Software Process Models

Target group: 4th year Computer engineering students

ECE, GIT,DTU

By Misganaw Aguate

MSc. In Computer Engineering

ethiomisgie@gmail.com

YouTube:ethioptech

https://project.ethioptec.com

https://academics.ethioptec.com

Topics includes…

- PREDICTIVE MODELS
- ITERATIVE MODELS
- RAD MODELS

SW process Models

- A **Software Process Model** is a structured framework that defines the sequence and methods used to
- Develop
- Deploy and
- maintain software.
- It acts as a **roadmap** for software development, **specifying** the activities, **their order**, and the **roles of team** members.
- The primary goal of these models is to improve software quality, manage project timelines, and ensure effective collaboration among stakeholders

SW process Models

Key Characteristics of Software Process Models

- **Systematic Approach:** Breaks down software development into manageable phases.
- **Guidance:** Provides a clear pathway for project execution.
- **Repeatability:** Ensures consistent outcomes for similar projects.
- **Flexibility:** Some models adapt to changing requirements or uncertainties.
- **Scalability:** Models can be scaled to suit small, medium, or large projects

Predictive Models

- In a ***predictive development*** *model*, you predict in advance what needs to be done and then you go out and do it.
- You use the requirements to design the system, and you use the design as a **blueprint** to write the code.

Predictive Models cont…

Success Indicators:

- User involvement
- Clear vision
- Limited size
- Experienced team
- Realistic
- Established technology

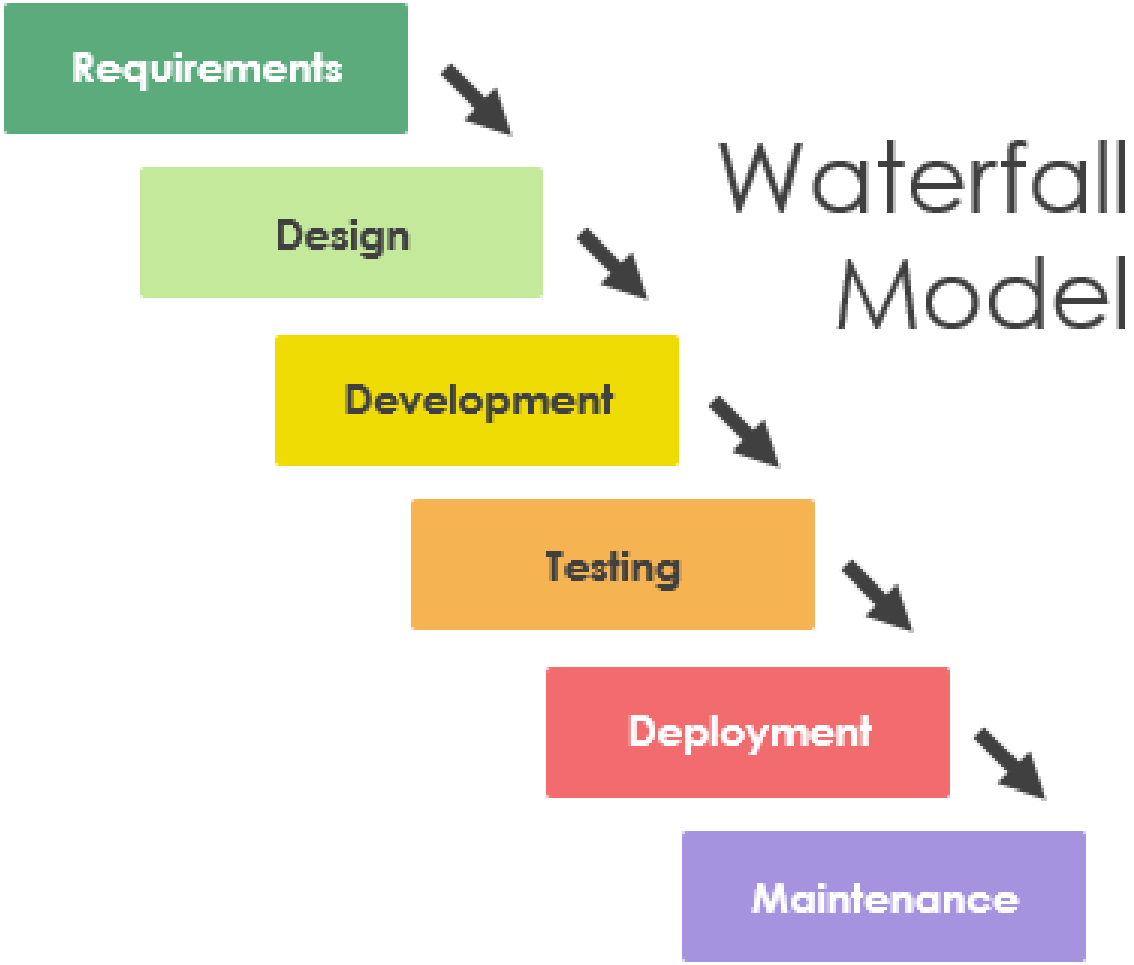
Failure Indicators:

- Incomplete requirements
- Unclear requirements
- Changing requirements
- No resources

Predictive Models cont…

Waterfall Model

Waterfall is the predictive mode. It assumes that you finish each step completely and thoroughly before you move on to the next step.



