Murdoch UNIVERSITY

Approaches to system development

Topic 12

ICT284 Systems Analysis and Design

About this topic

Although all systems development projects pass through some sort of life cycle or SDLC, there are many different approaches that can be taken. In the final topic of the unit, we'll look at *predictive* and *adaptive* approaches to the SDLC, and at some specific system development methodologies and philosophies.

Murdoch UNIVERSITY

Unit learning outcomes addressed in this topic

1. Explain how information systems are used within organisations to fulfil organisational needs
2. **Describe the phases and activities typically involved in the systems development life cycle**
3. Describe the professional roles, skills and ethical issues involved in systems analysis and design work
4. Use a variety of techniques for analysing and defining business problems and opportunities and determining system requirements
5. Model system requirements using UML, including use case diagrams and descriptions, activity diagrams and domain model class diagrams
6. Explain the activities involved in systems design, including designing the system environment, application components, user interfaces, database and software
7. Represent early system design using UML, including sequence diagrams, architectural diagrams and design class diagrams
8. Describe tools and techniques for planning, managing and evaluating systems development projects
9. **Describe the key features of several different systems development methodologies**
10. Present systems analysis and design documentation in an appropriate, consistent and professional manner

Topic learning outcomes

After completing this topic you should be able to:

- Compare the underlying assumptions and uses of a predictive and an adaptive system development life cycle (SDLC)
- Explain what makes up a system development methodology, including the SDLC as well as models, tools, and techniques
- Describe the key features of Agile development
- Understand and briefly describe the key features of the Unified Process, Extreme Programming, and Scrum Agile system development methodologies

Murdoch UNIVERSITY

Resources for this topic

READING

- Satzinger, Jackson & Burd, Chapter 10

  You only need to skim the section on 'The Unified Process, Extreme Programming and Scrum' very briefly.

Except where otherwise referenced, all images in these slides are from those provided with the textbook: Satzinger, J., Jackson, R. and Burd, S. (2016) *Systems Analysis and Design in a Changing World*, 7th edition, Course Technology, Cengage Learning: Boston. ISBN-13 9781305117204

Murdoch UNIVERSITY

Topic outline

- The Systems Development Life Cycle (SDLC)
  - Predictive and Adaptive SDLC
- Methodologies, models, tools and techniques
- Structured and object-oriented approaches to software construction and modelling
- Agile development
- Unified Process, Xtreme Programming, Scrum

The SDLC

Murdoch UNIVERSITY

Introduction

- Topic 1 discussed the SDLC and some approaches to system development
- Later topics focused on Systems Analysis activities and tasks and some System Design activities and tasks
- Now we return to look at the SDLC and related concepts in more detail

  Predictive versus Adaptive SDLC variations

  Models, Methodologies, Tools and Techniques

  Impacts of Traditional versus OO development

  Agile Development

The System Development Life Cycle (SDLC)

Murdoch UNIVERSITY

The SDLC allows us to think about the development of a system as a progressive process

There are two general approaches to the SDLC

- Predictive Approach

  Waterfall model

  - Assumes the project can be planned in advance and that the information system can be developed according to the plan
  - Requirements are well understood and/or low technical risk

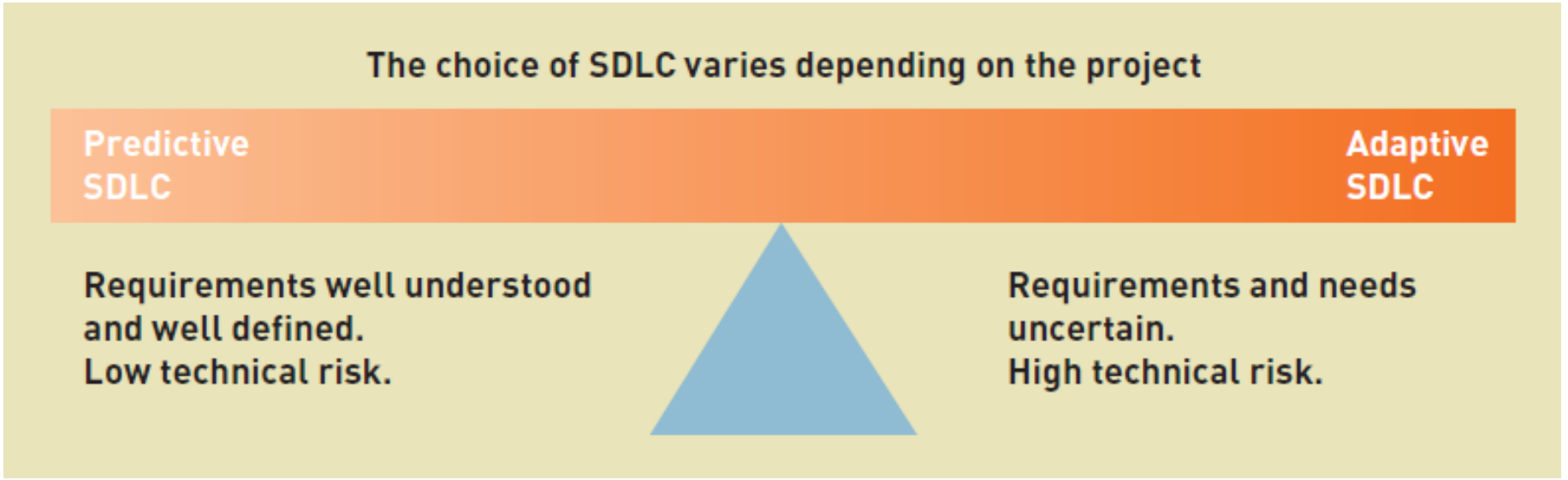
- Adaptive Approach

  Iterative model

  - Assumes the project must be more flexible and adapt to changing needs as the project progresses
  - Requirements and needs are uncertain and/or high technical risk

The SDLC

- Most projects fall on a continuum between Predictive and Adaptive



Traditional predictive SDLC

- Earlier approach based on engineering
- Typically have sequential *Phases*
- Waterfall model
  - Phases are related groups of development activities, such as planning, analysis, design, implementation, and deployment
  - SDLC that assumes phases can be completed sequentially with no overlap or iteration
  - Once one phase is completed, you fall over the waterfall to the next phase, no going back

