Unified Software Development Process

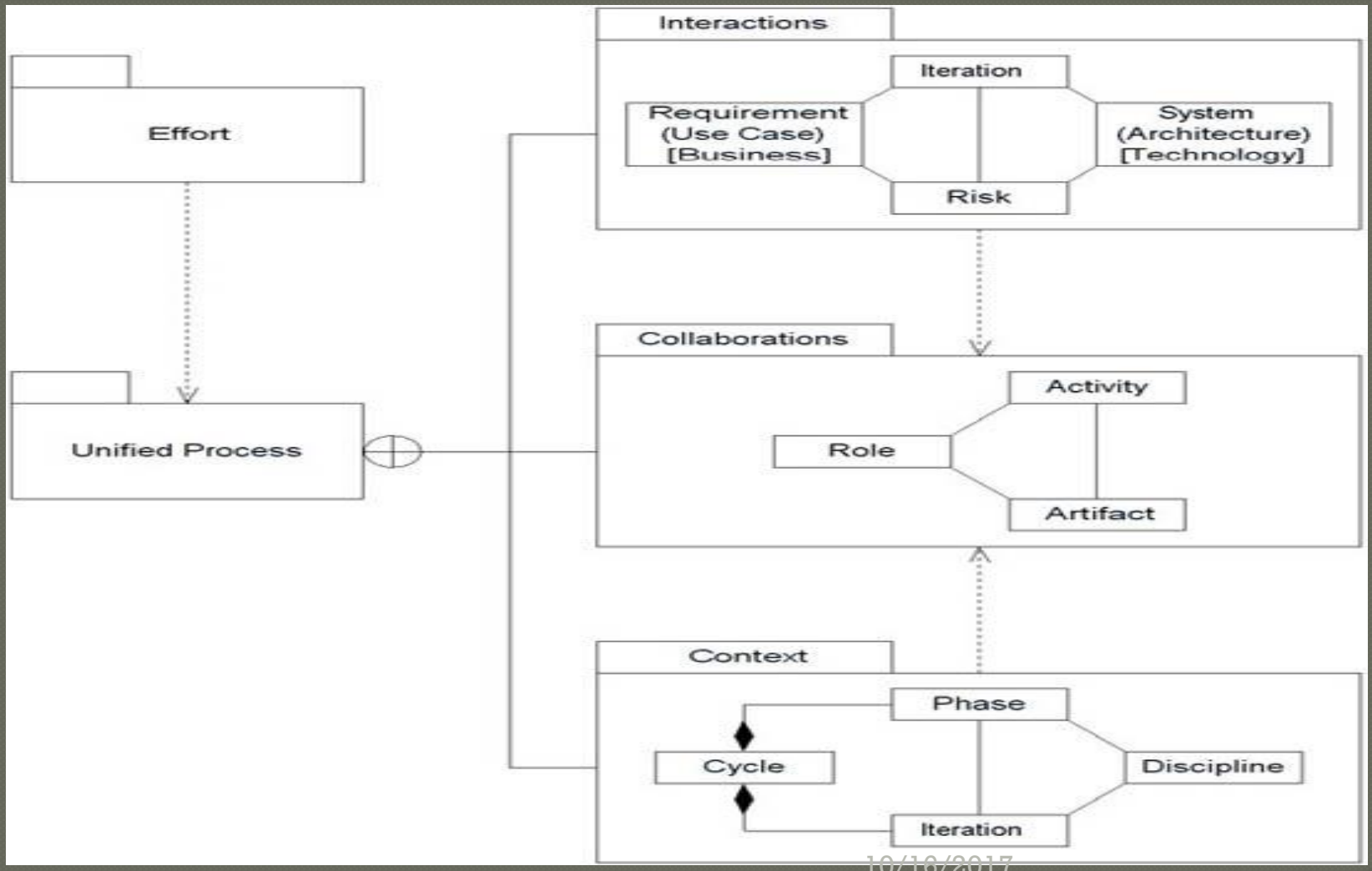
USDP

- The **Unified Software Development Process** or **Unified Process** is a popular iterative and incremental software development process framework.
- The best-known and extensively documented refinement of the Unified Process is the Rational Unified Process (RUP).
- Other examples are OpenUP and Agile Unified Process.
- The Unified Process is not simply a process, but rather an extensible framework which should be customized for specific organizations or projects.
- The *Rational Unified Process* is, similarly, a customizable framework.
- As a result it is often impossible to say whether a refinement of the process was derived from UP or from RUP, and so the names tend to be used interchangeably.

USDP

- The Unified Process (UP) is a use-case-driven, architecture-centric, iterative and incremental development process framework that leverages the Object Management Group's (OMG) UML and is compliant with the OMG's SPEM.
- The process consists of four main phases: ***inception***, ***elaboration***, ***construction*** and ***transition***.
- Each of these contains one or more iterations across five core workflows: ***requirements***, ***analysis***, ***design***, ***implementation*** and ***test***.
- The UP is broadly applicable to different types of software systems , including small-scale and large-scale projects having various degrees of managerial and technical complexity, across different application domains and organizational cultures.
- The UP is an "idea," a process framework that provides an infrastructure for executing projects but not all of the details required for executing projects; essentially, it is a software development process framework, a lifecycle model involving context, collaborations, and interactions.

