PROCESS MODEL

Lecture 8

What is a Process Model ?

*It is a **description** of*

*i) <u>**what tasks need to be performed**</u> **in***

*ii) <u>**what sequence**</u> **under***

*iii) <u>**what conditions**</u> **by***

*iv) <u>**whom**</u> **to***

achieve the "desired results."

Why Have A Process Model?

- Provide **"guidance"** for a systematic coordination and controlling of
  - a) **the tasks** and of
  - b) **the personnel** who perform the tasks

Note the key words: coordination/control , tasks, people

Extending the "Simple" Process

As projects got *larger* and more *complex.*

- Needed to **clarify and stabilize the requirements**
- Needed to **test more functionalities**
- Needed to **design more carefully**
- Needed to **use more existing software & tools**
    - Database
    - Network
    - Code control
- Needed **more people** to be involved

Resulting in **more tasks** ***and*** **more people**

Effectiveness of using Correct Process Model

By changing the process model, we can **improve** :

- *Development speed (time to market)*
- *Product quality*
- *Project visibility*
- *Administrative overhead*
- *Risk exposure*
- *Customer relations, etc.*

Normally, a process model covers the entire **lifetime of a product**.

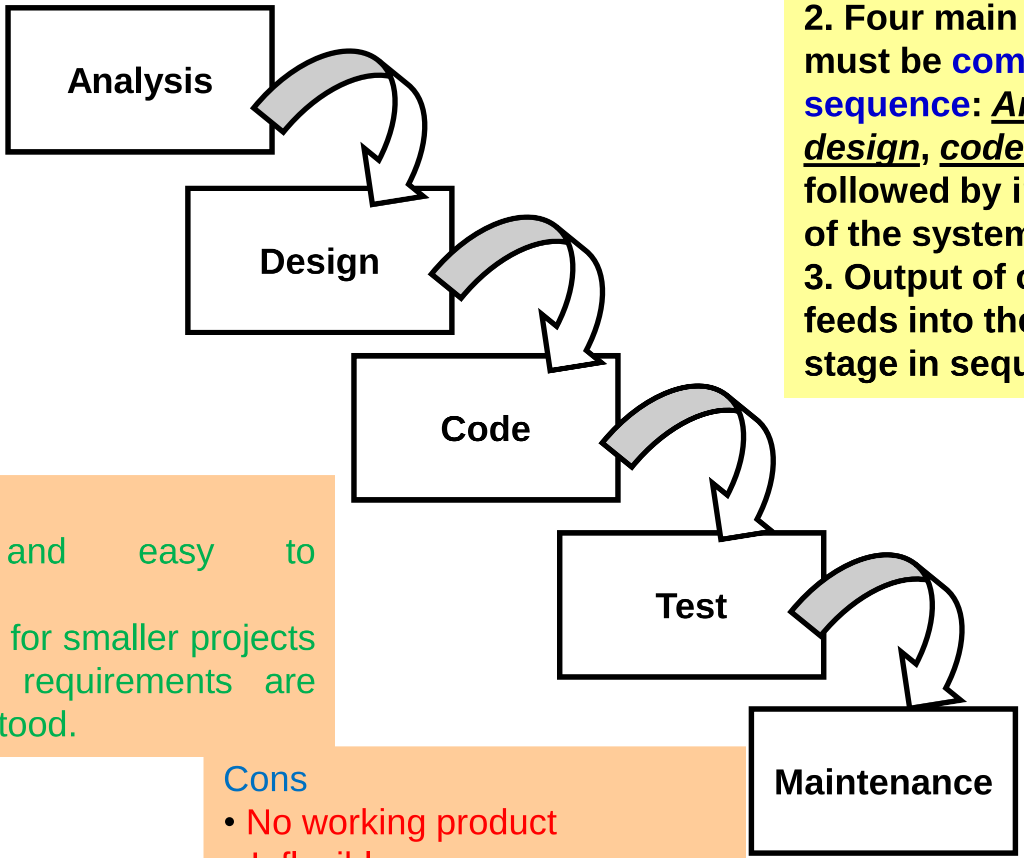
From ***birth of a commercial idea***
to ***final installation of last release***

Common Activities of Process Model

Many different software processes but all involve:

- Specification – defining what the system should do;
- Design and implementation – defining the organization of the system and implementing the system;
- Validation – checking that it does what the customer wants;
- Evolution – changing the system in response to changing customer needs.