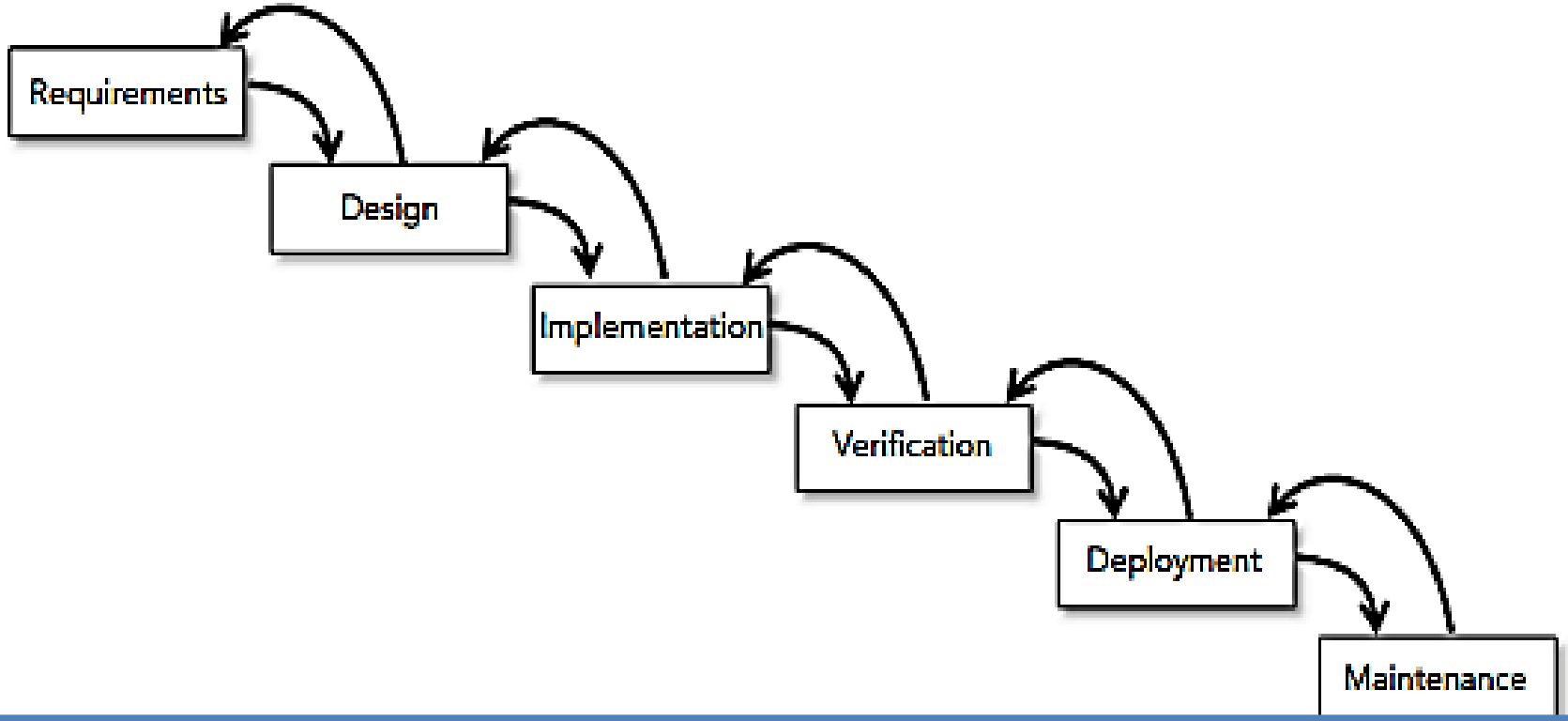
Predictive Models cont…

Assumptions on Waterfall Model

- The requirements are precisely known in advance.
- The requirements include no unresolved high-risk items.
- The requirements won't change much during development.
- The team has previous experience with similar projects so that they know what's involved in building the application.
- There's enough time to do everything sequentially.

Predictive Models cont…

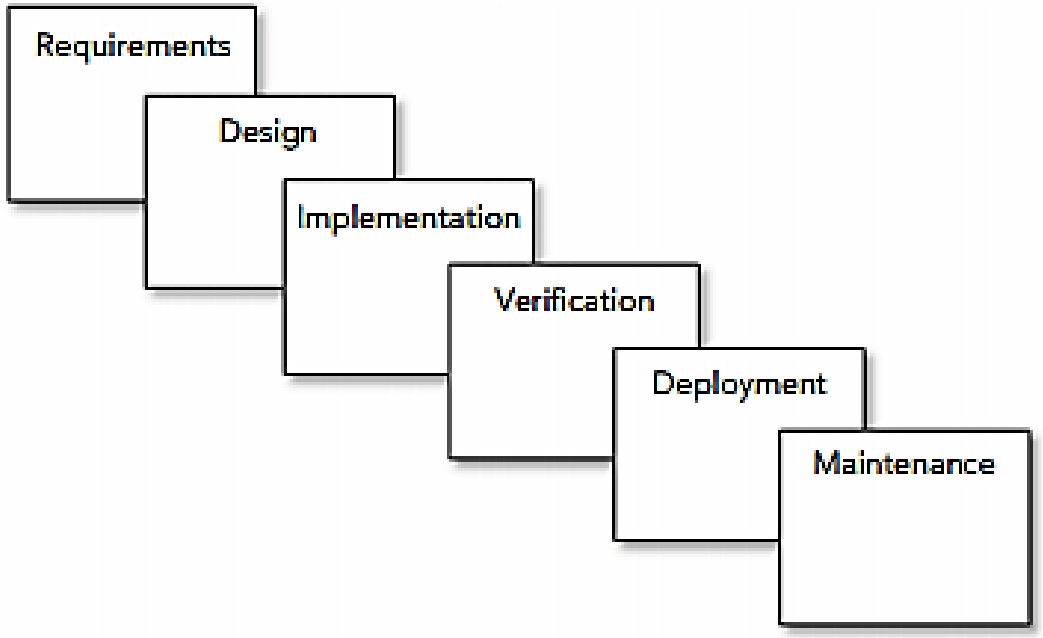
Waterfall with Feedback Model. The *waterfall with feedback* variation enables you to move backward from one step to the previous step.



Predictive Models cont…

Sashimi Model

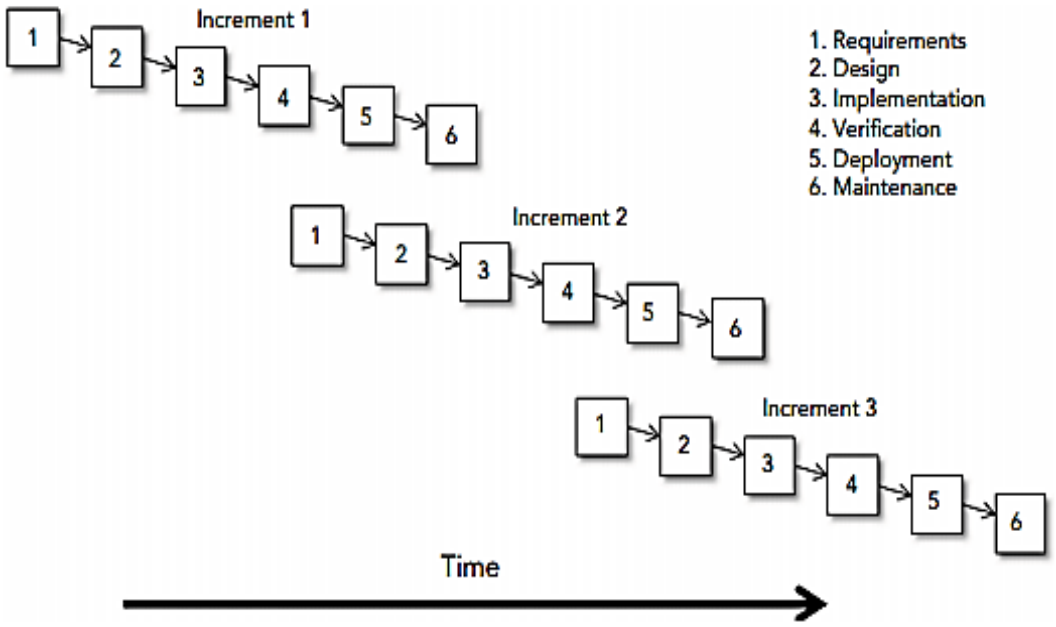
- *Sashimi* (also called *sashimi waterfall* and *waterfall with overlapping phases*) is similar to the waterfall model except the steps are allowed to overlap.
- (Much as the thin slices of fish overlap in the Japanese dish sashimi.)



