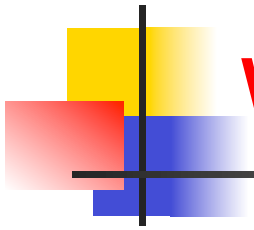
Modified Waterfall – Characteristics

- Work product flows down the primary, stepwise path of normal development.
- Reverse flow represents iterative changes applied to a prior deliverable
  - Need for change has been only recognized in the next phase or even later phase
- Iterations implies rework
  - Means work from prior phases is to be repeated (not efficient).



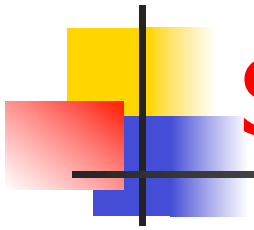
Waterfall, does it work?

- For very small non-complex projects the waterfall is sometimes adequate.
  - Easy to get all the requirements correct the first time
- For large production projects, may also be adequate.
  - Knowledge acquired during previous similar projects allows us to properly gather and identify all the user and design requirements, and where little new development is required.
- For large development projects, it is NOT adequate.
  - Almost impossible to properly identify all the user and design requirements; many changes to the product are then required.



Waterfall : Conclusion

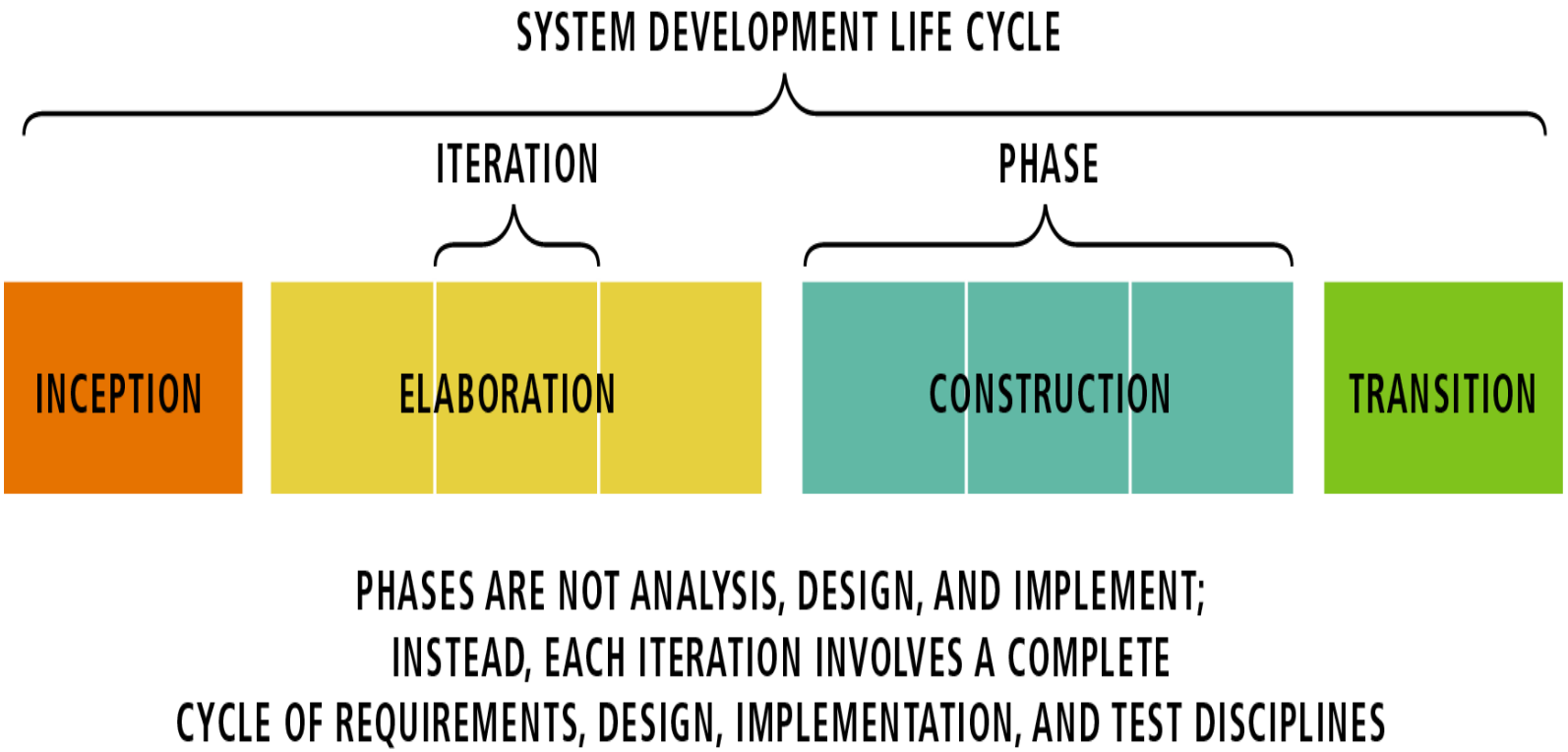
- First formally defined
- First widely accepted
- Still the most widely used
- Simple
- Flexible : many variants overcome weaknesses (prototyping inside requirements phase, feedback loops allow iteration)