Waterfall Model

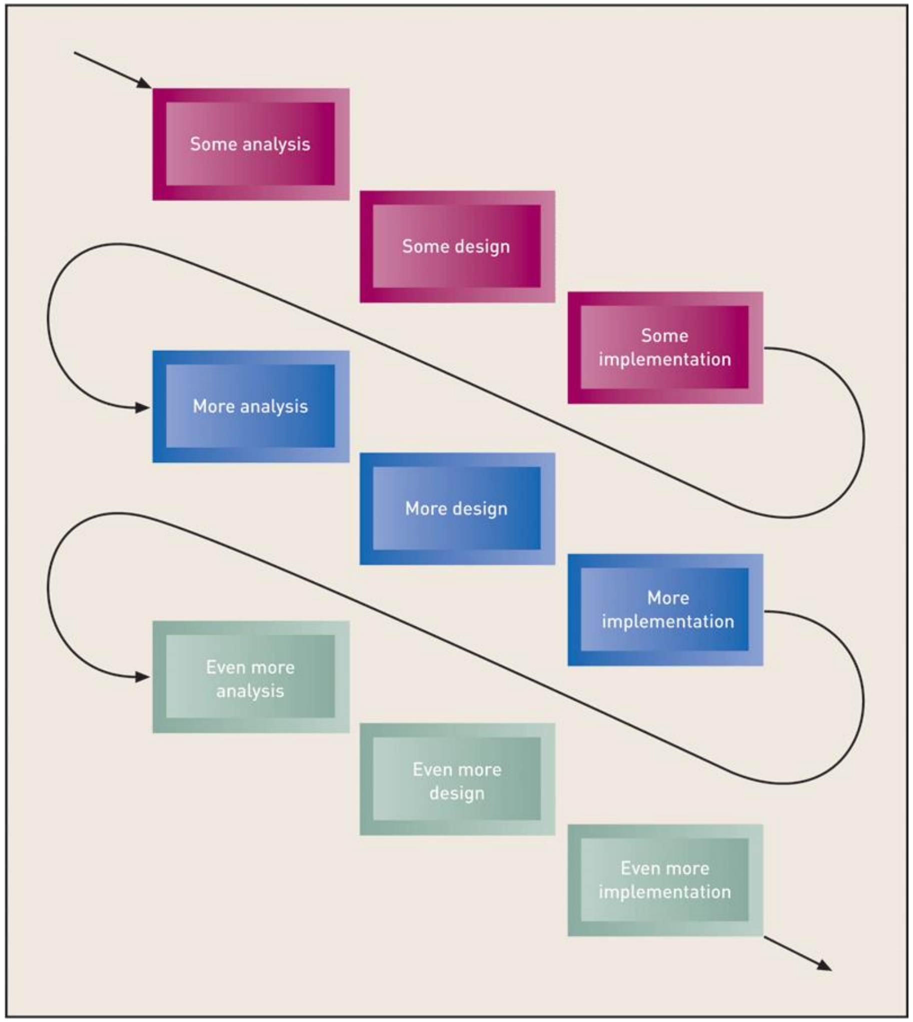


1. Requirements must be specified.
2. Four main tasks must be completed in sequence: ***Analysis***, ***design***, ***code***, and ***test,*** followed by integration of the system.
3. Output of one stage feeds into the next stage in sequence

Pros
- Simple and easy to understand.
- Works well for smaller projects where the requirements are well-understood.

Cons
- No working product
- Inflexible
- Poor model for large projects.