Predictive Models cont…

Incremental Waterfall model

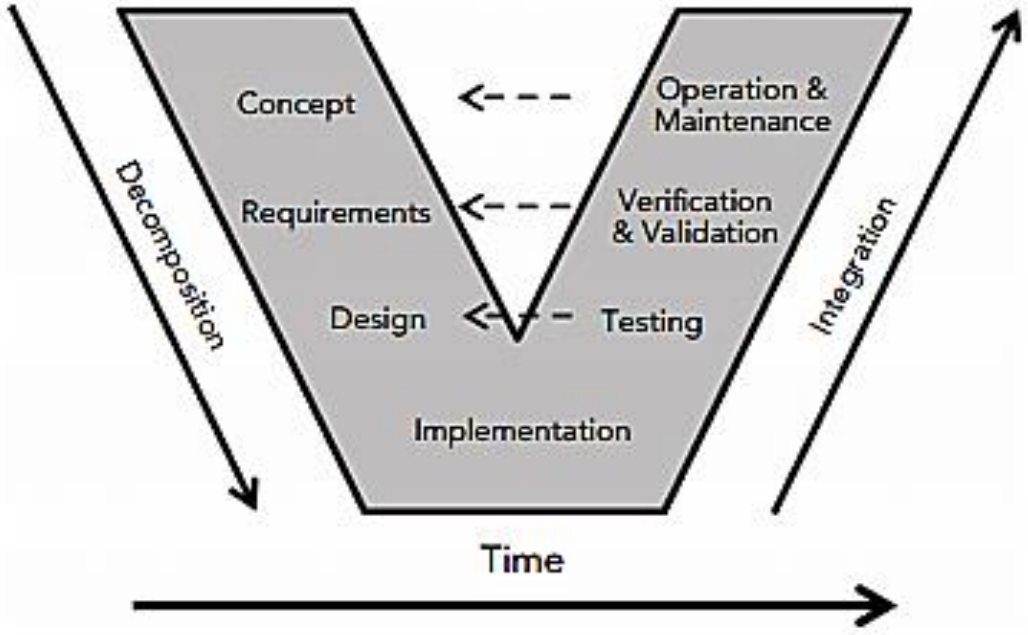
The *incremental waterfall* model (also called the *multi-waterfall* model) uses a **series of separate** waterfall cascades. Each cascade ends with the **delivery** of a usable application called an ***increment***. Each increment includes more features than the previous one, so you're building the final application incrementally.



Predictive Models cont…

V Model

V-model is basically a waterfall that's been bent into a **V shape**. The tasks on the **left side** of the V break the application down from its highest conceptual level into more and more detailed tasks. This process of breaking the application down into pieces that you can implement is called *decomposition*.



Predictive Models cont…

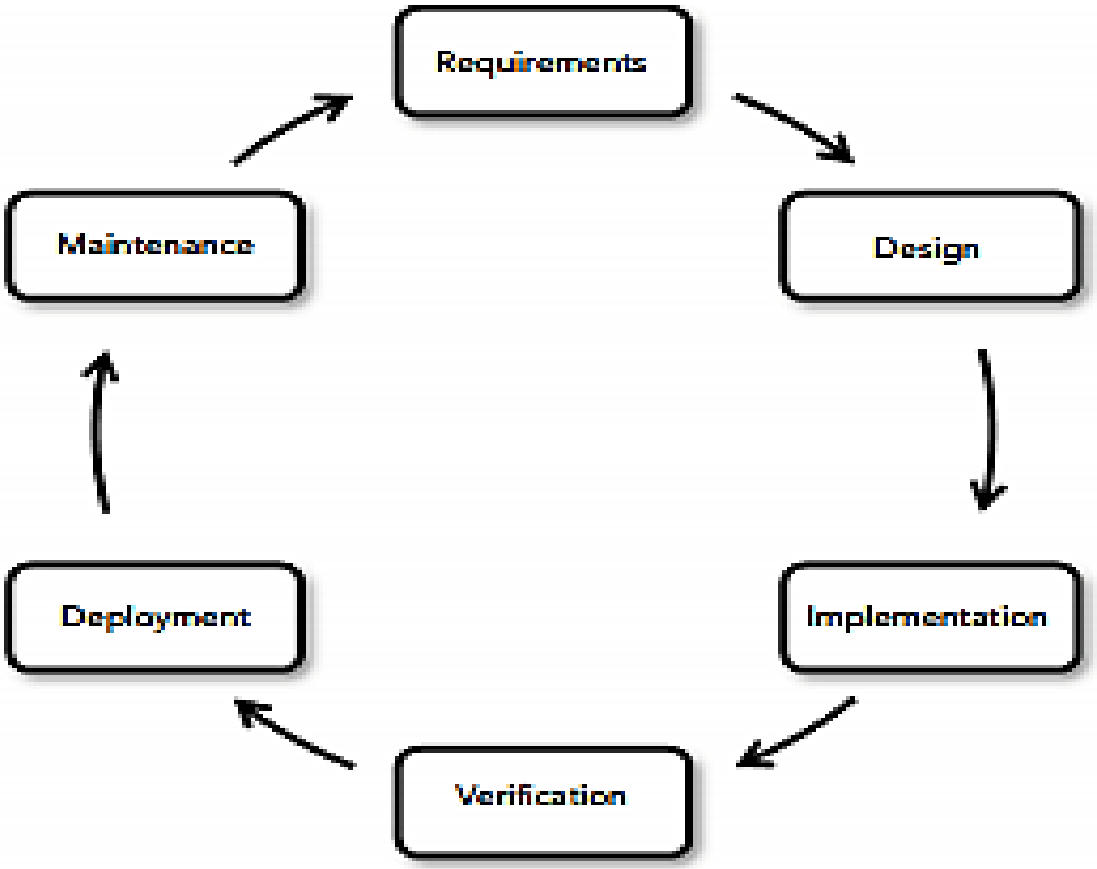
- The tasks on the right side of the V consider the **finished application** at greater and greater levels of abstraction.
- At the **lowest level**, testing verifies that the code works.
- At the **next level**, **verification** confirms that the application satisfies the requirements, and validation confirms that the application meets the customers' needs. This process of working back up to the conceptual top of the application is called ***integration***.

Predictive Models cont…

Systems Development Life Cycle.

The *software development life cycle* (*SDLC*) covers all the tasks that go into a software engineering project from start to finish:

- Requirements
- Design
- implementation, and so forth.



Iterative Models

Iterative Model

- Iterative models address those problems by building the application **incrementally**.
- They start by building the **smallest** program that is reasonably useful.
- Then they use a series of ***increments*** to add more features to the program until it's finished (if it is ever finished).

Iterative Models cont…

Prototypes: Typically, a software prototype is a program that **mimics** part of the application you want to build.