Software Development Models

- Waterfall Model
- Incremental
- Prototyping
- Spiral
- Rational Unified Process (RUP)

Rational Unified Process (RUP)

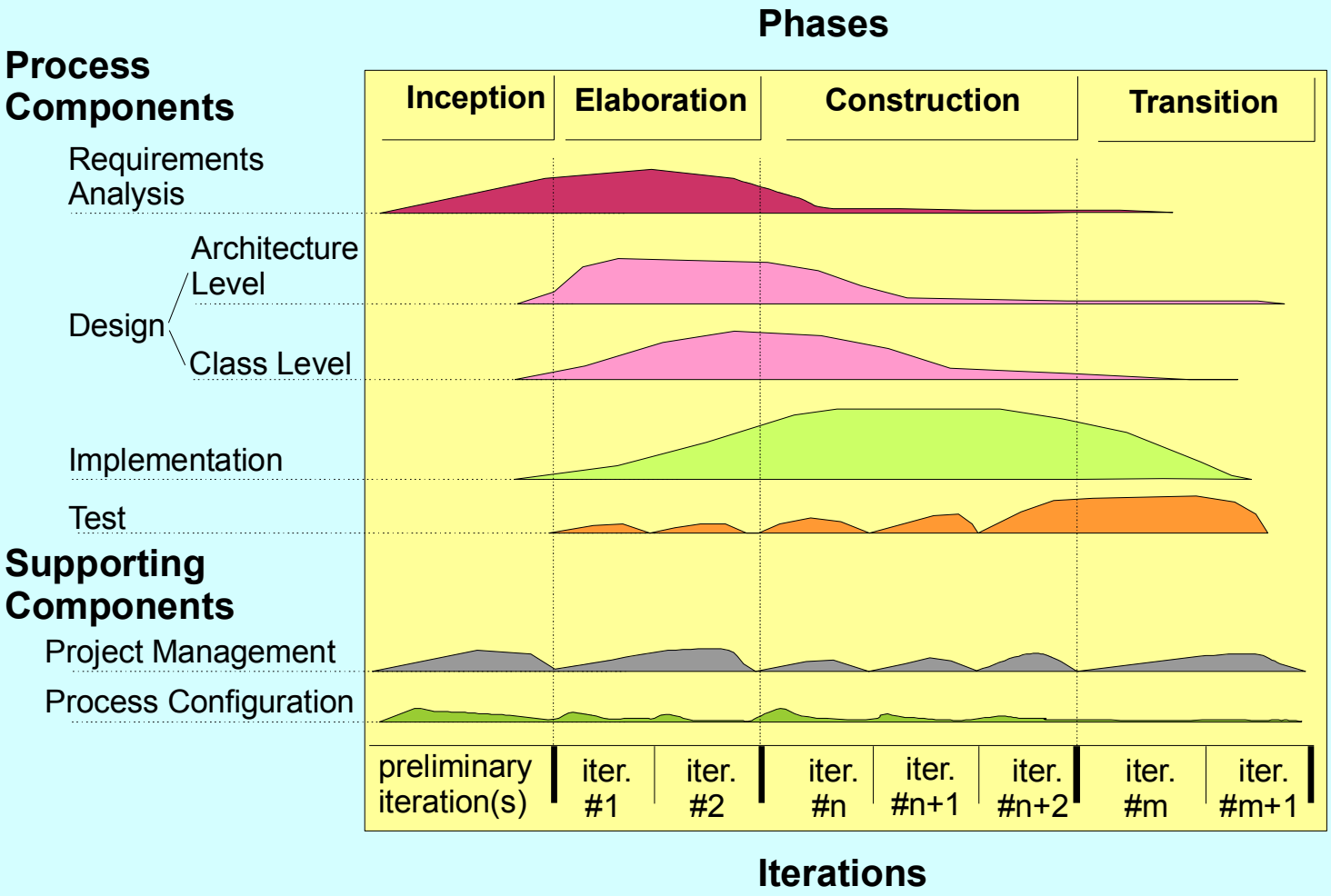


The Rational Unified Process (RUP)
System Development Life Cycle

RUP Life Cycle – Phases

- Inception
- Determine the scope and purpose of the project
- Elaboration
- Create a project plan, capture the requirements and determining the structure of the system