Traditional predictive SDLC waterfall

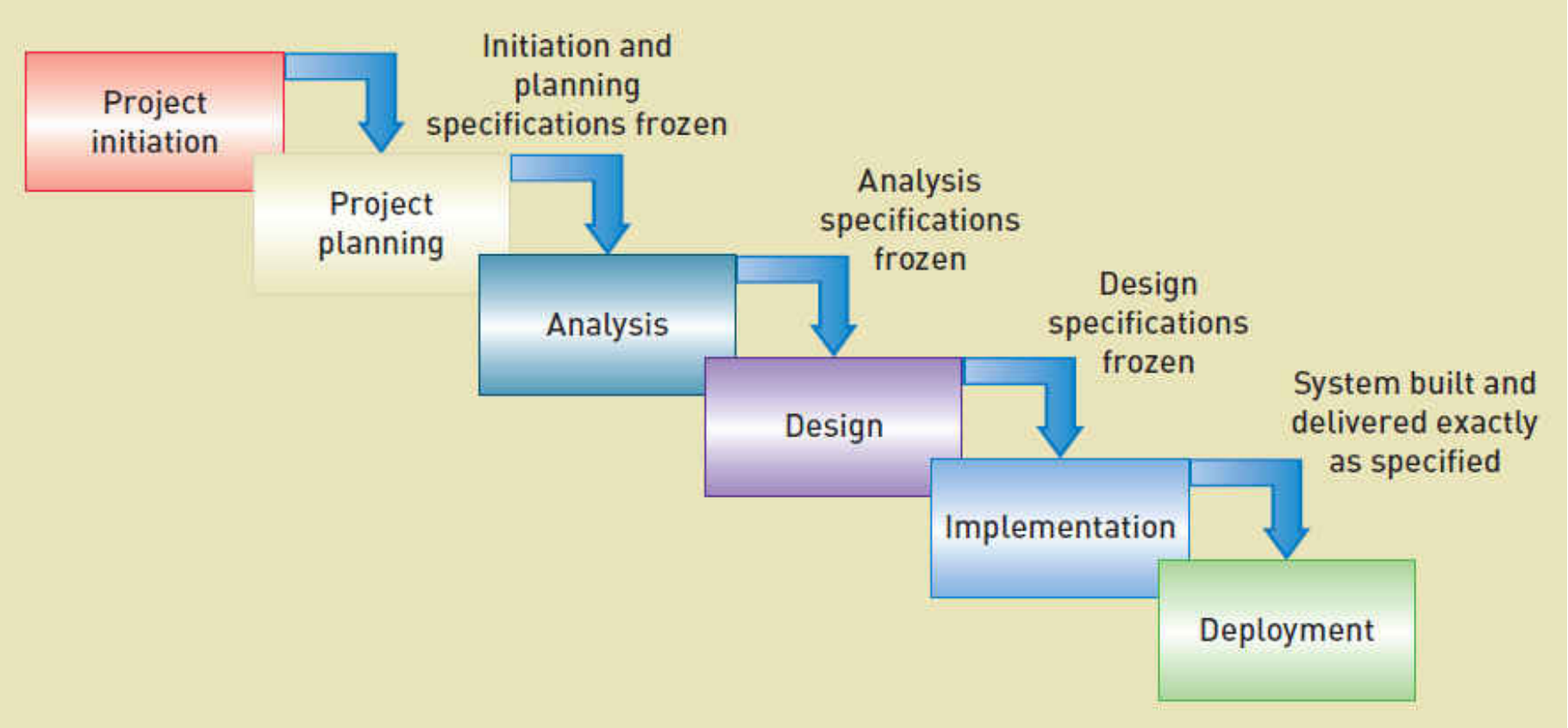
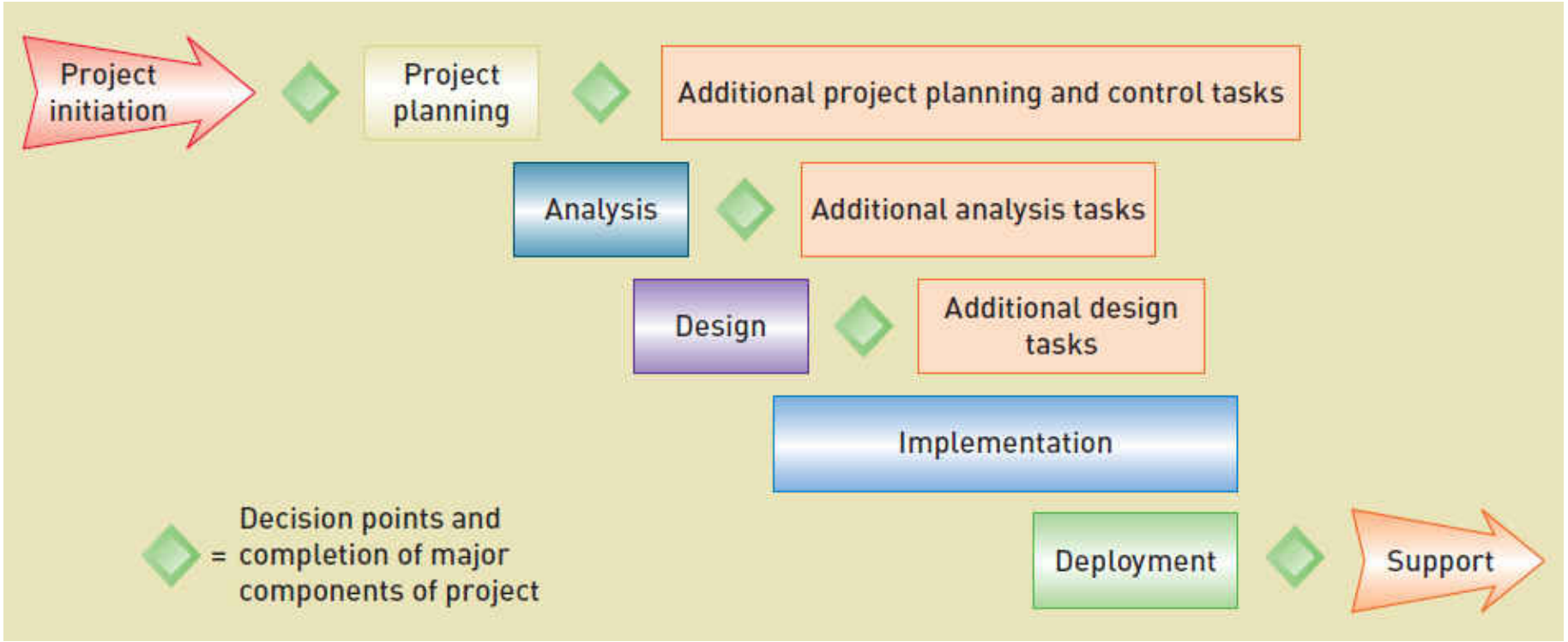


Image from: Systems Analysis and Design in a Changing World, 7th Edition ©2016. Cengage Learning

Newer predictive SDLC overlapping phases

Murdoch UNIVERSITY

More flexibility, but still assumes predictive planning and sequential phases



Adaptive SDLC

Murdoch UNIVERSITY

- Emerged in response to increasingly complex requirements and uncertain technological environments
- Always includes **iterations** where some of design and implementation is done from the beginning
- Many developers claim it is the ***only*** way to develop information systems
- Many IS managers are still sceptical

Iterative model

Using iterations, the project is able to adapt to changes as it proceeds.