Iteration of System Development Activities



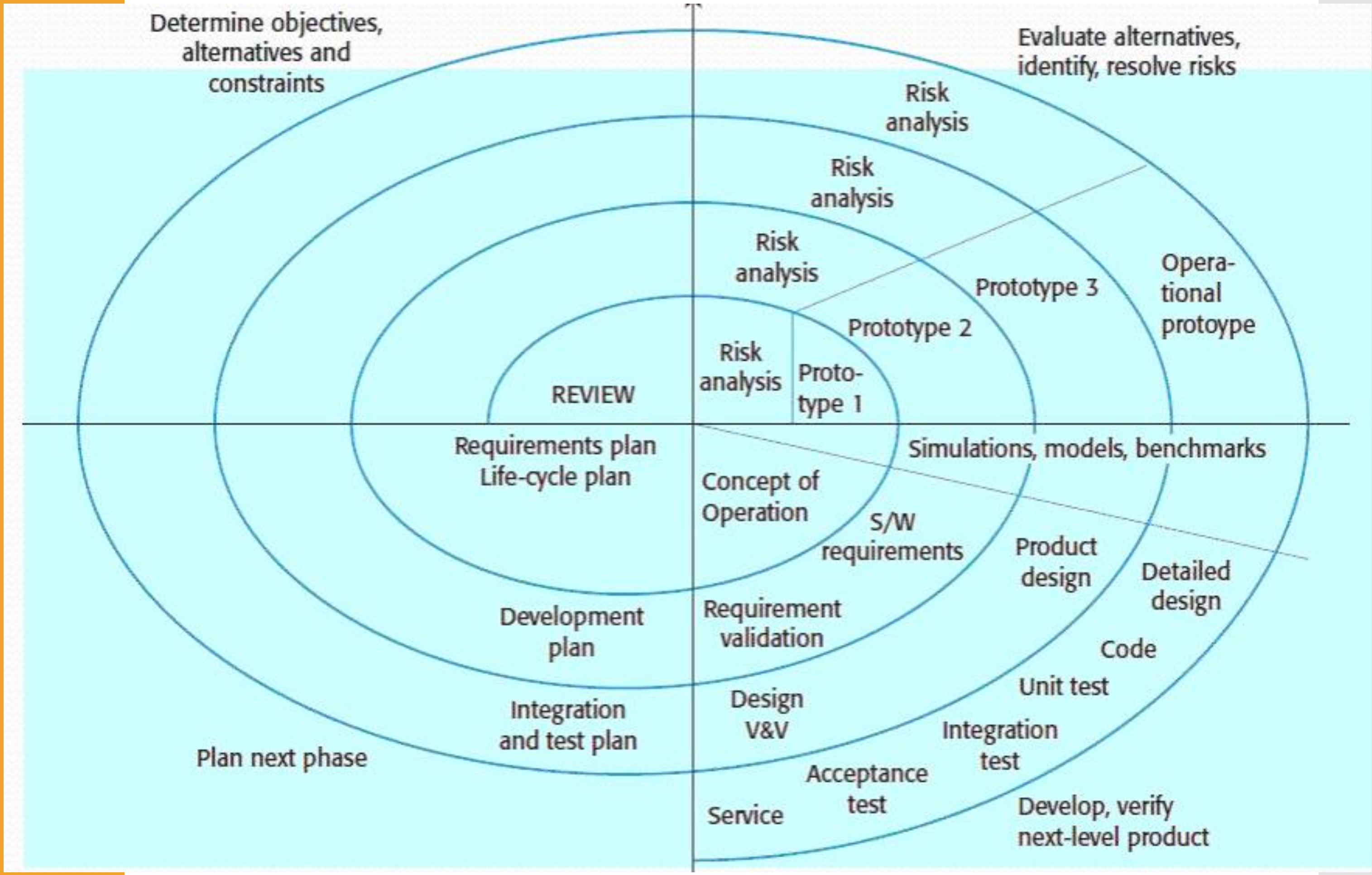
Problems

- Lack of process visibility
- Systems are often poorly structured

Applicability

- For small or medium-size interactive systems
- For parts of large systems
- For short-lifetime systems

Iteration Model



Agile Model

Definition

The requirements and solutions evolve through collaboration between self-organizing, cross-functional teams.

Break tasks into small increments with minimal planning and do not directly involve long-term planning.

Iterations are short time frames (time boxes) that typically last from one to four weeks.

Agile

- **Each iteration involves a *cross functional* team working in all functions:**
  - Planning, and requirements analysis
  - Design, and coding
  - unit testing and acceptance testing
- **At the end of the iteration a working product is demonstrated to stakeholders.**

Characteristics of Agile Mode

Iterative & Incremental

Test Driven Development

People Oriented

Ligtweight

UNIFIED PROCESSING

Unified Processing

What is an Unified Processing?