Types:

1. In a ***throwaway*** *prototype*, you use the prototype to study some **aspect** of the system and then you **throw it** away and write code from scratch.
2. In an ***evolutionary*** *prototype*, the prototype demonstrates some of the application's features. As the project progresses, you refine those features and add new ones until the prototype morphs into the finished application.
3. In ***incremental*** *prototyping*, you build a collection of prototypes of that separately demonstrate the finished application's features. You then combine the prototypes

Iterative Models cont…

Advantages:

- Improved requirements
- Common vision
- Better design

Disadvantages:

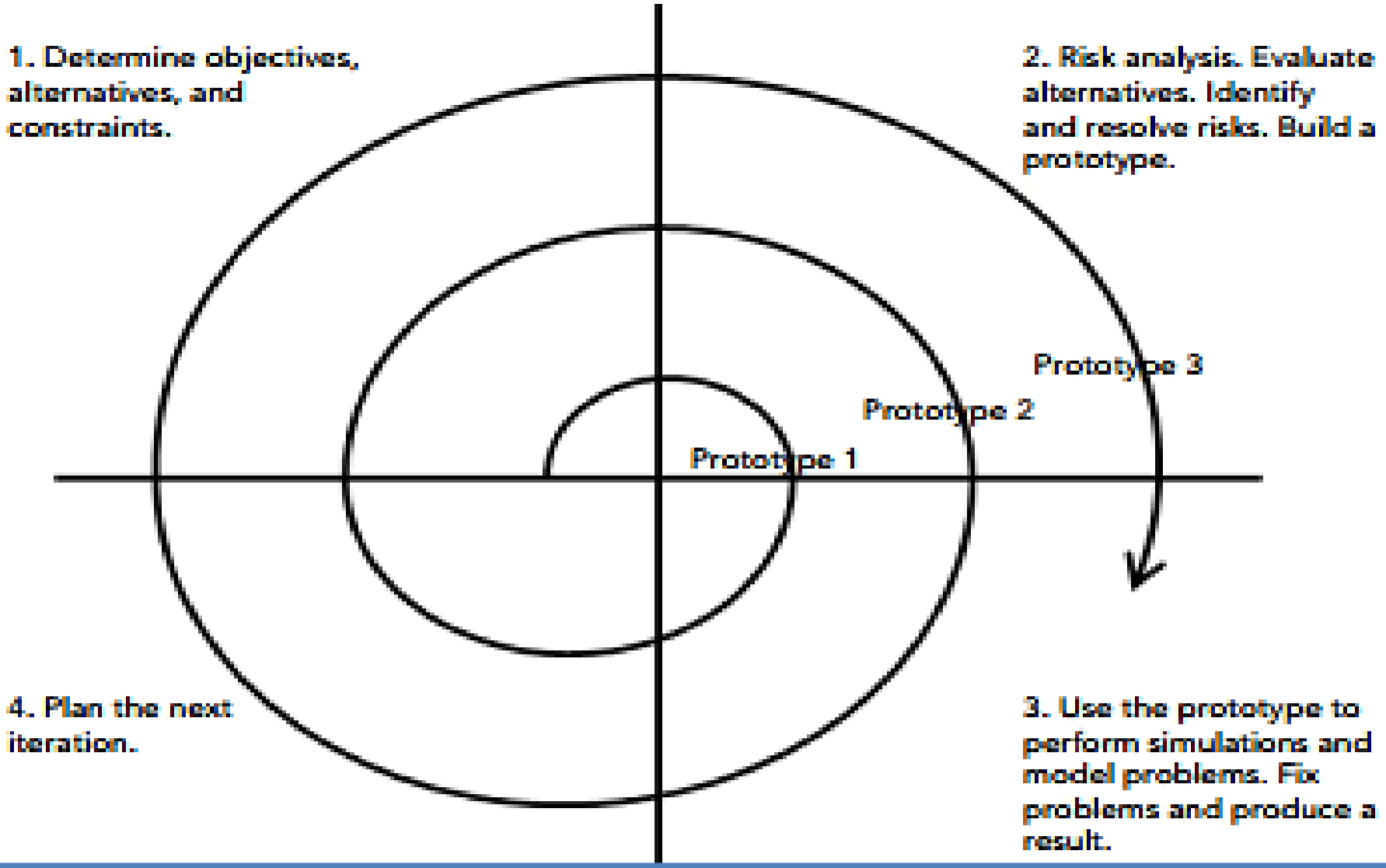
- Narrowing vision
- Customer impatience
- Schedule pressure
- Raised expectations
- Attachment to code
- Never-ending prototypes

Iterative Models cont…

Spiral Model

- The spiral model is a “process model **generator**.”
- It uses a risk-driven approach to help project teams decide on what development approach to take for various parts of the project.
- For example, if you don’t understand all the requirements, then you might use an iterative approach for developing them.

Iterative Models cont…



Iterative Models cont…

Advantages:

- Its spiral structure gives stakeholders a lot of points for review and making “go” or “no-go” decisions.
- It emphasizes risk analysis. If you identify and resolve risks correctly, it should lead to eventual success.
- It can accommodate change reasonably well. Simply make any necessary changes and then run through a cycle to identify and resolve any risks they create.
- Estimates such as time and effort required become more accurate over time as cycles are finished and risks are removed from the project.

Iterative Models cont…

Disadvantages:

- It’s complicated.
- Because it’s complicated, it often requires more resources than simpler approaches.
- The complication isn’t always worth the effort, particularly for low-risk projects.
- Stakeholders must have the time and skills needed to review the project periodically to make sure each cycle is completed satisfactorily.
- Time and effort estimates initially may not be good.
- It doesn’t work well with small projects. You could end up spending more time on risk analysis than you’d need to build the entire application with a simpler

Iterative Models cont…

Unified Process.

- Despite its name, the *Unified Process* (*UP*) isn't actually a process.
- Instead it's an **iterative** and **incremental** development framework that you can customize to fit your business and projects.

Phases:

1. Inception

- During this phase you come up with the project's idea.
- This should be a short phase where you provide a business case, identify risks, provide an initial schedule, and sketch out the project's general goals.
- It should not include detailed requirements that might restrict the developers.

Iterative Models cont…

2. Elaboration

- During this phase you create the project requirements.
- You build use cases, architectural diagrams, and class hierarchies.
- You need to specify the system, but you still don't want to restrict developers with unnecessarily detailed requirements.
- The main goals are to identify and address risks so that the project doesn't fail later.
- Normally, this phase is divided into several iterations with the first addressing the most important risks.

Iterative Models cont…

3. Construction

- During this phase you write, test, and debug the code.
- This phase is divided into several iterations, each of which ends with a tested, high-quality working executable program that you can release to the users.
- The iterations implement the most important features first.

Iterative Models cont…

4. Transition

- During this phase you transfer the **project to customers** and the **long-term** maintenance team.
- Based on **feedback** from users, you might make **changes** and refinements and then release a new **version**, so this phase can include several iterations.
- This phase includes all the usual transitioning tasks such as **staging**, **building** the **user environment** (computers, networks, coffee machines, and so forth), user documentation, and user training.