Elements of UP



Characteristics of UP

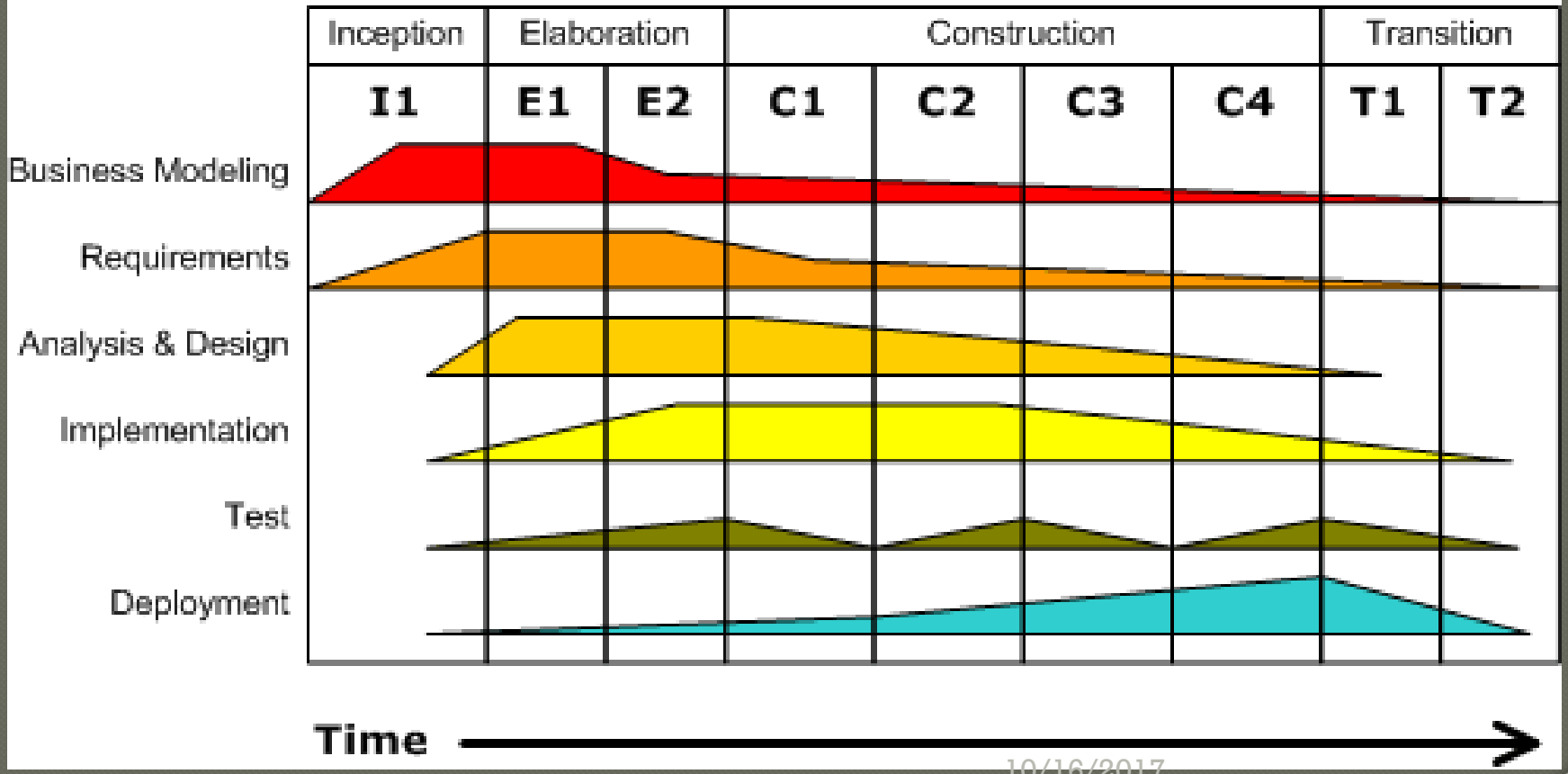
- Iterative and Incremental
  - The Unified Process is an iterative and incremental development process.
  - The Elaboration, Construction and Transition phases are divided into a series of time boxed iterations.
  - The Inception phase may also be divided into iterations for a large project.
  - Each iteration results in an *increment*, which is a release of the system that contains added or improved functionality compared with the previous release.
  - Although most iterations will include work in most of the process disciplines (*e.g.* Requirements, Design, Implementation, Testing) the relative effort and emphasis will change over the course of the project.

Characteristics of UP

- Architecture Centric
  - The Unified Process insists that architecture sit at the heart of the project team's efforts to shape the system.
  - Since no single model is sufficient to cover all aspects of a system, the Unified Process supports multiple architectural models and views.
  - One of the most important deliverables of the process is the executable architecture baseline which is created during the Elaboration phase.
  - This partial implementation of the system serves to validate the architecture and act as a foundation for remaining development.
- Risk focused
  - The Unified Process requires the project team to focus on addressing the most critical risks early in the project life cycle.
  - The deliverables of each iteration, especially in the Elaboration phase, must be selected in order to ensure that the greatest risks are addressed first.

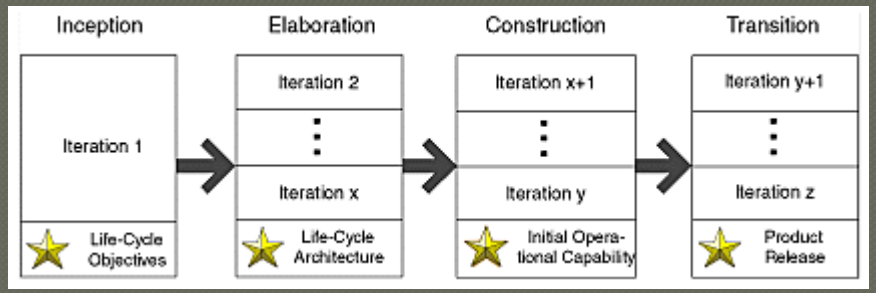
Iterative Development

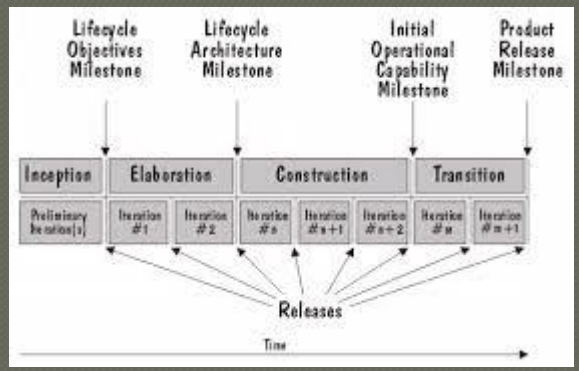
Business value is delivered incrementally in time-boxed cross-discipline iterations.