A process model that was created 1997 to give a framework for Object-oriented Software Engineering

Iterative, incremental model to adapt to specific project needs

The Unified Process is an adaptable methodology

The Unified Process is a modeling technique

Unified Processing

What are Characteristics of the Unified Process?

- Object Oriented
- Use Case Driven
- Architecture Centric
- Iteration & Increment

- Utilizes object oriented technologies.
- Classes are extracted during OOA and designed during OOD.

Focus core architecture in the early iterations

In earliest iterations, get high valued requirements

View of the whole design with the important characteristics made morevisible

Expressed with classdiagram

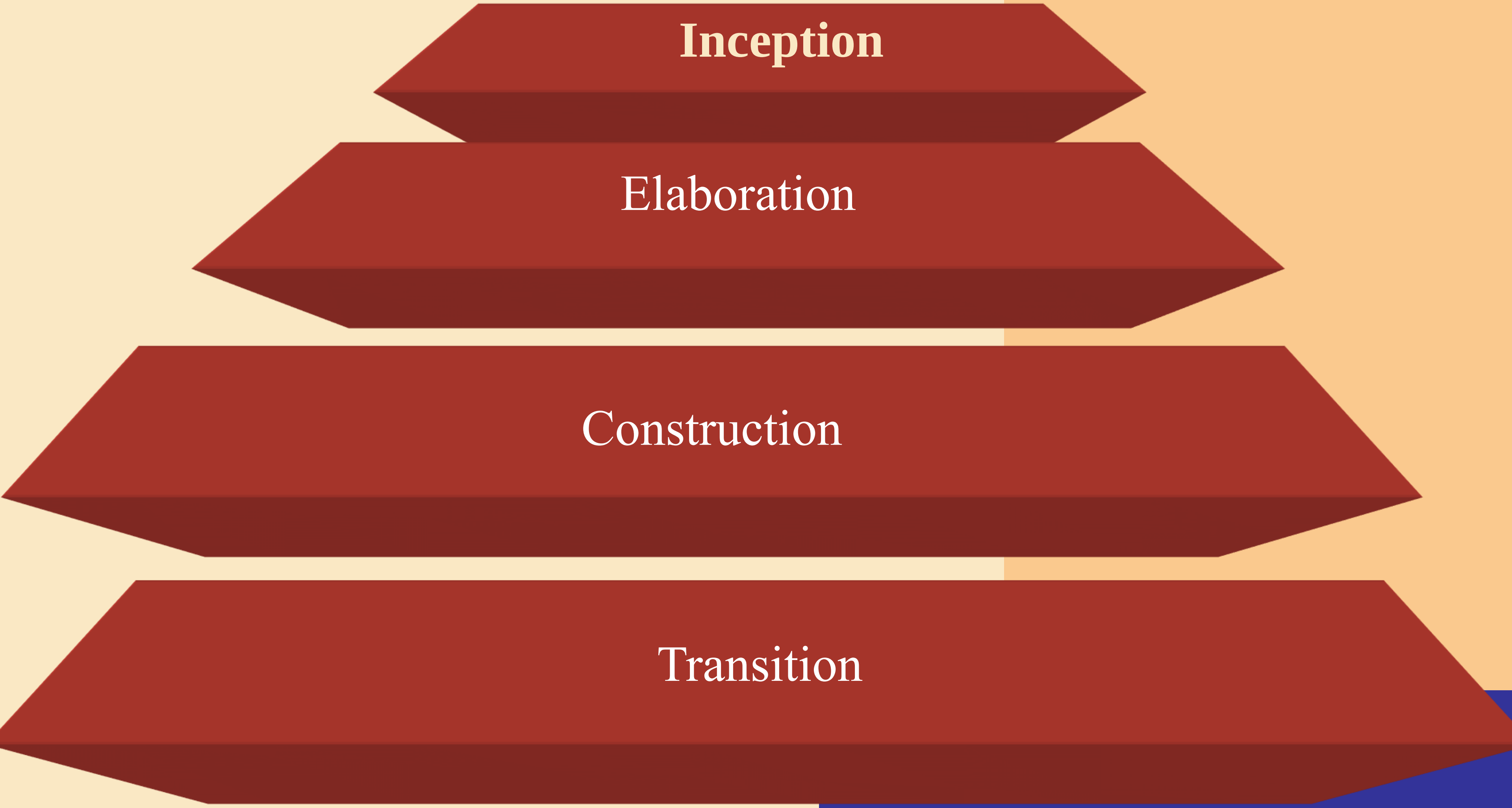
- Utilizes use case model to describe complete functionality of the system

•Iterations are time boxed (i.e. fixed in length)

• Best iteration length (2-6 weeks)

Unified Processing

What are the phases of UP?