Parts of the system are available early on for user evaluation and feedback

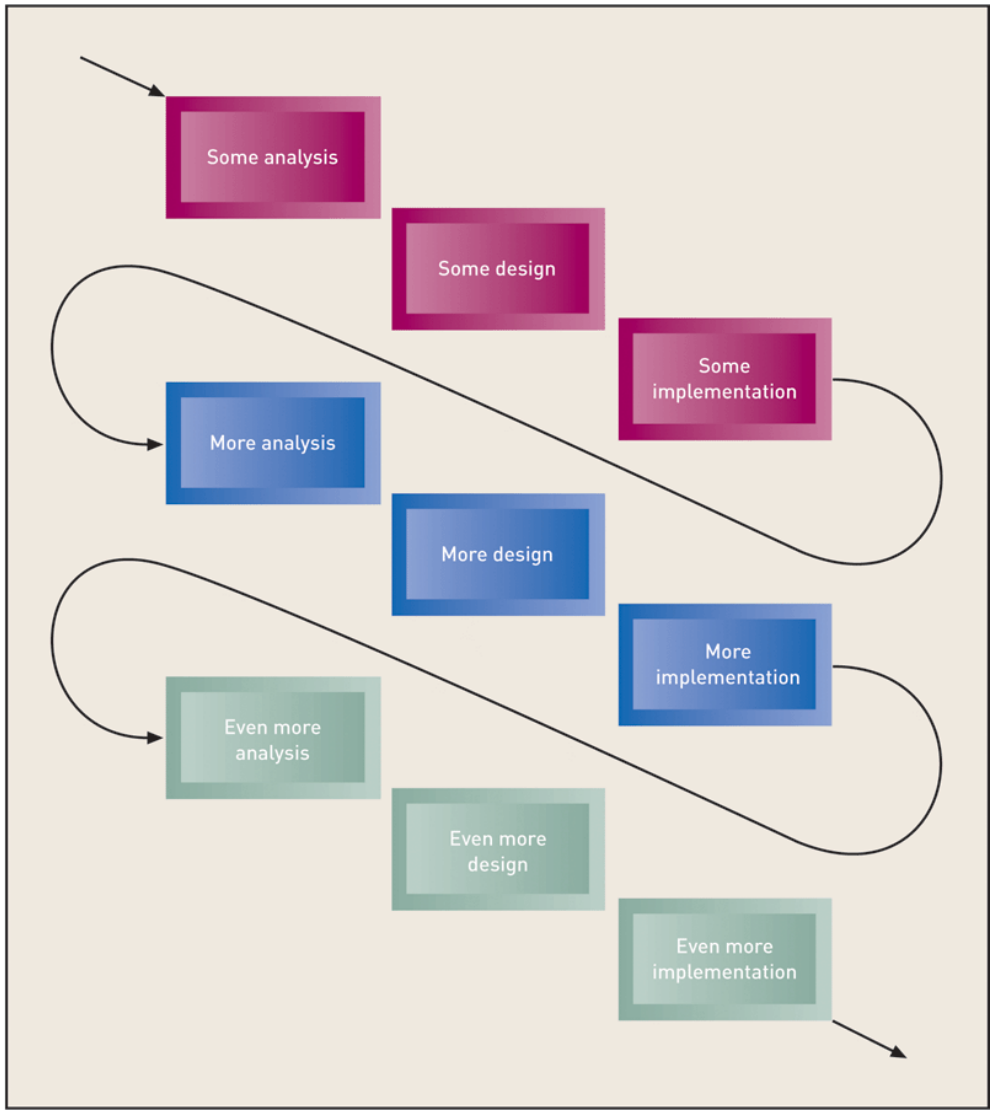


Image from Satzinger, J., Jackson, R. and Burd, S. (2004) *Systems Analysis and Design in a Changing World*, 3rd edition, Course Technology, Thomson.