Life cycle of UP

- The lifecycle describes the time dimension of project, that is, how a project is divided into phases and iterations.
- It divides a project into four **phases: Inception, Elaboration, Construction,** and **Transition,** each ending with a well-defined milestone.
- Each phase has specific objectives.





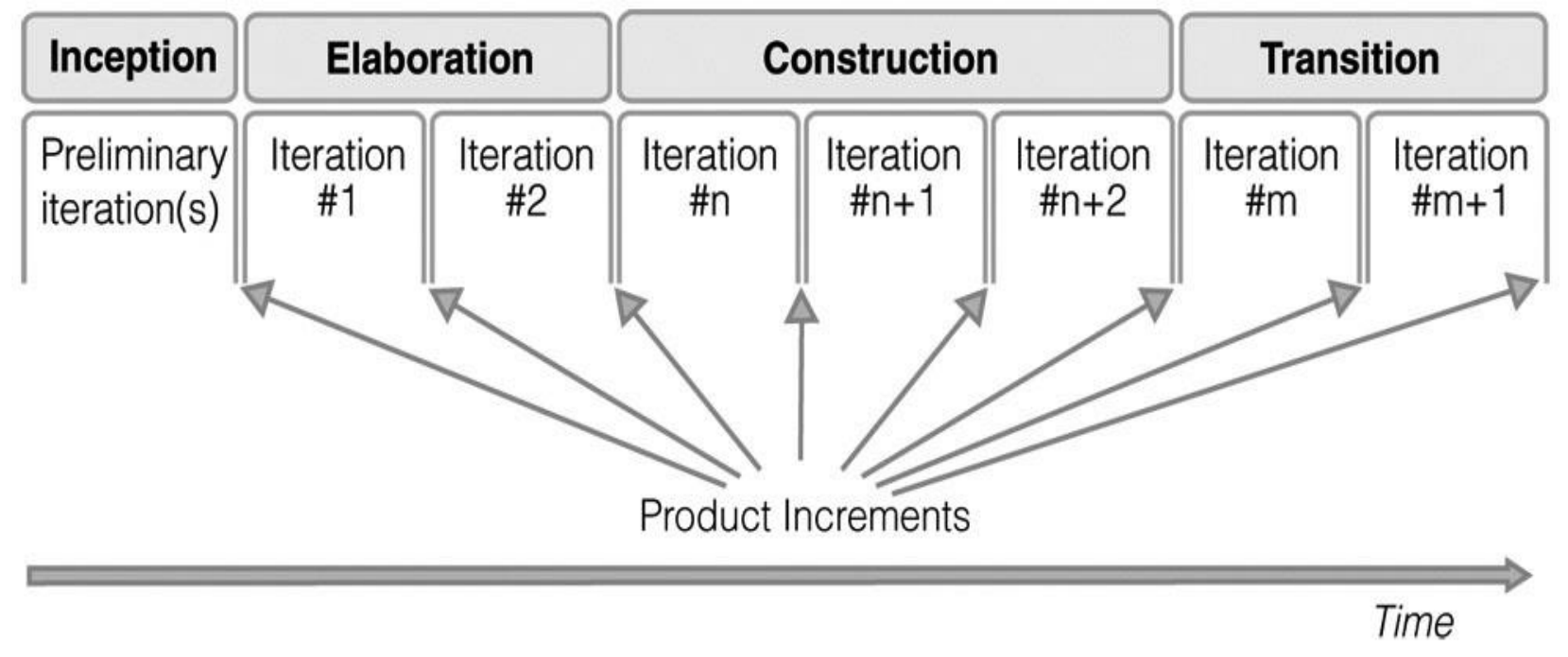
Life cycle of UP

- Inception:
  - Establish the scope of the system, including a good understanding of what system to build, by reaching a high-level understanding of the requirements.
  - Mitigate many of the business risks and produce the business case for building the system and a vision document to get buy-in from all stakeholders on whether or not to proceed with the project.
  - This is similar to what many agile processes refer to as *Iteration 0*.
- Elaboration:
  - Reduce major risks to enable cost and schedule estimates to be updated and to get buy-in from key stakeholders.
  - Mitigate major technical risks by taking care of many of the most technically difficult tasks.
  - Design, implement, test, and baseline an executable **architecture,** including subsystems, their interfaces, key components, and architectural mechanisms such as how to deal with interprocess communication or persistency.
  - Address major business risks by defining, designing, implementing, and testing key capabilities, which are validated with the customer.
  - Do not define and analyze all requirements at this time, as doing so would lead to waterfall development.
  - Detail and analyze only the requirements required to address the above risks.

Life cycle of UP

- Construction:
  - Undertake a majority of the implementation as you move from an executable architecture to the first operational version of your system.
  - Deploy several internal and alpha releases to ensure that the system is usable and addresses user needs.
  - End the phase by deploying a fully functional beta version of the system, including installation and supporting documentation and training material (although the system will likely still require fine-tuning of functionality, performance, and overall quality).
- Transition:
  - Ensure that software addresses the needs of its users by testing the product in preparation for release and making minor adjustments based on user feedback.
  - At this point in the lifecycle, user feedback focuses mainly on fine-tuning, configuration, installation, and usability issues; all the major structural issues should have been worked out much earlier in the project lifecycle.