Inception

- A vision of the product is created. Questions discussed are:
- What is the product supposed to do?
- Why should my organization embark on a project to build this particular product?
- Does my organization have the resources to build this product?
- Is it feasible to do so?
- How much will this product cost and how much will it bring in?
- What will be the duration of the project?
- Risk analysis is performed.
- Decision whether to go ahead with the project or not is taken.

Elaboration

- System to be built is analyzed in detail.
- Use cases used to document the requirements. Main aim: Get the core architecture and as many use cases as possible.
- The core architecture is coded, verified with user and baselined.
- Other high-risk requirements are identified and coded.
- A project plan is drawn in this phase, resources are allocated and a schedule is planned.
- UML diagrams are used to model the system under design.

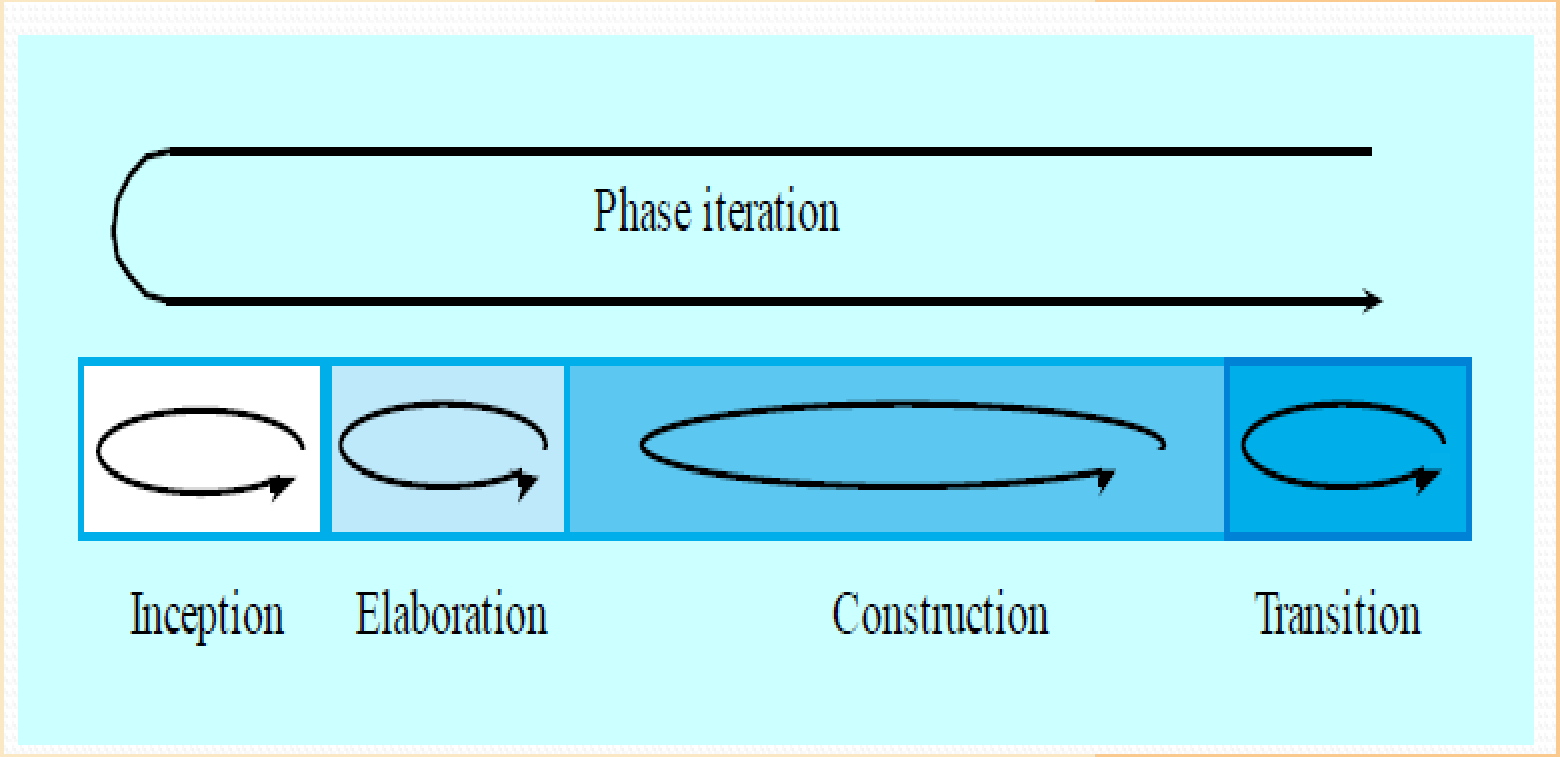
Construction

- Remaining use cases are implemented.
- If any new use cases are discovered, they are implemented.
- Test cases are written, actual tests are carried out and a test report is prepared.
- Documentation for the system as well as guides for the users are written.

Transition

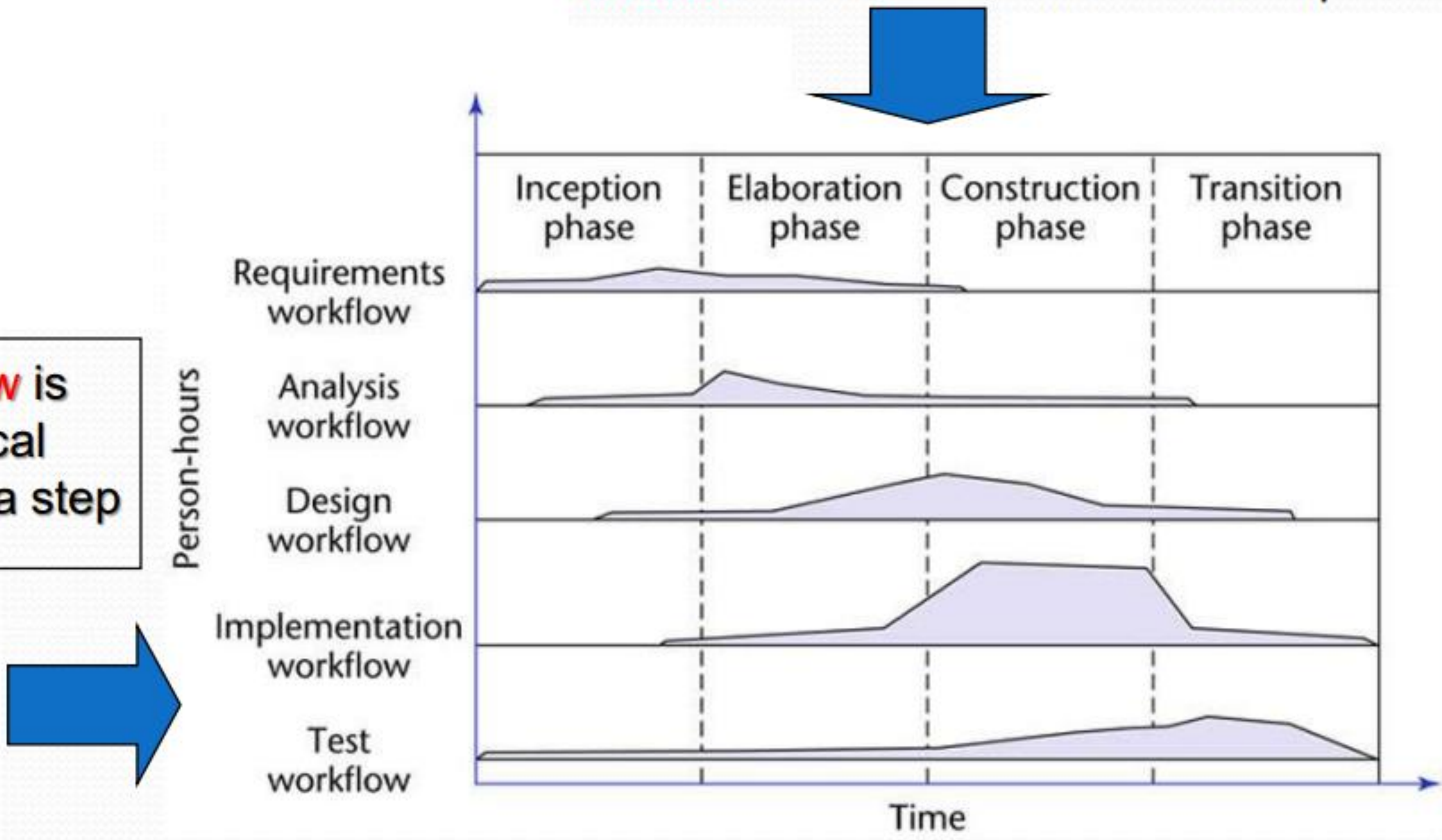
- System is installed in its environment and beta-tested.
- Feedback is received and System is refined and tuned to adapt in response to the feedback.
- It also includes activities like marketing of the product and training of users.