- Construction
- Build the software system
- Transition
- Install and rollout the product

Rational Unified Process (RUP)



1. Inception Phase

Project starts

Define your business case

Do Requirements analysis – Requirement Doc

Make a rough estimation of the cost and project duration and effort – Project Plan

1. Inception Phase

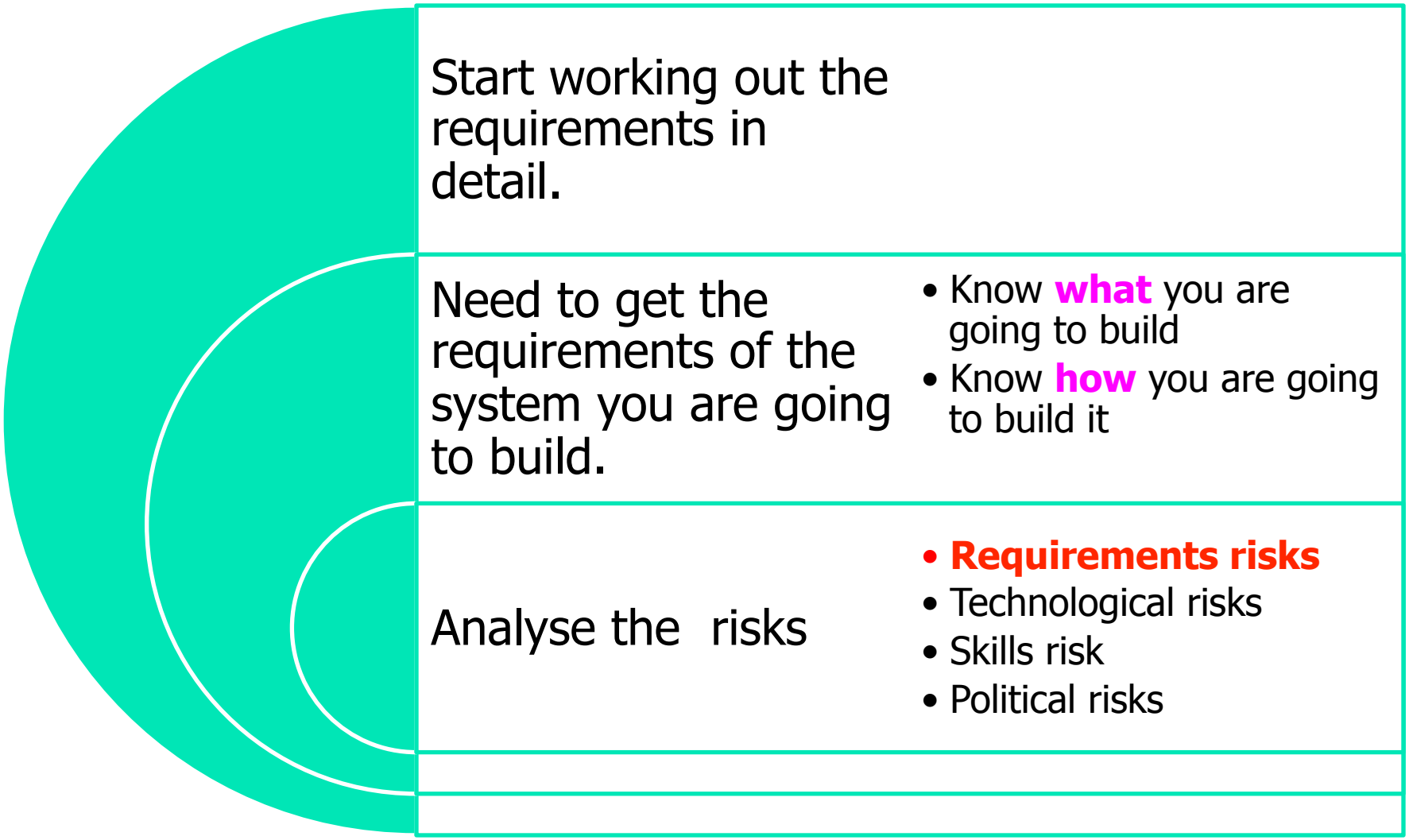
Do a few days' work to see if it's worth doing a few months' work