Life cycle of UP

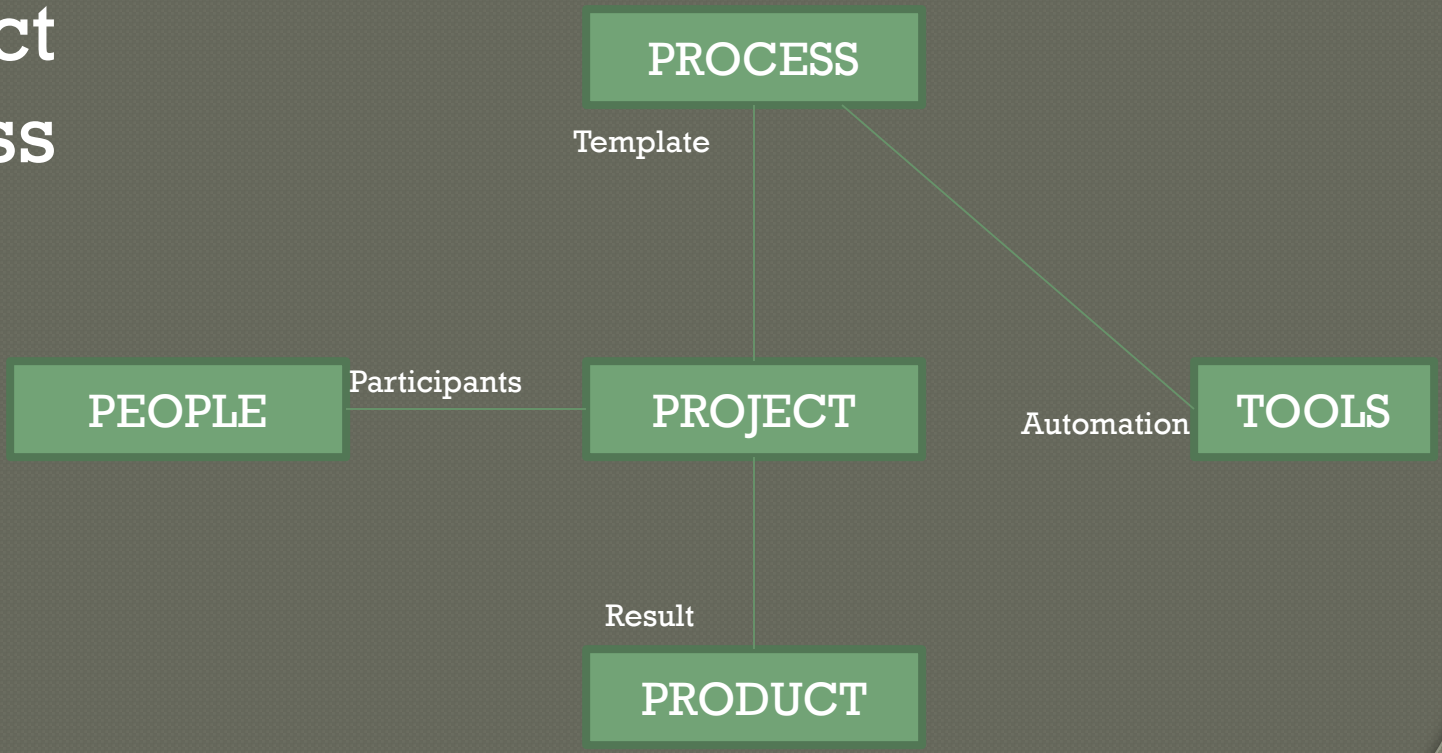


Life cycle of UP

- Each phase contains one or more **iterations** which focus on producing a product increment, that is, the working code and other deliverables necessary to achieve the business objectives of that phase.
- There are as many iterations as it takes to adequately address the objectives of that phase, but *no more*.
- If objectives cannot be adequately addressed within the planned phase, another iteration should be added to the phase, but this will delay the project.
- To avoid such a delay, make sure that each iteration is sharply focused on *just* what is needed to achieve the business objectives of that phase, but no less.
- For example, focusing too heavily on requirements in Inception is counterproductive; so is not involving stakeholders.
- Each of the four phases in the Unified Process lifecycle consists of one or several iterations.
- Each iteration builds on the result of previous iterations, delivering a product increment one step closer to the final release.
- Product increments should include working software, with a possible exception being the product increments produced in the Inception phase for new applications.

Four *P*s in software development

- People
- Project
- Product
- Process



Four *P*s in software development

- People:
  - The architects, developers, testers, and their supporting management, plus users, customers, and other stakeholders are the prime movers in a software project.
- Project:
  - The organizational element through which software development is managed.
  - Outcome of project is a released product.
- Product:
  - Artifacts that are created during the life of the project such as models, source code, executables and documentations
- Process:
  - Definition of complete set of activities needed to transform user's requirements into a product.
  - Is a template for creating projects.
- Tools:
  - Software that is used to automate the activities defined in process.

Four *P*s in software development

- People:
  - Throughout the entire life cycle of software development, people are involved in one or another way.
  - So the process that guides this development must be people oriented.
    - Development process affect people
      - project feasibility
      - Risk management
      - Team structure
      - Project schedule
      - Project understandability
      - Sense of accomplishment
    - Roles will change
      - Uniform development process enables developers to build better software in terms of time to market, quality and cost, also to select an suitable architecture.
      - Here, developers will find themselves working with many other developers.
      - Providing guidance will result in developers "working smarter".
      - To develop more effective software product, choosing right people make them effective and allow them to do what human can.