Spiral model: the first adaptive SDLC

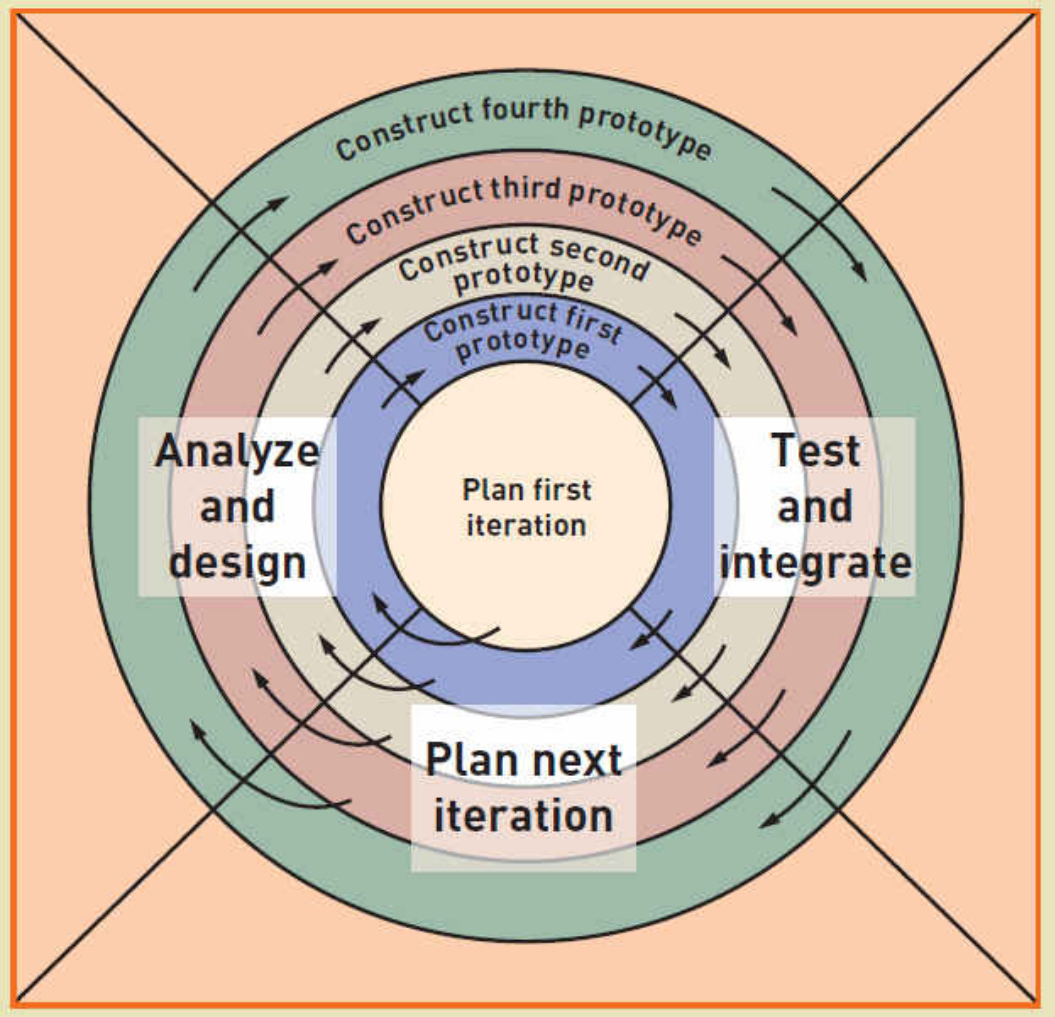


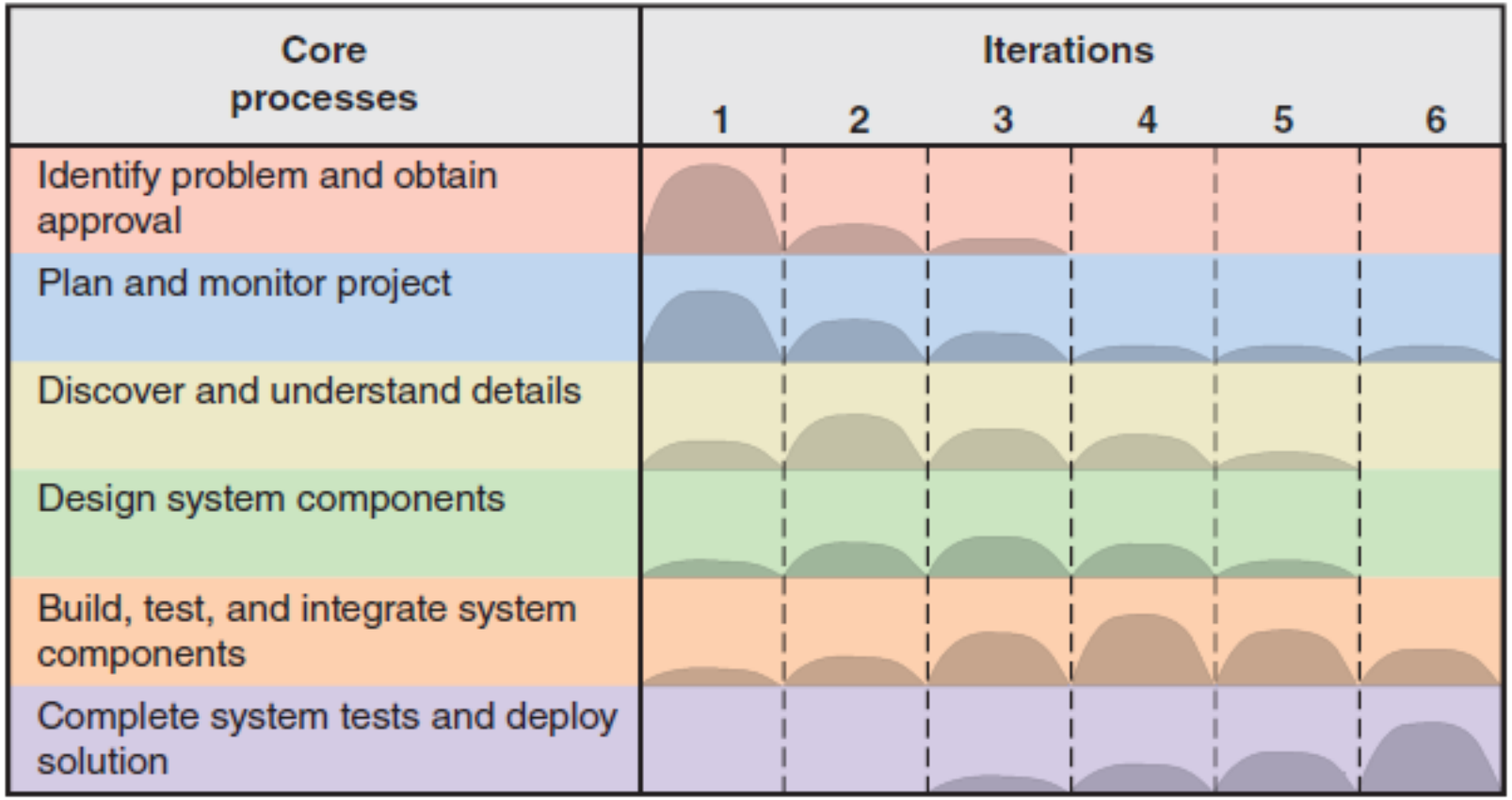
Image from Satzinger, J., Jackson, R. and Burd, S. (2004) *Systems Analysis and Design in a Changing World*, 3rd edition, Course Technology, Thomson.

The adaptive SDLC used in the text

Murdoch UNIVERSITY

Shows core processes, not phases, plus iterations in a sequence for management checkpoints

Based on the Unified Process SDLC



Additional adaptive concepts

- Incremental Development
  - An approach that completes portions of the system in increments
  - A system is implemented and partially deployed in steps during the project
  - Gets part of working system into users' hands sooner

Image source: https://www.infoq.com/news/2008/01/iterating-and-incrementing

- Walking Skeleton
  - An approach in which the complete system structure is built early, but with bare-bones functionality

Image source: https://www.slideshare.net/hepphep/walking-skeleton

Summing up…

The SDLC allows us to think about the development of a system as a progressive process. There are two general approaches to the SDLC:

Predictive approach

- Assumes the project can be planned in advance and that the system can be developed according to the plan
- Requirements are well understood and/or low technical risk

Adaptive approach

- Emerged in response to increasingly complex requirements and uncertain technological environments
- Using **iterations**, the project is able to adapt to changes as it proceeds and they occur.

Murdoch UNIVERSITY

Methodologies, models, tools and techniques

Murdoch UNIVERSITY

Aids for systems developers

As well as the concept of a SDLC, system developers have a variety of aids to help them complete activities and tasks. These include