Iterative Models cont…

5. Production

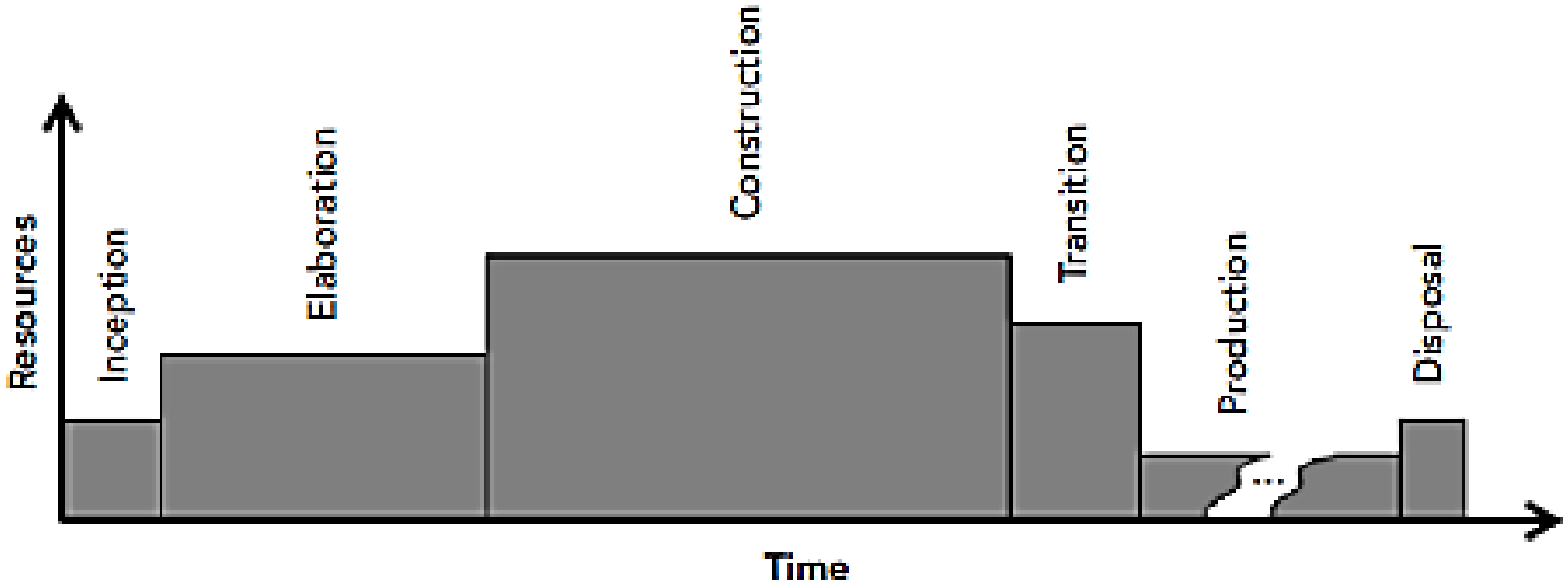
- During this phase users use the **application**.
- The normal **Unified Process** assumes that the development **team** doesn't **continue producing new versions** of the application during this phase.

6. Disposal

- During this phase you remove the application and move users to a replacement system.
- If you're building the replacement, then this phase overlaps with the new project's transition phase.

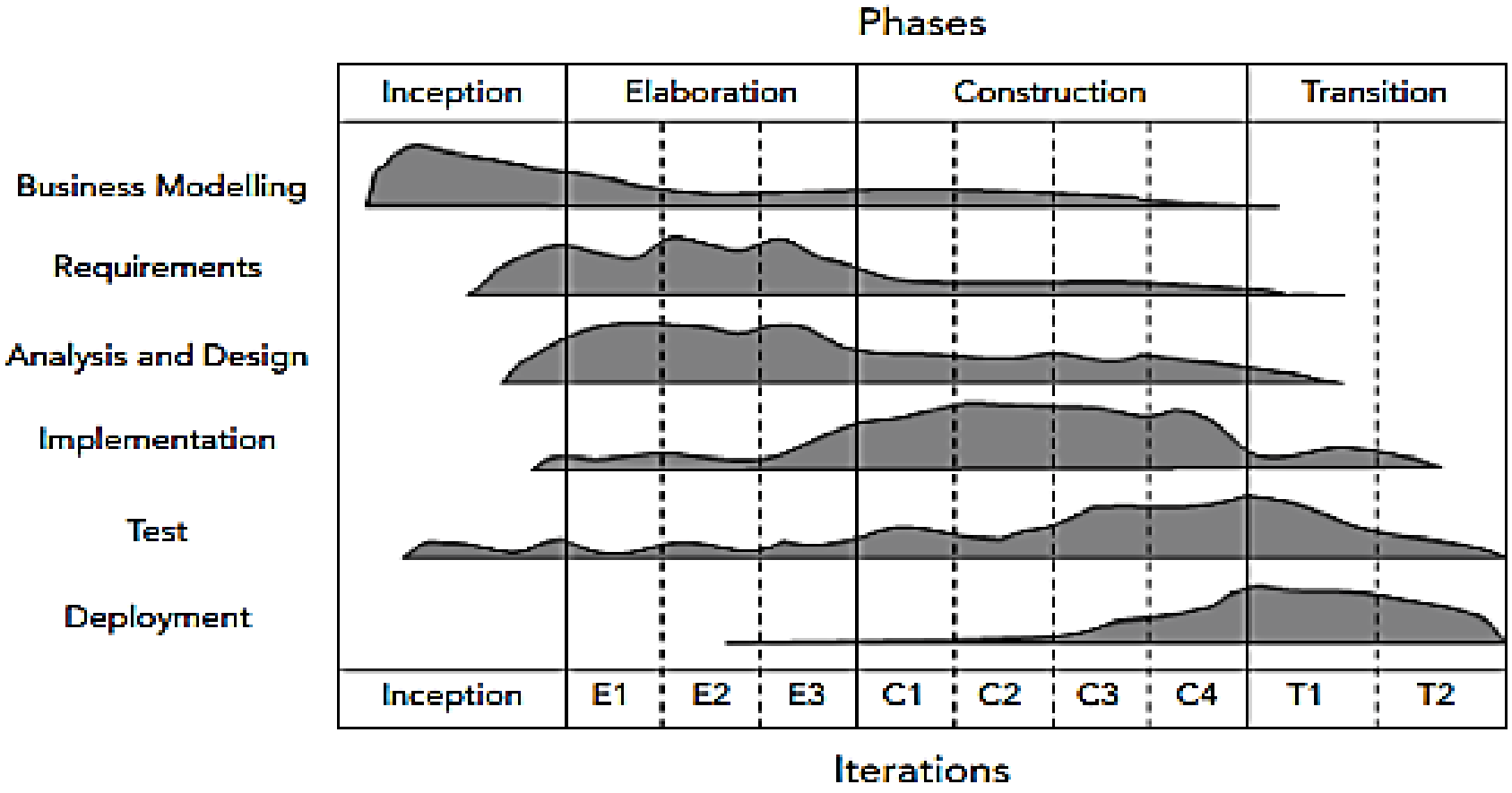
Iterative Models cont…

In the **Unified Process**, construction takes more time and effort than the other phases.



Iterative Models cont…

In the Unified Process, the amounts of different kinds of work grow and shrink during different **project phases**.