Unified Process Model



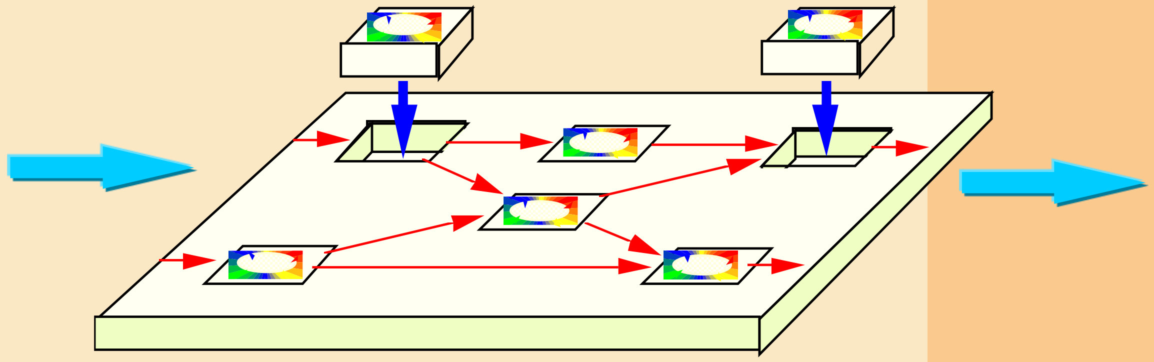
- Phase is Business context of a step

Workflow is Technical context of a step



Design Activities in the UP Life Cycle

The Unified Process is a Process Framework



There is NO Universal Process!

- The Unified Process is designed for flexibility and extensibility
  - » allows a variety of lifecycle strategies
  - » selects what artifacts to produce
  - » defines activities and workers
  - » models concepts

Repeated verification of quality

- The UP stresses on the repeated verification of quality.
- Here, quality refers to both, the quality of the software to be developed as well as the quality of the process itself.
- At the end of each iteration the team must produce executable software.
- This achievement indicates the success of the iteration.
- The software is tested and verified right from the start unlike the traditional SDLC where Quality Analysis is done basically at the end.

Phase Deliverables

Inception Phase	Elaboration Phase	Construction Phase	Transition Phase
• The initial version of the domain model • The initial version of the business model • The initial version of the requirements artifacts • A preliminary version of the analysis artifacts • A preliminary version of the architecture • The initial list of risks • The initial ordering of the use cases • The plan for the elaboration phase • The initial version of the business case	• The completed domain model • The completed business model • The completed requirements artifacts • The completed analysis artifacts • An updated version of the architecture • An updated list of risks • The project management plan (for the rest of the project) • The completed business case	• The initial user manual and other manuals, as appropriate • All the artifacts (beta release versions) • The completed architecture • The updated risk list • The project management plan (for the remainder of the project) • If necessary, the updated business case	• All the artifacts (final versions) • The completed manuals

Examples Role in UP

- Stake Holder
  - Customer, product manager, etc.
- Software Architect
  - Establish and maintain architectural vision
- Process Engineer
  - Leads definition and refinement of development use cases
- Graphics Artist
  - Assists in user interface design

Some UP guidelines

- *Attack risks early on* and continuously so, before they will attack you
- Stay focused on *developing executable software* in early iterations
- Prefer *component-oriented* architectures and *reuse existing components*
- Quality is a way of life, not an afterthought