Four *P*s in software development

- Project:
  - Results in a new release of a product.
  - First project in life cycle develops and release initial system.
  - Successive project cycles extend the life of the system over many releases.
  - Throughout life cycle, a project team has to be concerned with change , iterations and the organizational pattern with in which the project is conducted.
    - A sequence of change
      - Every cycle, phase and every iteration changes the system from one thing to something else.
      - Each cycle leads to a ***release*** and changes continues for ***generations***.
    - A series of iterations
      - With in each phase of cycle, series of iterations of activities are carried.
      - Each iteration implements set of use case or mitigates some risks.
      - An iteration can be thought as ***miniproject.***
    - An organizational pattern
      - There is a pattern within which people executes a project.
      - The pattern indicates the type of workers the project needs and the artifacts with which it is to work.

Four *P*s in software development

- Product:
  - Is whole software system, not just code.
  - Software can be described as a binary form that can be read and understood by a computer.
  - It is a description that is written by programmers that can be read and understood by a compiler.
  - Software design in terms of subsystems, classes, interaction diagrams and other artifacts and also requirements, testing, sales etc can be viewed as part of system.
  - So, a system is all the artifacts that it take to represent it in machine or human readable form to the machines, the workers, and the stakeholders.

Four *P*s in software development

- Product:
  - Artifacts is a general term for any kind of information created , produced , changed or used by workers in developing system.
  - E.g. UML diagrams, prototypes, components, test plans and test procedures.
  - Two types:
    - Engineering artifacts
      - Created during various phases of process.
    - Management artifacts
      - Have a short lifetime.
      - Like business case, development plans etc.
      - Also include specifications of the development environment.

Four *P*s in software development

- Product:
  - Model is one of artifacts employed in Unified Process.
  - Building a system is thus a process of model building using different models to describe all the different perspective of the system.
  - UP provides a carefully selected set of models with which to start to satisfy those worker's need for information.
  - Model is an abstraction of a system that the architects and developers build.
  - Each model is a self contained view of the system.

Four *P*s in software development

- Product:
  - Most engineering models are defined by a carefully selected subset of UML.
  - Like use case model consisting of use cases and actors.
  - There is a hierarchy of elements in model.
  - The system is not just collection of models but also the relationships between them as well.
  - Trace relationship between elements in different models add no semantic information to help understand the related model themselves, they just connect the models.

Four *P*s in software development

- Process:
  - In UP process refers to a concept that works as a template that can be reused by creating instances of it.
  - Is compare to class in object oriented paradigm from which objects can be created.
  - Here, process instance represents *project*
  - *Software development process is a definition of complete set of activities needed to transform user's requirements into a consistent set of artifacts that represents a software product and later to transform changes in those requirements into a new, consistent set of artifacts.*
  - The value added result of the process is a consistent set of artifacts.
  - A process is a definition of a set of activities ,not their execution.

Four *P*s in software development

- Process:
  - We can describe a process in terms of workflows, where a workflow is a set of activities.
  - In UML, a workflow is a stereotype collaboration in which workers and artifacts are participants.
  - E.g. requirement workflow
    - Here, workers are: system analyst, architect , use case specifier and user interface designer .
    - Artifacts are: use case models, use cases and others.
  - No single software development process can be applied everywhere.
  - Unified process is a generic process that is process framework.
  - UP may be specialized to fit different applications and organizational needs.

Four *P*s in software development

- Process:
  - It is also desirable with in organization, the process be fairly consistent.
  - This consistency will allow components to be used interchangeably, people and managers to transfer between projects readily and accomplishment metrics to be comparable.
  - Factors that influence how process will differ:
    - *Organizational factors*
    - *Domain factors*
    - *Life cycle factors*
    - *Technical factors*

Four *P*s in software development

- Process:
  - Benefits:
    - Everyone on the development team can understand what to do.
    - Developers can better understand what other developers are doing.
    - Supervisors and Managers even those who cannot read code, can, thanks to architectural drawing, understand what developers are doing.
    - Developers , supervisors and managers can transfer between projects or divisions without having learn a new process.
    - Training can be standardized within a company.
    - The course of software development is repeatable.

Four *P*s in software development

- Tools:
  - Tools are integral to the process.
  - Are good at automating repetitive tasks, keeping things structured, managing large amount of information and guiding along a particular development path.
  - The tools that implement an automated process should be easy to use.
  - There has to be balance between process and tools.
  - On one hand ,process drives tool development.
  - On the other hand, tools guide process development.
  - *Successful development of process automation (tools) cannot be achieved without the parallel development of the process framework in which the tools are to function.*

Business Modeling

- **Business process modeling** (**BPM**) in systems engineering is the activity of representing processes of an enterprise, so that the current process may be analyzed or improved.
- BPM is typically performed by business analysts, who provide expertise in the modeling discipline; by subject matter experts, who have specialized knowledge of the processes being modeled; or more commonly by a team comprising both.
- Alternatively, the process model can be derived directly from events' logs using process mining tools.
- The business objective is often to increase process speed or reduce cycle time; to increase quality; or to reduce costs, such as labor, materials, scrap, or capital costs.
- In practice, a management decision to invest in business process modeling is often motivated by the need to document requirements for an information technology project.
- In the field of software engineering, the term 'business process modeling' opposed the common software process modeling, aiming to focus more on the state of the practice during software development.
- In that time (early 1990s) all existing and new modeling techniques to illustrate business processes were consolidated as 'business process modeling languages.
- In the Object Oriented approach, it was considered to be an essential step in the specification of business application systems.