Iterative Models cont…

Advantages:

- The iterative approach to the elaboration, construction, and transition phases enables you to incrementally define the requirements and assemble the application.
- The elaboration iterations focus on risks and risk mitigation to improve the project's chance of success.
- It can accommodate different development models flexibly. For example, you could use a series of waterfalls or an agile approach to the construction phase.
- The inception and elaboration phases generate a lot of documentation that can help new developers join the team later.
- It can enable incremental releases if wanted.

Iterative Models cont…

Disadvantages:

- It's complicated (although not quite as confusing as the spiral approach).
- Because it's complicated, it often requires more resources than simpler approaches.
- Risk analysis can be difficult.
- The complication isn't always worth the effort, particularly for low-risk projects.
- It doesn't work well with small projects. You could end up spending more time on inception and elaboration than you'd need to build the entire application with a simpler approach.

Iterative Models cont…

Basic Principles.

- **Formal methods:** Code is produced using formal mathematical methods that help ensure that the code satisfies the design models. Code reviews also help verify that the code correctly implements the required behavior.
- **Statistical quality control:** Code is produced incrementally. Each increment's quality is measured to ensure that the project is making acceptable progress.
- **Statistical testing:** Testing uses statistical experiments to estimate the application quality. (This requires some serious statistical analysis so that you can estimate not only the application's quality but so that you can also calculate a level of confidence for that estimate.)

Rapid Application Dev't Models

- RAD methods take iterative ideas to the extreme.
- Instead of using iterations lasting a year or two, their iterations last a month, a week, or even less.
- Some RAD techniques also apply iteration to everything, not just to programming.
- They apply iteration to requirement gathering, requirement validation, and design.

RAD Models cont…

Common Techniques used in RAD

- Small teams (approximately one-half a dozen people or fewer). That leads to projects of limited scope. (Six people probably can't write a million-line application in a year.)
- Requirement gathering through focus groups, workshops, facilitated meetings, prototyping, and brainstorming.
- Requirement validation through iterated prototypes, use cases, and constant customer testing of designs.
- Repeated customer testing of designs as they evolve.
- Constant integration and testing of new code into the application.
- Informal reviews and communication among team members.
- Short iterations lasting between a few months and as little as a week.
- Deferring complicated features for later releases.
- *Timeboxing*, which is RADspeak for setting a tight delivery schedule for producing something, usually the next iteration of the application.

RAD Models cont…

Advantages:

- More accurate requirements.
- The ability to track changing requirements.
- Frequent customer feedback and involvement.
- Reduced development time.
- Encourages code reuse.
- Possible early releases with limited functionality
- Constant testing promotes high-quality code and eases integration issues.
- Risk mitigation.
- Greater chance of success.

RAD Models cont…

Disadvantages:

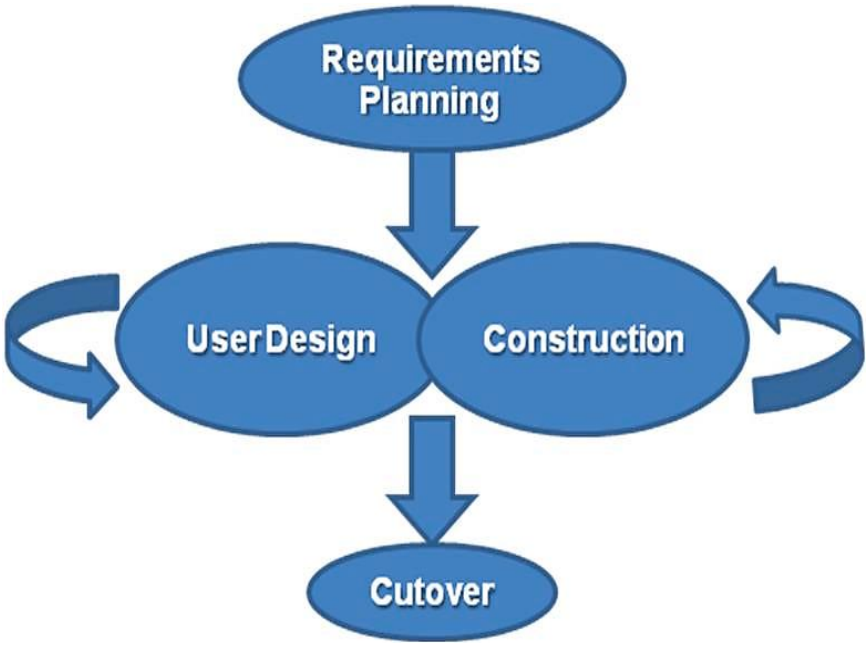
- Resistance to change.
- Doesn’t handle large systems well.
- Requires more skilled team members.
- Requires access to scarce resources.
- Adds extra overhead if the requirements are known completely and correctly in advance.
- Less managerial control.
- Sometimes results in a less than optimal design..
- Unpredictability.

RAD Models cont…

Four Phases of RAD:

Requirements planning

- During this phase, the users, executive champion, management, team leaders, and other stakeholders agree on the project's general goals and requirements.
- The requirements should be specified in a general way so that they don't restrict later development unnecessarily.
- When the stakeholders agree on the requirements and the project receives approval to continue, the user design

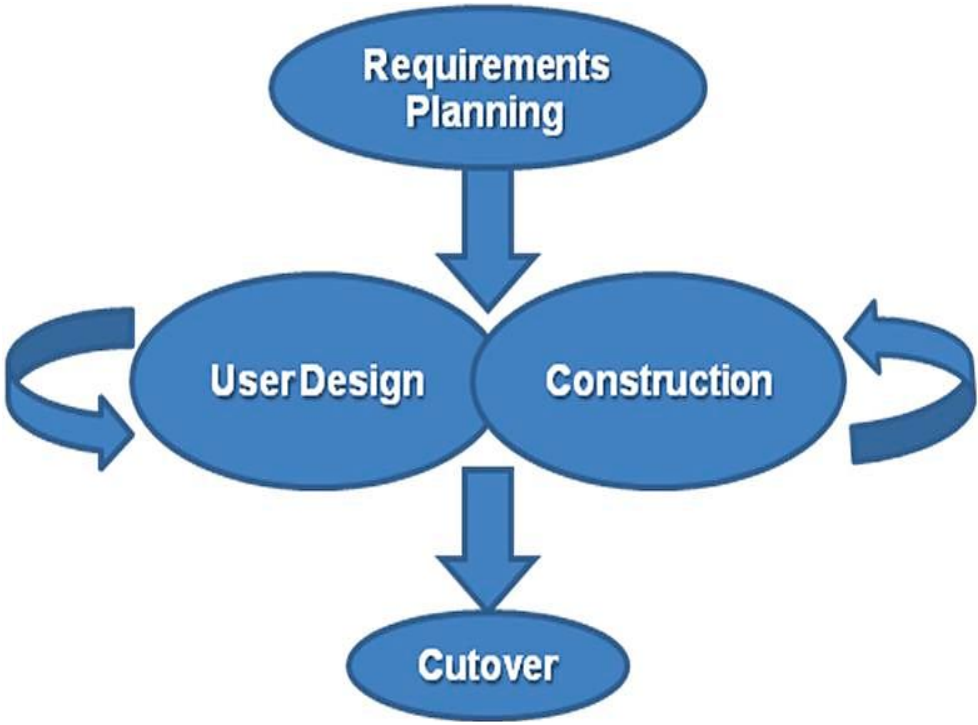


RAD Models cont…

Four Phases of RAD:

User design

- The users and team members work together to convert the requirements into a workable design.
- They use techniques such as focus groups, workshops, prototyping, and brainstorming to come up with a workable design.