- Methodologies
- Models
- Tools
- Techniques

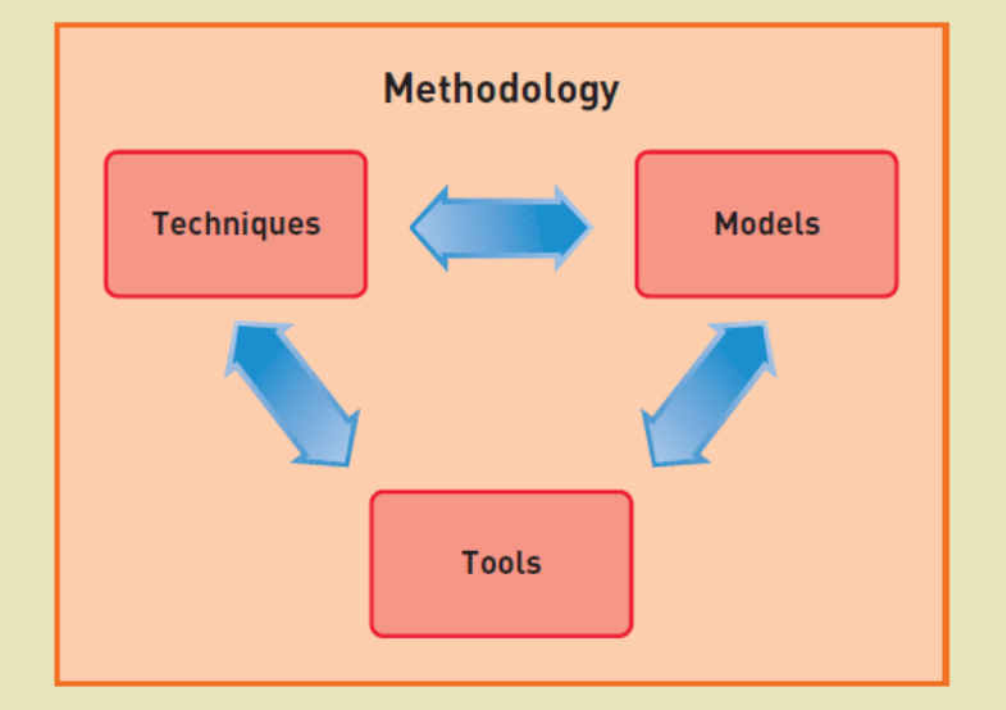


Image from: Systems Analysis and Design in a Changing World, 7th Edition ©2016. Cengage Learning

Methodologies

Provide *guidelines* for every facet of system development: *what to do when, why and how*

Methodology includes a collection of techniques that are used to complete activities and tasks, including modelling

- Specifies an SDLC with activities and tasks
- Specifies project planning and project management models and reporting
- Specifies analysis and design models to create
- Specifies implementation and testing techniques
- Specifies deployment and support techniques

Another term used is *System Development Process*

Models

A model is *an abstraction of an important aspect of the real world*

- Makes it possible to understand a complex concept by focusing only on a relevant part
- Each model shows a different aspect of the concept
- Crucial for communicating project information

In IS, some models are of system components; some models are used to manage the development process

Models

Murdoch UNIVERSITY

Some models of system components
- Use case diagram
- Domain model class diagram
- Design class diagram
- Sequence diagram
- Package diagram
- Screen design template
- Dialog design storyboard
- Entity-relationship diagram (ERD)
- Database schema

Some models used to manage the development process
- Gantt chart
- Organizational hierarchy chart
- Financial analysis models—NPV, payback period
- System development life-cycle model
- Stakeholders list
- Iteration plan

Tools

Software applications that *assists developers in creating models* or other components required for a project

- Integrated Development Environment (IDE) - a set of tools that work together to provide a comprehensive development environment
- Visual Modelling Tools - tools to create graphical models

Techniques

A collection of *guidelines that help an analyst complete an activity or task*

Learning techniques is the key to having expertise in a field

Strategic planning techniques
Project management techniques
User-interviewing techniques
Data-modeling techniques
Relational database design techniques
Structured programming techniques
Software-testing techniques
Process modeling techniques
Domain modeling techniques
Use case modeling techniques
Object-oriented programming techniques
Architectural design techniques
User-interface design techniques

Summing up…

- A **methodology** provides guidelines for every facet of system development: what to do when, why and how
- A methodology includes a collection of **techniques** that are used to complete activities and tasks
- **Tools** are software applications that assists developers in creating models or other components required for a project
- A **model** is an abstraction of an important aspect of the real world, used to help in understanding

Murdoch UNIVERSITY

Structured and object-oriented approaches

Murdoch UNIVERSITY

Two approaches to software construction and modelling

The Structured approach

- Early approach
- Assumes a system is a collection of processes that interact with data
- Involves structured analysis, structured design, and structured programming

The Object-Oriented approach

- More recent approach
- Assumes a system is a collection of objects that interact to complete tasks
- Involves OO analysis, OO design, and OO programming

The Structured approach

Top down, modular programming

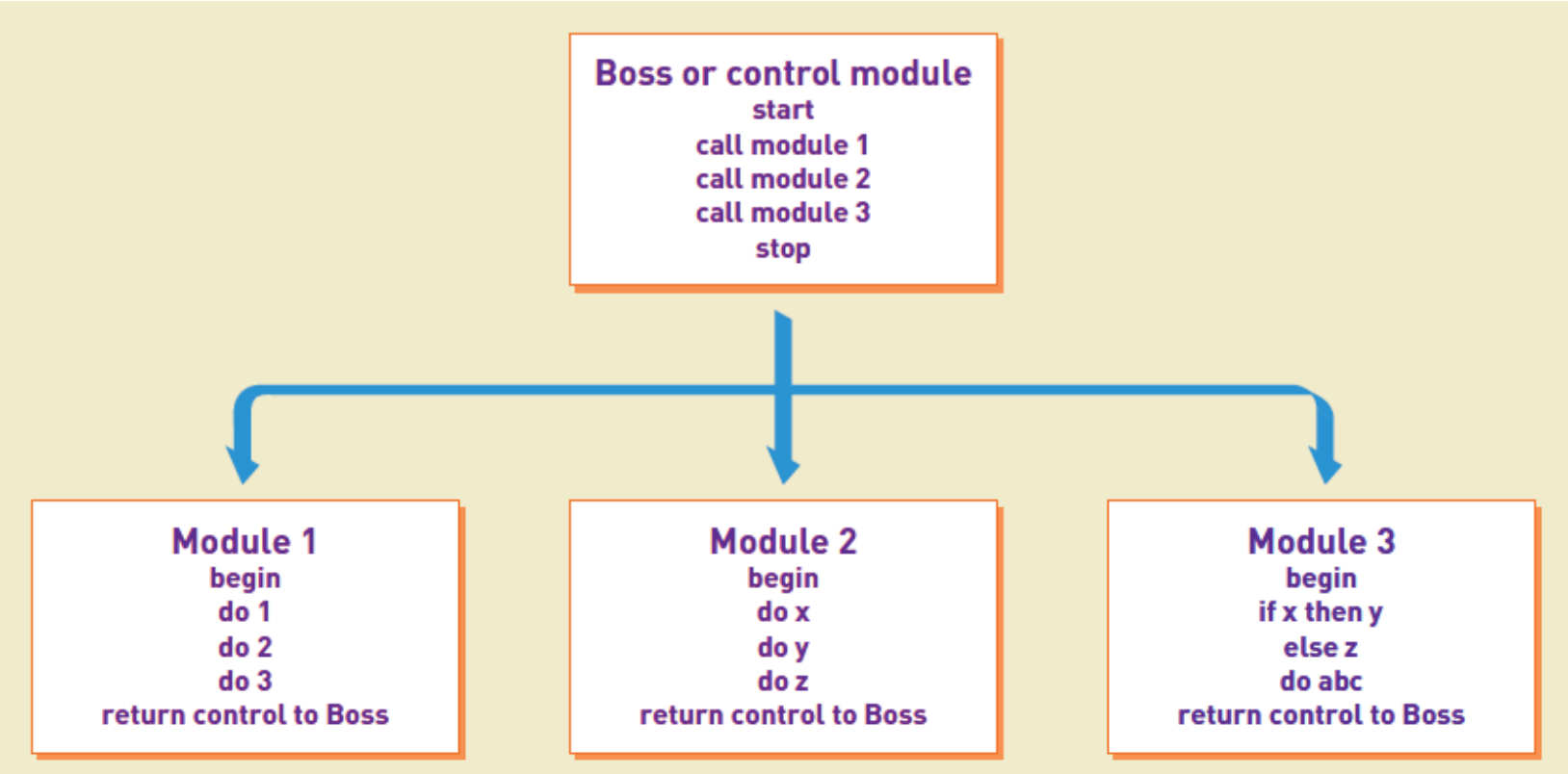


Image from Satzinger, J. Jackson, R. & Burd, S. (2007) Systems Analysis and Design in a Changing World,5th edition. Course Technology, Thomson Learning.