Business Modeling

- Business process modeling became the base of new methodologies, for instance those that supported data collection, data flow analysis, process flow diagrams and reporting facilities.
- A business model is a framework for creating economic, social, and/or other forms of value.
- The term 'business model' is thus used for a broad range of informal and formal descriptions to represent core aspects of a business, including purpose, offerings, strategies, infrastructure, organizational structures, trading practices, and operational processes and policies.
- In the most basic sense, a business model is the method of doing business by which a company can sustain itself. That is, generate revenue.
- The business model spells-out how a company makes money by specifying where it is positioned in the value chain.

Business Modeling

- A business process is a collection of related, structured activities or tasks that produce a specific service or product (serve a particular goal) for a particular customer or customers.
- There are three main types of business processes:
  - Management processes, that govern the operation of a system.
    - Typical management processes include corporate governance and strategic management
  - Operational processes, that constitute the core business and create the primary value stream.
    - Typical operational processes are purchasing, manufacturing, marketing, and sales.
  - Supporting processes, that support the core processes.
    - Examples include accounting, recruitment, and technical support.

Business Modeling

- A business process can be decomposed into several sub-processes, which have their own attributes, but also contribute to achieving the goal of the super-process.
- The analysis of business processes typically includes the mapping of processes and sub-processes down to activity level.
- A business process model is a model of one or more business processes, and defines the ways in which operations are carried out to accomplish the intended objectives of an organization.
- Such a model remains an abstraction and depends on the intended use of the model.
- It can describe the workflow or the integration between business processes.
- It can be constructed in multiple levels.
- A workflow is a depiction of a sequence of operations, declared as work of a person, of a simple or complex mechanism, of a group of persons, of an organization of staff, or of machines.
- Workflow may be seen as any abstraction of real work, segregated into workshare, work split or other types of ordering.
- For control purposes, workflow may be a view of real work under a chosen aspect.

Business Modeling

- The artifact-centric business process model has emerged as a holistic approach for modeling business processes, as it provides a highly flexible solution to capture operational specifications of business processes.
- It particularly focuses on describing the data of business processes, known as "artifacts", by characterizing business-relevant data objects, their lifecycles, and related services.
- The artifact-centric process modelling approach fosters the automation of the business operations and supports the flexibility of the workflow enactment and evolution.
- Business process modeling tools provide business users with the ability to model their business processes, implement and execute those models, and refine the models based on as-executed data.
- As a result, business process modeling tools can provide transparency into business processes, as well as the centralization of corporate business process models and execution metrics.

Business Modeling

- A business model, which may be considered an elaboration of a business process model, typically shows business data and business organizations as well as business processes.
- By showing business processes and their information flows a business model allows business stakeholders to define, understand, and validate their business enterprise.
- The data model part of the business model shows how business information is stored, which is useful for developing software code.
- Usually a business model is created after conducting an interview, which is part of the business analysis process.
- The interview consists of a facilitator asking a series of questions to extract information about the subject business process.
- The interviewer is referred to as a facilitator to emphasize that it is the participants, not the facilitator, who provide the business process information. Although the facilitator should have some knowledge of the subject business process, but this is not as important as the mastery of a pragmatic and rigorous method interviewing business experts.

Business Modeling

- The method is important because for most enterprises a team of facilitators is needed to collect information across the enterprise, and the findings of all the interviewers must be compiled and integrated once completed.
- Business models are developed as defining either the current state of the process, in which case the final product is called the "as is" snapshot model, or a concept of what the process should become, resulting in a "to be" model. By comparing and contrasting "as is" and "to be" models the business analysts can determine if the existing business processes and information systems are sound and only need minor modifications, or if reengineering is required to correct problems or improve efficiency.

Requirement analysis

- It is a statement that identifies a necessary attribute, capability, characteristic, or quality of a system for it to have value and utility to a customer, organization, internal user, or other stakeholder.
- A requirement specification (often imprecisely referred to as the **spec**, because there are different sorts of specifications) refers to an explicit *set of* requirements *to be satisfied* by a material, design, product, or service.
- In the classical engineering approach, sets of requirements are used as inputs into the design stages of product development.
- Requirements are also an important input into the verification process, since tests should trace back to specific requirements.
- Requirements show what elements and functions are necessary for the particular project.
- However, when iterative methods of software development or agile methods are used, the system requirements are incrementally developed in parallel with design and implementation.
- **Product requirements** prescribe properties of a system or product.
- **Process requirements** prescribe activities to be performed by the developing organization.
  - For instance, process requirements could specify the methodologies that must be followed, and constraints that the organization must obey.

Requirement analysis

- Types of requirements:
  - Architectural requirements:
    - Architectural requirements explain what has to be done by identifying the necessary systems structure and systems behavior, i.e., systems architecture of a system.
  - Business requirements:
    - High-level statements of the goals, objectives, or needs of an organization. They usually describe opportunities that an organization wants to realise or problems that they want to solve. Often stated in a business case.
  - User (stakeholder) requirements:
    - Mid-level statements of the needs of a particular stakeholder or group of stakeholders. They usually describe how someone wants to interact with the intended solution. Often acting as a mid-point between the high-level business requirements and more detailed solution requirements.

Requirement analysis

- Types of requirements:
  - Functional (solution) requirements:
    - Usually detailed statements of capabilities, behaviour, and information that the solution will need. Examples include formatting text, calculating a number, modulating a signal. They are also known as *capabilities*.
  - Quality-of-service (non-functional) requirements
    - Usually detailed statements of the conditions under which the solution must remain effective, qualities that the solution must have, or constraints within which it must operate. Examples include: reliability, testability, maintainability, availability. They are also known as *characteristics*, *constraints.*
  - Implementation (transition) requirements
    - Usually detailed statements of capabilities or behaviour required only to enable transition from the current state of the enterprise to the desired future state, but that will thereafter no longer be required. Examples include: recruitment, role changes, education, migration of data from one system to another.