Do a few months' work to see if it's worth doing a few years' work.

Get the agreement of the project sponsor to go ahead with the project.

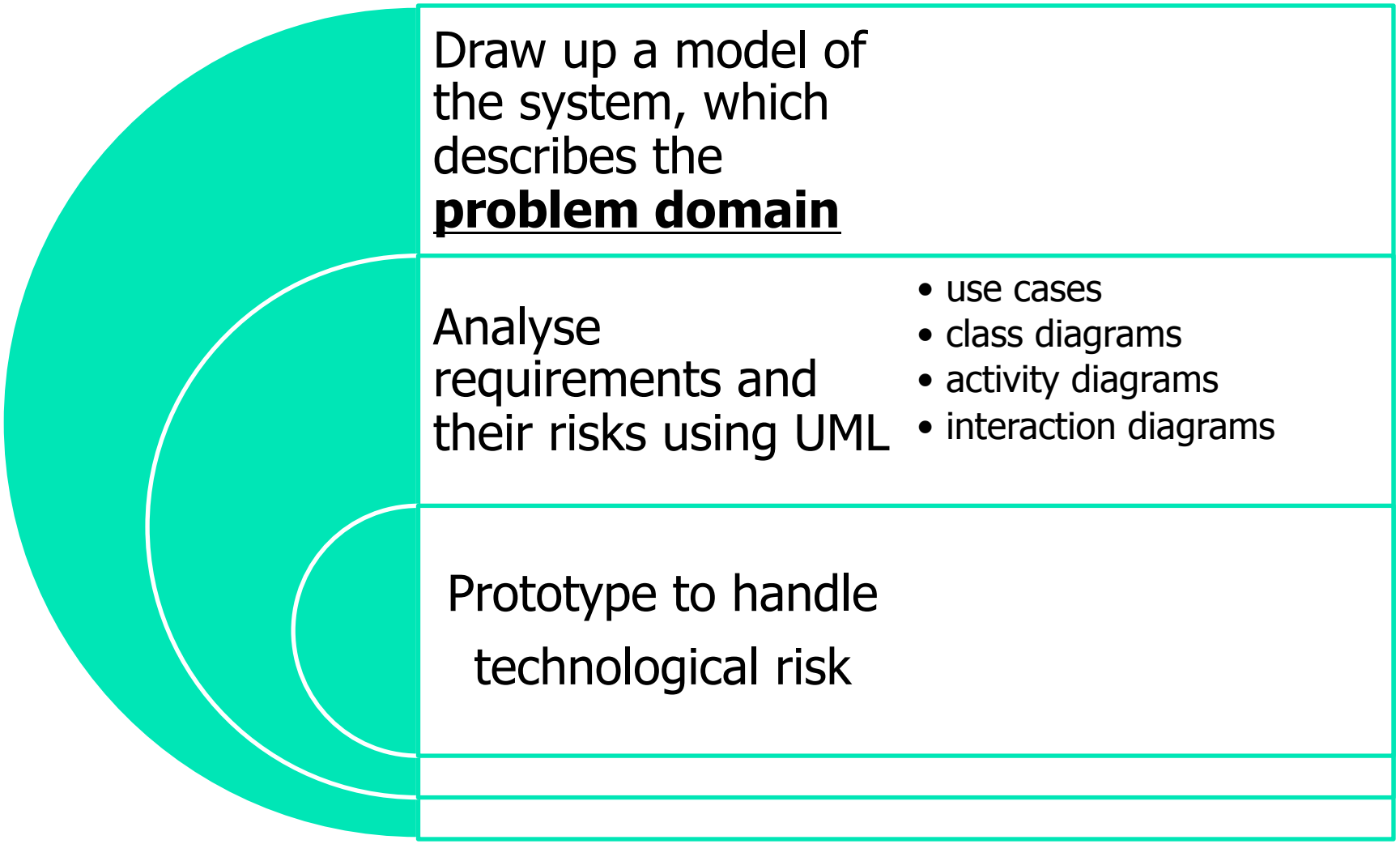
Get green light or red light from sponsor