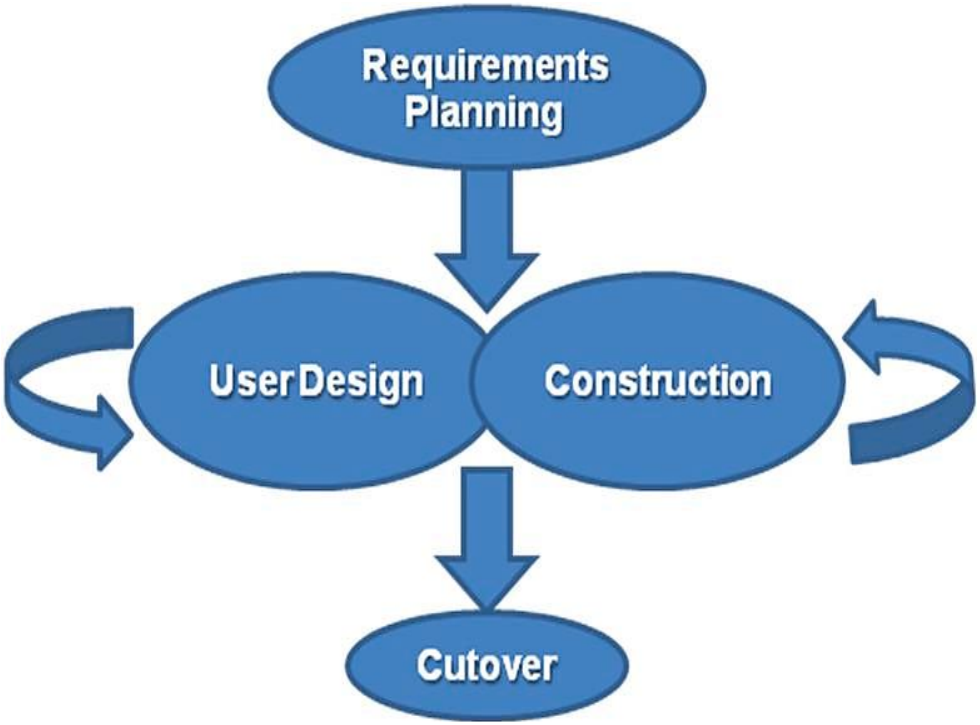


RAD Models cont…

Four Phases of RAD:

Construction

- The developers go to work building the application.
- The users continue to review the pieces of the application as the developers build them to make corrections and suggestions for improvements.

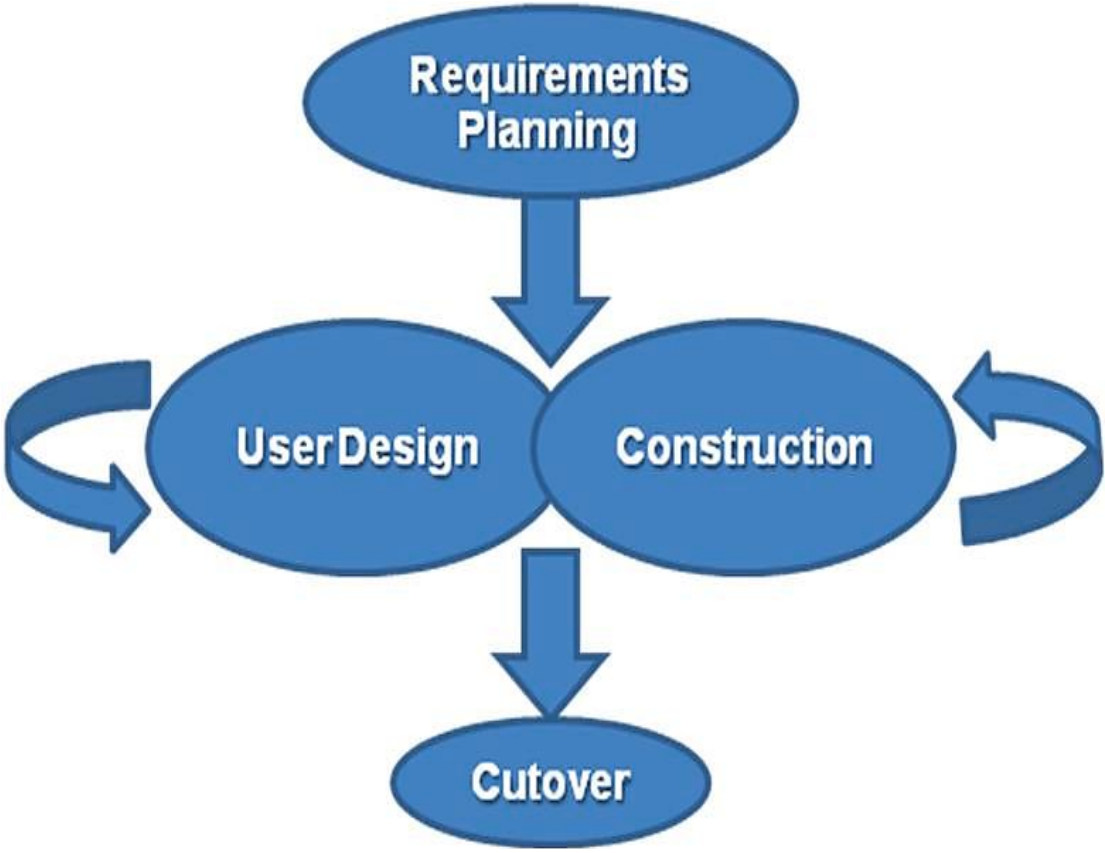


RAD Models cont…

Four Phases of RAD:

Cutover

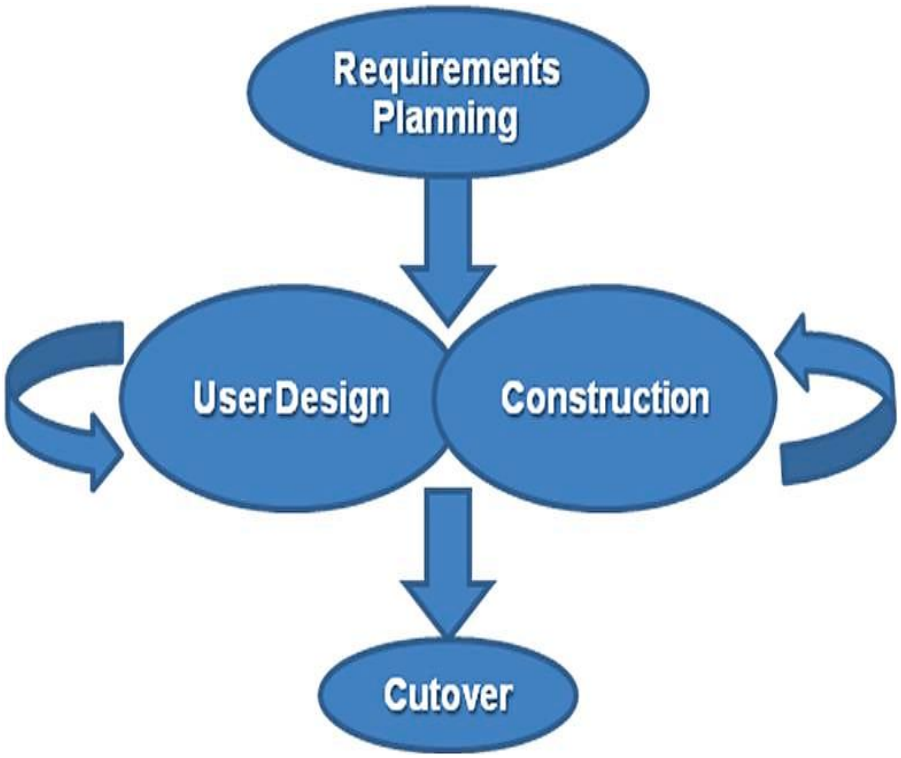
- The developers deliver the finished application to the users.
- You can use the usual cutover strategies such as staged delivery, gradual cutover, or incremental



RAD Models cont…

Note:

- The user **design** and **construction** phases overlap with the users constantly providing adjustments to the developers in a sort of continuous feedback loop.
- The project iterates the user design and construction phases as needed.
- When the application has met all the requirements, it is delivered to



RAD Models cont…

Type of RAD model

AGILE

- Agile development is more a set of **guidelines** than an actual development model.
- It includes a set of principles that its founders believe can help with any development effort.
- Because it's a set of guidelines, there are many ways you can interpret its rules.
- For **example**, people often say a particular method is "an agile technique" because it attempts to address one or more of the guidelines.

RAD Models cont…

Techniques:

Communication

- Agile projects use frequent (sometimes practically continuous) **customer communication** to keep the project on track.
- The customers **examine** the most recent iteration and then offer corrections, suggestions, and change requests.
- The developers adjust the next iteration accordingly.

RAD Models cont…

Techniques cont…

Incremental Development.

- Agile projects are iterative and incremental.
- The iterations are relatively short, with durations of a week to a couple months.
- Iterations are timeboxed to keep the project moving briskly along.
- Each iteration incorporates every development step, including requirements analysis, design, programming, testing, and verification.
- An iteration is basically a mini-project all by itself.

RAD Models cont…

Techniques cont…

Focus on Quality.

- In agile projects, all development must have high quality.
- Because of the fast iteration cycle, developers don't have time to spend chasing down bugs that entered the code months ago.

RAD Models cont…

Extreme Programming (XP).

- On this model two (or possibly even three) programmers sit in front of the same monitor and working on a piece of code together.
- One of them (called the driver or pilot) controls the keyboard.
- As he/she types, the driver keeps up a steady monologue explaining what he's/she;s doing.
- Or more properly, the monologue explains what the driver thinks he's/she's doing.

RAD Models cont…

Extreme Programming (XP).

- The second programmer of the pair (called the observer, navigator, or pointer) watches and reviews each line of code as it is typed.
- The observer makes sure the code makes sense and does what the driver thinks it does.
- The observer can also think about possible improvements or future changes.
- Basically, the observer performs an extreme, real-time, line-by-line code review.

RAD Models cont…

SCRUM

- Scrum was named after a procedure in rugby that the teams use to put the ball back into play after an accidental rule violation such as the ball going out of bounds.
- The players huddle together in a big interlocked mob, and then someone throws the ball under the players' legs.
- The players try to hook the ball out with their feet and take possession of it.

Selection of a Life Cycle Model

Selection of a model is based on:

- Requirements
- Development team
- Users
- Project type and associated risk

Selection of a Life Cycle Model cont…

Based On Characteristics Of Requirements

Requirements	Waterfall	Prototype	Iterative enhancement	Evolutionary development	Spiral	RAD
Are requirements easily understandable and defined?	Yes	No	No	No	No	Yes
Do we change requirements quite often?	No	Yes	No	No	Yes	No
Can we define requirements early in the cycle?	Yes	No	Yes	Yes	No	Yes
Requirements are indicating a complex system to be built	No	Yes	Yes	Yes	Yes	No

Selection of a Life Cycle Model cont…

Based On Status (experience) Of Development Team

Development team	Waterfall	Prototype	Iterative enhancement	Evolutionary development	Spiral	RAD
Less experience on similar projects?	No	Yes	No	No	Yes	No
Less domain knowledge (new to the technology)	Yes	No	Yes	Yes	Yes	No
Less experience on tools to be used	Yes	No	No	No	Yes	No
Availability of training if required	No	No	Yes	Yes	No	Yes

Selection of a Life Cycle Model cont…

Based On User's Participation

Involvement of Users	Waterfall	Prototype	Iterative enhancement	Evolutionary development	Spiral	RAD
User involvement in all phases	No	Yes	No	No	No	Yes
Limited user participation	Yes	No	Yes	Yes	Yes	No
User have no previous experience of participation in similar projects	No	Yes	Yes	Yes	Yes	No
Users are experts of problem domain	No	Yes	Yes	Yes	No	Yes

Selection of a Life Cycle Model cont…

Based On Type Of Project With Associated Risk

Project type and risk	Waterfall	Prototype	Iterative enhancement	Evolutionary development	Spiral	RAD
Project is the enhancement of the existing system	No	No	Yes	Yes	No	Yes
Funding is stable for the project	Yes	Yes	No	No	No	Yes
High reliability requirements	No	No	Yes	Yes	Yes	No
Tight project schedule	No	Yes	Yes	Yes	Yes	Yes
Use of reusable components	No	Yes	No	No	Yes	Yes
Are resources (time, money, people etc.) scare?	No	Yes	No	No	Yes	No

Review questions

1. What are the criteria to select the software process model for our project?
2. Suppose you early define the requirement to do a software project that has stable funding and needs limited number of user participations. The participants are new for the technology on which the project was developed. What type of process model you are going to apply?
3. Suppose you want to a software project with stable fund that needs a complex requirement to be built. The project needs users involvement in all phases in tight project schedule and users have less experience in the previous similar project. What type of process model you are going to apply?

Review questions

4. Suppose you early define the requirement to enhance the existing software project that needs limited number of user participations. The participants are new for the technology on which the project was developed. What type of process model you are going to apply?

5. If you want to develop a software project that has the following characteristic, what type process model will you apply?

- the project will require a complex system
- The participant will have less experience in project tools and similar projects
- The participant have also limited participation and have no previous experience on similar projects.
- Your project will use reusable components and tight schedule.

Review questions

End of chapter 2