Requirement analysis

- Characteristics of good requirements:
  - Unitary
  - Complete
  - Consistent
  - Traceable
  - Unambiguous
  - Verifiable
- Requirements analysis or requirements engineering is the set of activities that lead to the derivation of the system or software requirements.
- Requirements engineering may involve a feasibility study or a *conceptual analysis phase* of the project and requirements elicitation (gathering, understanding, reviewing, and articulating the needs of the stakeholders) and requirements analysis , analysis (checking for consistency and completeness), specification (documenting the requirements) and validation (making sure the specified requirements are correct).
- Requirements analysis is critical to the success of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design.

Analysis and Design

- Object-oriented analysis and design (OOAD) is a popular technical approach for analyzing, designing an application, system, or business by applying the object-oriented paradigm and visual modeling throughout the development life cycles to foster better stakeholder communication and product quality.
- OOAD in modern software engineering is best conducted in an iterative and incremental way.
- Iteration by iteration, the outputs of OOAD activities, analysis models for OOA and design models for OOD respectively, will be refined and evolve continuously driven by key factors like risks and business value.
- The primary tasks in object-oriented analysis (OOA) are:
  - Find the objects
  - Organize the objects
  - Describe how the objects interact
  - Define the behavior of the objects
  - Define the internals of the objects

Analysis and Design

- Common models used in OOA are use cases and object models.
- Use cases describe scenarios for standard domain functions that the system must accomplish.
- Object models describe the names, class relations (e.g. Circle is a subclass of Shape), operations, and properties of the main objects.
- User-interface prototypes can also be created to help understanding.
- During object-oriented design (OOD), a developer applies implementation constraints to the conceptual model produced in object-oriented analysis.
- Important topics during OOD also include the design of software architectures by applying architectural patterns and design patterns with object-oriented design principles.
- OOM is a main technique heavily used by both OOA and OOD activities in modern software engineering.

Implementation

- Implementation is the realization of an application, or execution of a plan, idea, model, design, specification, standard, algorithm, or policy.
- an implementation is a realization of a technical specification or algorithm as a program, software component, or other computer system through computer programming and deployment.
- in object-oriented programming, when a concrete class implements an interface; in this case the concrete class is an *implementation* of the interface and it includes methods which are *implementations* of those methods specified by the interface.

Test

- The purposes of testing are:
  - To verify the interaction between objects.
  - To verify the proper integration of all components of the software.
  - To verify that all requirements have been correctly implemented.
  - To identify and ensure defects are addressed prior to the deployment of the software.
- The Rational Unified Process proposes an iterative approach, which means that you test throughout the project.
- This allows you to find defects as early as possible, which radically reduces the cost of fixing the defect.
- Tests are carried out along three quality dimensions reliability, functionality, application performance and system performance.
- For each of these quality dimensions, the process describes how you go through the test lifecycle of planning, design, implementation, execution and evaluation.
- Strategies for when and how to automate test are described. Test automation is especially important using an
- iterative approach, to allow regression testing at then end of each iteration, as well as for each new version of the
- product.

Deployment

- Software deployment is all of the activities that make a software system available for use.
- The general deployment process consists of several interrelated activities with possible transitions between them.
- These activities can occur at the producer side or at the consumer side or both. Because every software system is unique, the precise processes or procedures within each activity can hardly be defined.
- Therefore, "deployment" should be interpreted as a *general process* that has to be customized according to specific requirements or characteristics.

Deployment

- Deployment activities:
  - Release
  - Install and activate
  - Deactivate
  - Adapt
  - Update
  - Version tracking
  - Uninstall

Configuration and change management

- **Configuration management** (**CM**) is a systems engineering process for establishing and maintaining consistency of a product's performance, functional and physical attributes with its requirements, design and operational information throughout its life.
- CM verifies that a system performs as intended, and is identified and documented in sufficient detail to support its projected life cycle.
- CM emphasizes the functional relation between parts, subsystems, and systems for effectively controlling system change.
- It helps to verify that proposed changes are systematically considered to minimize adverse effects.
- CM verifies that changes are carried out as prescribed and that documentation of items and systems reflects their true configuration.
- A complete CM program includes provisions for the storing, tracking, and updating of all system information on a component, subsystem, and system basis.

Configuration and change management

- CM is the practice of handling changes systematically so that a system maintains its integrity over time.
- CM implements the policies, procedures, techniques, and tools that are required to manage, evaluate proposed changes, track the status of changes, and to maintain an inventory of system and support documents as the system changes.
- CM programs and plans provide technical and administrative direction to the development and implementation of the procedures, functions, services, tools, processes, and resources required to successfully develop and support a complex system.
- Change Management focuses on how people and teams are affected by an organizational transition.
- It deals with many different disciplines, from behavioral and social sciences to information technology and business solutions.
- In a project management context, change management may refer to the change control process wherein changes to the scope of a project are formally introduced and approved.

Project Management

- **Project management** is the discipline of initiating, planning, executing, controlling, and closing the work of a team to achieve specific goals and meet specific success criteria.
- A project is a temporary endeavor designed to produce a unique product, service or result with a defined beginning and end (usually time-constrained, and often constrained by funding or deliverables) undertaken to meet unique goals and objectives, typically to bring about beneficial change or added value.
- The primary challenge of project management is to achieve all of the project goals and constraints.