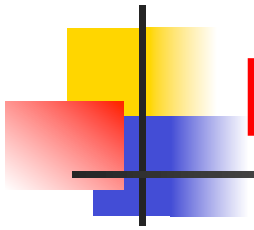
2. Elaboration Phase



- Requirements risk is mitigated by gathering as many **use cases** as possible, hopefully all of them.
- At the same time, a conceptual model of the domain is produced using **class diagrams** drawn from the conceptual perspective.
- **Activity diagrams** might well be used to model business processes and identify activities that can take place in parallel.
- **State Chart diagrams** might be drawn for classes that have interesting life cycles
- **Class diagrams** that occupy more than an A4 page could well be broken up using package diagrams.

2. Elaboration Phase





Notes on Elaboration Phase

- Collecting further requirements
- Do high level analysis and design to establish a baseline architecture
- Basically reaching closure on the model and prepare a plan for construction

3. Construction Phase

Main part of the RUP in this phase

Do detailed design and the software implementation

Use UML diagrams

- Use Case + Class diagrams
- Interaction Diagrams
- State diagrams

Usually done in 2-3 or even more iterations.

System is built incrementally.

- The plan for construction is **use case-based** where use cases are divided up first according to importance and then according to complexity.

- The construction phase consists of a number of **time-boxed iterations** each of which produces production quality software.

- The iterations may be internal, limited to early users or completely external and revenue-earning.

4. Transition Phase

Final Phase- **Install and rollout the product**

Activities

- Error fixing - major activity here
- Performance tuning
- Document the system
- Beta testing
- User training

Note:

no extra functionality added in this phase

RUP – Strengths & Weaknesses

- Strengths:
- UML is not dependent on RUP but RUP is a process that helps glue UML to a process
  - Iterative
  - Incremental
  - Model-based
    - uses UML
  - Use case driven
    - Functional requirements / scenarios / testing
  - Object Oriented
    - Classes & relationships
  - Can be tailored to fit an organisation
  - Encourages continuous quality-control and risk management