2. Iterative, Evolutionary, and Agile

Dr. Ziad Kobti
School of Computer Science
University of Windsor

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Objectives

- Motivation
- Iterative Process
- Agile Process
- Unified Process

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Motivation: Iterative and evolutionary

- **Waterfall lifecycle**
  - Big upfront speculative requirements and design steps before programming
  - Highest failure rates for software projects
- **Iterative and evolutionary development**
  - Involves early programming and testing of a partial system, in repeating cycles.
  - Assumes development starts before all the requirements are defined in detail
  - Feedback is used to clarify and improve the evolving specifications
  - Rely on short quick development steps, feedback, and adaptation to clarify the requirements and design
  - Higher success rates for software projects

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Motivation: Unified Process (UP)

- Software Development Process
  - Describes an approach to building, deploying, and possibly maintaining software
- Unified Process (UP)
  - Emerged as a popular *iterative* software development process for building OO systems
  - UP practices provide an example *structure* for how to do OOA/D
  - UP is flexible, and can be applied in a lightweight and *agile* approach that includes practices from other agile methods (such as eXtreme Programming [XP] and Scrum)
- Rational Unified Process (RUP)
  - A detailed refinement of the UP, widely adopted

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Iterative Process

- Iterative Development
  - A key practice in UP and most modern methods
  - In this lifecycle, development is organized into a series of short, fixed-length mini-projects called **iterations**
  - The outcome of each iteration is a tested, integrated, and executable *partial* system
  - Each iteration includes its own requirements analysis, design, implementation, and testing activities

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Iterative Process

- Iterative and Evolutionary (Incremental) Development
  - Lifecycle is based on the successive enlargement and refinement of a system through multiple iterations
  - Early iterative process ideas were known as spiral development and evolutionary development
  - The result of each iteration is an executable but incomplete system; it is not ready to deliver into production
  - The system may not be eligible for production deployment until after many iterations (e.g.10-15)
  - The output of an iteration is NOT an experimental or throw-away prototype, and iterative development is not prototyping. Rather, the output is a production grade subset of the final system.

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Change on an iterative project

- Iterative development is based on an attitude of embracing change and adaptation as unavoidable and indeed essential drivers – rather than trying unsuccessfully to fully and correctly specify, freeze requirements (Water fall model)
- UP – balances need and stability (vs. reactive to feature creep)

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Benefits of Iterative Development

- Less project failure, better productivity, lower defect rates
- Early mitigation of high risks
- Early visible progress
- Early feedback, user engagement, and adaptation, leading to a refined system that more closely meets the real needs of the stakeholders
- Managed complexity
- Methodical learning to improve development one iteration at a time

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Iteration timeboxing

- Timeboxed
  - Fixed in length
  - Date slippage is illegal
  - What cannot be completed should be added to future iteration requirements

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Waterfall Lifecycle

- Waterfall
  - Sequential lifecycle process
  - There is an attempt to define, in detail, all or most of the requirements before programming
  - Research shows that it is a poor practice based on the 1960s and 1970s era.
  - Strong associated with high rates of failure, lower productivity, and higher defect rates
  - On average 45% of the features in waterfall requirements are never used, and early waterfall schedules and estimates vary up to 400% from the final actuals

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Why Waterfall is Failure-Prone

- Assumes specifications are predictable and stable and can be correctly defined at the start, with low change rates.
- Typical software project experienced a 25% change in requirements (Boehm and Papaccio)
- “new project development” domain – software development is on average a domain of high chance of instability

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Feedback and Adaptation needed

- In complex and changing systems feedback and adaptation are key ingredients for success
  - Feedback from early development
  - From tests and developers to refine the design models
  - From the progress of the team tackling early features to refine the schedule and estimates
  - From the client and marketplace to re-prioritize the features to tackle in the next iteration

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

How to do iterative and Evolutionary Analysis and Design

- See example page 25-27
- Common misunderstanding:
  - Extreme thinking that “complete” up-front analysis is skillful

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Risk-driven and Client driven iterative planning

- UP encourages a combination of risk-driven and client-driven iterative planning
- Goals of the early iteration are chosen to:
  - Identify and drive down the highest risks
  - Build visible features that the client cares most about
- Risk driven iterative dev. Includes more specifically the practice of **architecture-centric** iterative dev.
  - Early iterations focus on building, testing, and stabilizing the core architecture.
  - Not having a solid architecture is a common high risk.

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Agile Development

- Agile development methods
  - usually apply timeboxed iterative and evolutionary development,
  - employ adaptive planning,
  - Promote incremental delivery,
  - And include other values and practices that encourage **agility**(rapid and flexible response to change)
- Agile methods share short timeboxed iterations with evolutionary refinement of plans, requirements, and design; they promote practices and principles that reflect an agile sensibility of simplicity, lightness, communication, self-organizing teams…

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Agile Methods

- Example practices (Scrum)
  - Common project work room
  - Self organizing teams
  - XP: programming in pairs and test driven development

UP: “whatever works” attitude

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Agile modeling

- The purpose of modeling is primarily to understand, not to document
- Agile modeling
  - Adopting an agile method does not mean avoiding any modeling
  - The purpose of modeling and models is primarily to support understanding and communication, not documentation
  - Don't model or apply UML to all or most of the software design; defer simple design problems until programming
  - User the simplest tool possible
  - Teams, whiteboard
  - Create models in parallel
  - "good enough" simple notation – stick to simple UML elements
  - Know that all models will be inaccurate
  - Developers should do the OO design modeling

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

Agile UP

- Meant to be adopted and applied in spirit of adaptability and lightness
  - Prefer a small set of UP activities and artifacts
  - Requirements and designs not completed before implementation
  - Apply the uml with agile modeling practices
  - There isn't a detailed plan for the entire project
    - Phase plan – estimates the project end date and milestones
    - Iteration plan – greater detail of one iteration in advance

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved. Not to be duplicated without permission.

UP Phases

- 4 major phases
  - Inception
    - Approximate vision, business case, scope, vague estimates
  - Elaboration
    - Refined vision, iterative implementation of the core architecture, resolution of high risks, identification of most requirements and scope, more realistic estimates
  - Construction
    - Iterative implementation of the remaining lower risk and easier elements, and preparation for deployment
  - Transition
    - beta tests, deployment

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved.
Not to be duplicated without permission.

UP Disciplines

- Disciplines – set of activities in on e subject area, such as the activities within requirements analysis
  - Business modeling
  - Requirements
  - Design
- UP implementation means programming and building the system, not deploying it.

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved.
Not to be duplicated without permission.

Development case

- The choice of practices and UP artifacts for a project may be written up in a short document called the development case

(c) 2005 by Dr. Ziad Kobti. All Rights Reserved.
Not to be duplicated without permission.