The Object-Oriented approach

Object-oriented analysis (OOA)

The process of identifying and defining the use cases and sets of objects (classes) in the new system

Object-oriented design (OOD)

Defining all of the types of objects necessary to communicate with people and devices and showing how they interact to complete tasks

Object-oriented programming (OOP)

Writing statements that define the actual classes and what each object of the class does

Summing up…

- Two broad approaches to development methodologies are *structured* and *object-oriented*
- The older, **structured** approach assumes a system is *a collection of processes that interact with data*
- The more recent **object-oriented** approach assumes a system is *a collection of objects that interact to complete tasks*
- The OO approach involves all phases: OO analysis, OO design and OO programming

Murdoch UNIVERSITY

Agile development

Murdoch UNIVERSITY

Agile development

- A guiding philosophy and set of guidelines for developing information systems in an unknown, rapidly changing environment
- Complements Adaptive SDLCs and Methodologies that support it
- Takes adaptive and makes sure developers are fast on their feet to respond to changes

Agile development philosophies and values

The text emphasises agile values, as stated by the

Manifesto for Agile Development

http://agilemanifesto.org/

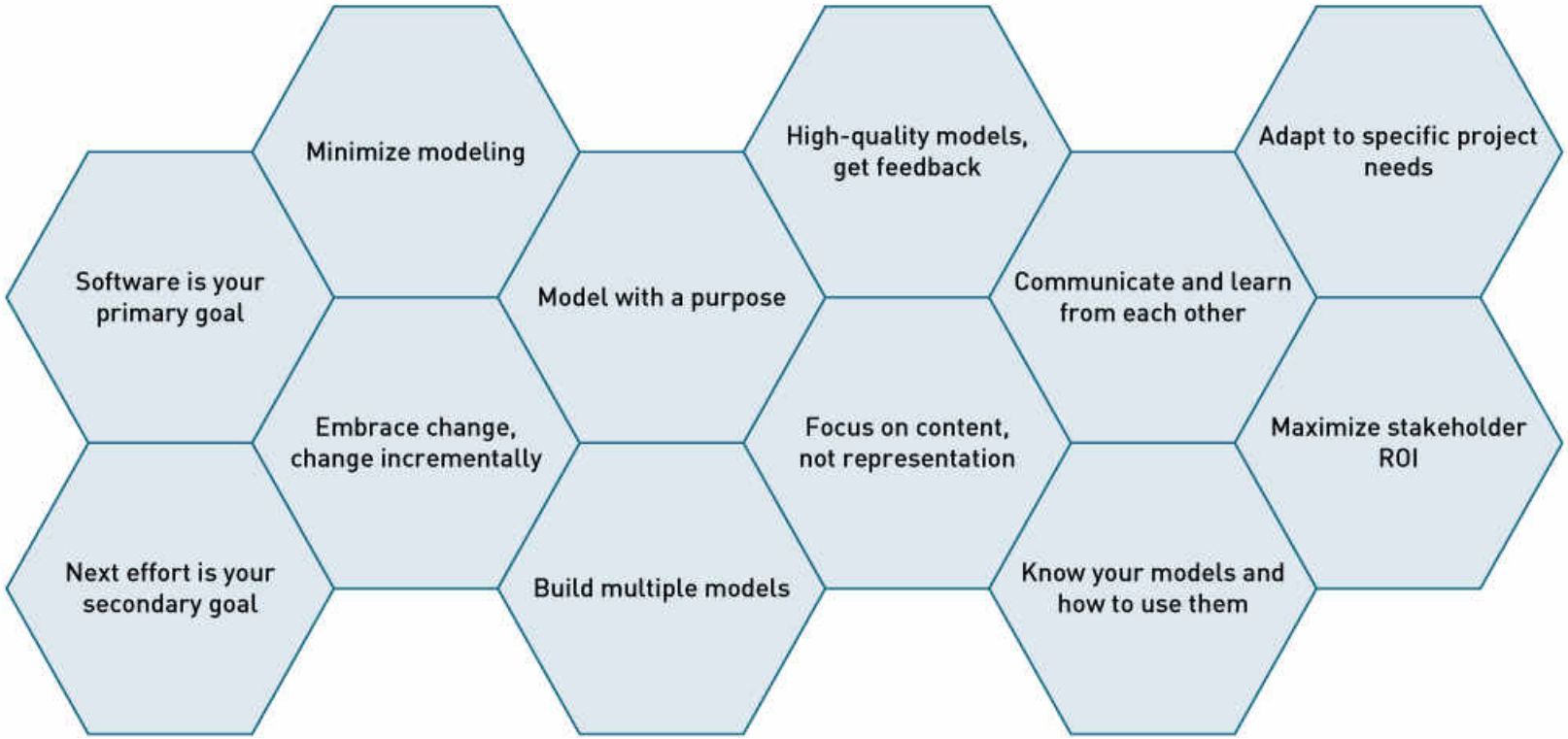
- Value *responding to change* over following a plan
- Value *individuals and interactions* over processes and tools
- Value *working software* over comprehensive documentation
- Value *customer collaboration* over contract negotiation

Example: Agile modelling

Murdoch UNIVERSITY

Based on 12 principles:

build only necessary models that are useful and at the right level of detail



Agile principles

Murdoch UNIVERSITY

- Develop software as the primary goal
  Don’t get distracted by documentation or models
- Enable the next effort as your secondary goal
  Be aware of next step versions or revisions
- Minimise your modelling activity
  Only build what helps move the project forward
- Embrace change and change incrementally
  Take small steps that keep you on-track and that can be reversed if necessary

Agile principles

Murdoch UNIVERSITY

- Model with a purpose
  - Model to understand
  - Model to communicate
- Build multiple models
  - Look at problems from different perspectives
- Build high-quality models and get feedback
- Focus on content rather than representation
  - Informal hand-drawn models are sometimes okay
  - Always focus on stakeholder needs

Agile principles

Murdoch UNIVERSITY

- Learn from each other with open communication
- Know your models and how to use them
- Adapt to specific project needs
- Maximise stakeholder ROI