Project Management

- Agile project management encompasses several iterative approaches based on the principles of human interaction management and founded on a process view of human collaboration.
  - It is the most consistent project management technique since it involves frequent testing of the project under development.
  - It is the only technique in which the client will be actively involved in the project development.
  - The only disadvantage with this technique is that it should be used only if the client has enough time to be actively involved in the project.

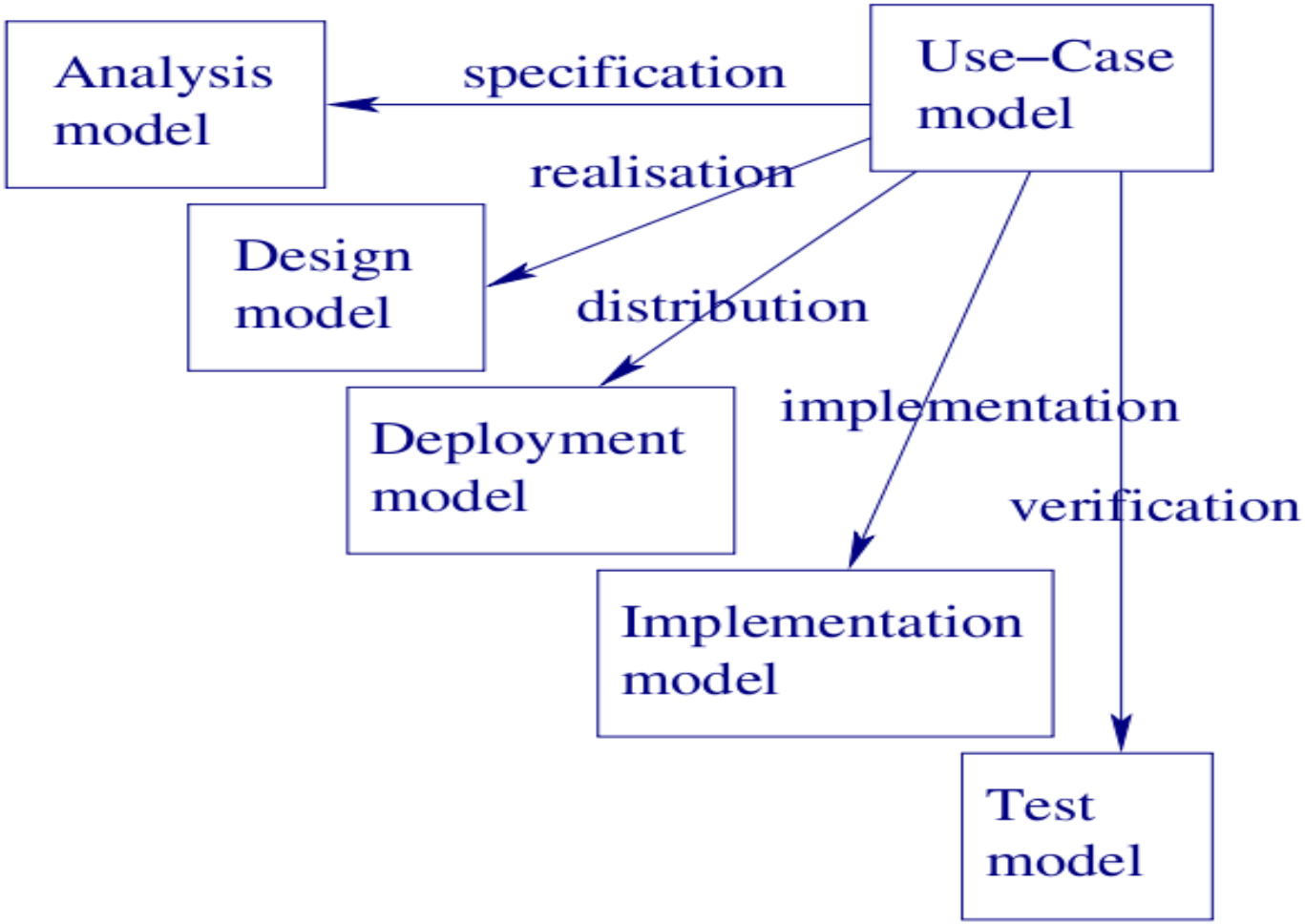
Environment

- Environment discipline refers to the tools and customizing the process for the project - that is, setting up the tool and process environment

Models evolution through Itertions

- Design in Unified Process proceeds through a series of cycles, each of which has following phases:
  - Inception
  - Elaboration
  - Construction
  - Transition
- Within these phases we may go through a number of iterations, each involving the normal forms of workflow activity(requirements , analysis , design , implementation , testing).
- A principal product of the unified process is the series of models, each appropriate to a key stage in system design.
- Since many different models are produced, each for a different design purpose but all related to the same system, we need some point of anchorage.
- This is provided by a use case model.

Models evolution through Itertions



Models evolution through Itertions

- The purpose of use case is to describe the functionality required of the system from the point of view of those concerned with its operation.
- The way a use case does this is by specifying a sequence of actions , including variants, that the system can perform and that yield an observable result of value to some actor.
- In unified process, this drives:
  - Requirement capture
  - Analysis and Design of how system realizes use case
  - Acceptance / system testing
  - Planning of development tasks
  - Traceability of design decisions back to use cases

Models evolution through Itertions

The Unified Software Development Process

- Use-Case Model
  - Use-Case Diagram
- Analysis Model
  - describe "Realization of a Use-Case" by a Collaboration Diagram and a Flow of Event Description
- Design Model
  - Class Diagram , Sequence Diagram, and Statechart Diagram
- Deployment Model
  - Deployment Diagram
- Implementation Model
  - Component Diagram
- Test Model
  - Test Case

Models evolution through Itertions