Six best “must” UP practices

1. Time-boxed iterations: *avoid speculative powerpoint architectures”*

2. *Strive for cohesive architecture* and reuse existing components:

- e.g. core architecture developed by small, co-located team
- then early team members divide into sub-project leaders

Six best "must" UP practices

3. Continuously verify quality: *test early & often*, and realistically by integrating all software at each iteration
4. *Visual modeling*: prior to programming, do at least some visual modeling to explore creative design ideas
5. *Manage requirements:* find, organize, and track requirements through skillful means

6. *Manage change:*
- disciplined configuration management protocol, version control,
- change request protocol
- baselined releases at iteration ends

WORKFLOWS

The Unified Process

Two Types

Phases

Describe the business steps needed to develop, buy, and pay for software development.

The business increments are identified as phases

WorkFlow

Describe the tasks or activities that a developer performs to evolve an information system over time

WorkFlows

Workflow	**Description**
Business modelling	The business processes are modelled using business use cases.
Requirements	Actors who interact with the system are identified and use cases are developed to model the system requirements.
Analysis and design	A design model is created and documented using architectural models, component models, object models and sequence models.
Implementation	The components in the system are implemented and structured into implementation sub-systems. Automatic code generation from design models helps accelerate this process.
Test	Testing is an iterative process that is carried out in conjunction with implementation. System testing follows the completion of the implementation.
Deployment	A product release is created, distributed to users and installed in their workplace.
Configuration and change management	This supporting workflow managed changes to the system
Project management	This supporting workflow manages the system development
Environment	This workflow is concerned with making appropriate software tools available to the software development team.

Primary Workflows

- The Unified Process

- PRIMARY WORKFLOWS
  - Requirements workflow
  - Analysis workflow
  - Design workflow
  - Implementation workflow
  - Test workflow

- Supplemental Workflows
  - Planning Workflow

Planning Workflow

- Define scope of Project
- Define scope of next iteration
- Identify Stakeholders
- Capture Stakeholders expectation
- Build team
- Assess Risks
- Plan work for the iteration
- Plan work for Project
- Develop Criteria for iteration/project closure/success
- UML concepts used: initial Business Model, using class diagram

Requirements Workflow

- The aim is to determine the client's needs
- First, gain an understanding of the *domain* and build domain model describing important concepts of the context
- Second, build a business model
  - Use UML to describe the client's business processes
  - If at any time the client does not feel that the cost is justified, development terminates immediately
- It is vital to determine the client's constraints
  - Deadline -- Nowadays software products are often mission critical
  - Parallel running

Analysis Workflow

- The aim of the analysis workflow To analyze and refine the requirements
- Two separate workflows are needed
  - The requirements artifacts must be expressed in the language of the client
  - The analysis artifacts must be precise, and complete enough for the designers
- Specification document ("specifications") Constitutes a contract. It must not have imprecise phrases like "optimal," or "98 percent complete"
- Having complete and correct specifications is essential for
  - Testing, and
  - Maintenance
- The specification document must not have
  - Contradictions
  - Omissions
  - Incompleteness

Design Workflow

- **The goal is to refine the analysis workflow until the material is in a form that can be implemented by the programmers**
  - Many nonfunctional requirements need to be finalized at this time, including: Choice of programming language, Reuse issues, Portability issues.
- **Classical Design**
- **Architectural design**
  - Decompose the product into modules
- **Detailed design**
  - Design each module using data structures and algorithms

Object Oriented Design

- **Classes are extracted during the object-oriented analysis workflow, and**
  - Designed during the design workflow

Implementation Workflow

- The aim of the implementation workflow is to implement the target software product in the selected implementation language
- Distribute the system by mapping executable components onto nodes in the deployment model
- Implement Design Classes and subsystems through packaging mechanism:
- Acquire external components realizing needed interfaces
- Unit test the components
- Integrate via builds

Test Workflow

- Carried out in parallel with other workflows
- Primary purpose
  - To increase the quality of the evolving system
- The test workflow is the responsibility of
  - Every developer and maintainer
  - Quality assurance group
- Develop set of test cases that specify what to test in the system
  - many for each Use Case
  - each test case will verify one scenario of the use case
  - based on Sequence Diagram

Deployment Workflow

- Activities include
  - Software packaging
  - Distribution
  - Installation
  - Beta testing