Summing up…

- Agile development is a *guiding philosophy and set of guidelines* for developing information systems in an unknown, rapidly changing environment
- Agile is not a methodology, but *a set of principles* that incorporate a set of values:

  Value responding to change over following a plan

  Value individuals and interactions over processes and tools

  Value working software over comprehensive documentation

  Value customer collaboration over contract negotiation

Murdoch UNIVERSITY

Some examples: Unified Process, Xtreme Programming, Scrum

Incorporating Agile principles

Murdoch UNIVERSITY

The Agile philosophy only proposes *principles* – methodologies that incorporate Agile principles are briefly discussed next:

- UP – developed to support OOSD in the 1990s
- XP - the beneficial elements of traditional software engineering practices are taken to "**extreme**" levels (also 1990s)
- SCRUM - a feedback-driven approach underpinned by the three pillars of transparency, inspection, and adaptation

The Unified Process (UP)

The UP was an early leader in adaptive approaches

- UP and UML (Unified Modeling Language) were developed together
- UP phases organise iterations into four primary areas of focus during a project
  - Inception phase – getting the project started
  - Elaboration – understanding the system requirements
  - Construction – building the system
  - Transitions – preparing for and moving to deploying the ne

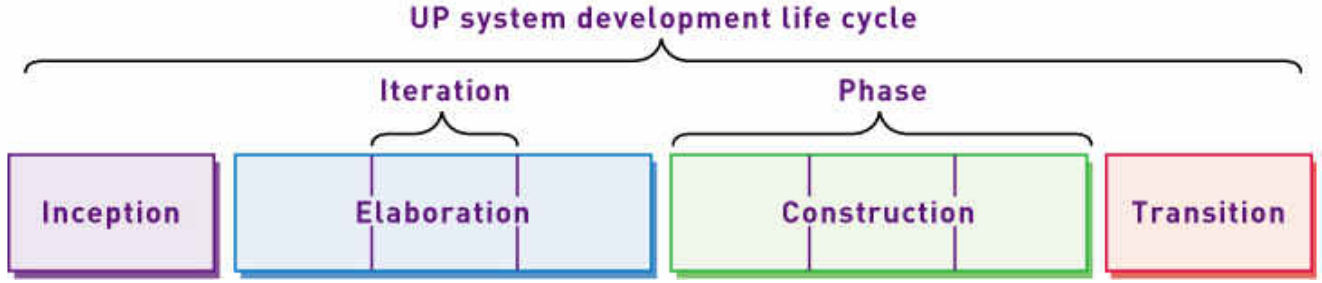


Image from: Systems Analysis and Design in a Changing World, 7th Edition ©2016. Cengage Learning

Unified Process – disciplines

Murdoch UNIVERSITY

A set of functionally related development activities

- Each discipline are all the activities related to achieving one objective in the development project
- Two types of disciplines

Development disciplines

Business modelling
Requirements
Design
Implementation
Testing
Deployment

Management – planning and control disciplines

Configuration and change management
Project management
Environment

Unified Process – disciplines

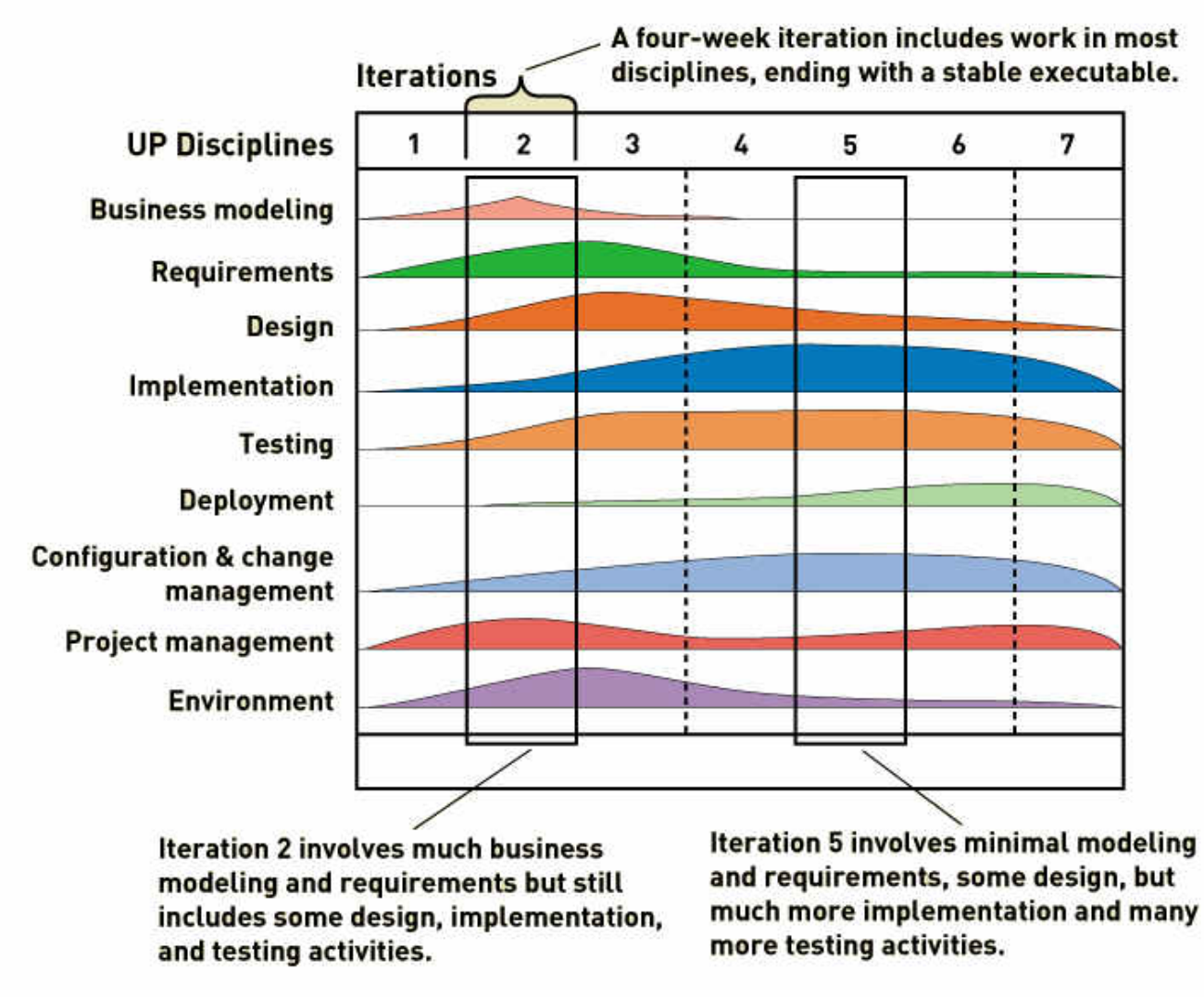


Image from: Systems Analysis and Design in a Changing World, 7th Edition ©2016. Cengage Learning

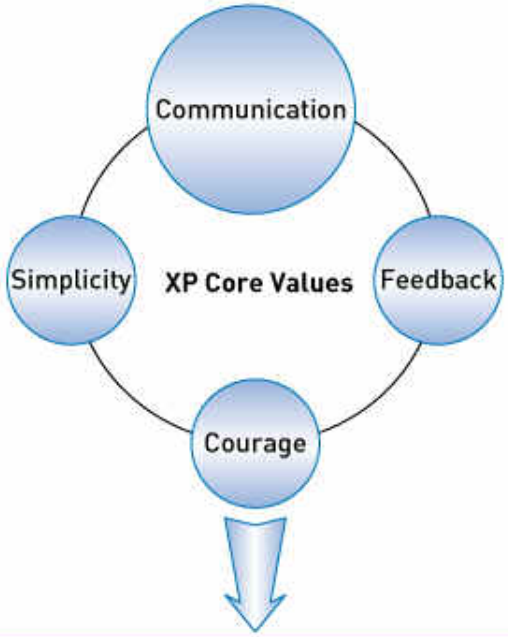
Extreme Programming (XP)

Murdoch UNIVERSITY

- Takes the best practices of software development and extends them "to the extreme"
  - Focus intensely on proven industry practices
  - Combine them in unique ways to get better results
- XP Core values
  - Communication - open, at the right time, level, and with the right people
  - Simplicity – reinforced by techniques
  - Feedback – frequent , meaningful and from the right stakeholders
  - Courage – to "do it right"

XP core values &

Planning – based on user stories
Testing – thorough testing at every step
Pair Programming – watch, inspect, trade off
Simple Designs – Agile modelling principles
Refactoring – redo and cleanup as you go
Owning the code collectively – egoless development, anyone can review and improve code
Continuous integration – grow the software continuously
On-site customer – get sign-off as you go
System metaphor – what should the final system look like
Small releases – turn over to user frequently
Forty-hour work week – don't overload the developers
Coding standards – follow standards for code



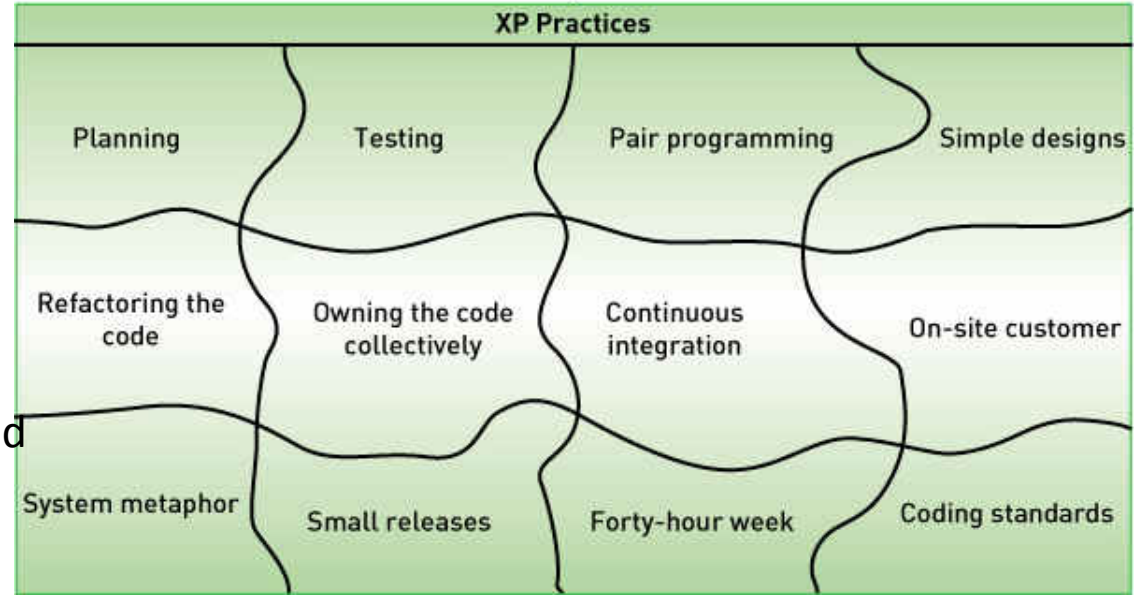


Image from: Systems Analysis and Design in a Changing World, 7th Edition ©2016. Cengage Learning

SCRUM

- Combination of principles of Rugby and Agile
  - Intense effort involving the entire team for a defined period of time
- Product backlog
  - Prioritised list of user requirements
- Product owner
  - The client stakeholder who controls backlog
- Scrum master
  - Scrum project
- Sprint

  A time-controlle
  mini-project to
  implement part
  the system

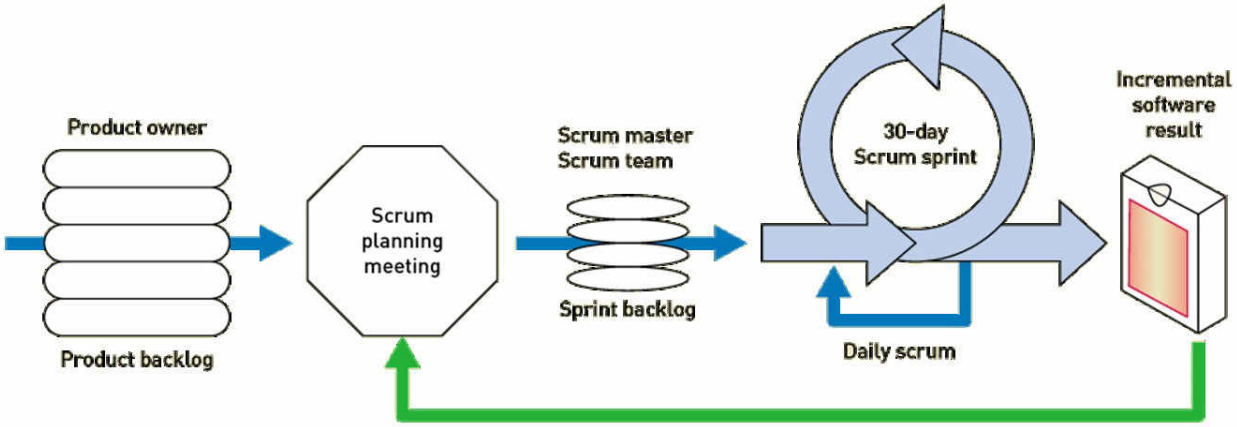


Image from: Systems Analysis and Design in a Changing World, 7th Edition ©2016. Cengage Learning

Summing up…

Some examples of methodologies incorporating Agile philosophies include UP, XP, Scrum:

- Unified Process is a formal iterative approach which uses UML models (with Agile philosophy) and UP disciplines
- Extreme Programming (XP) is an iterative approach which takes good practices to the extreme
- Scrum is an iterative approach using a Scrum Sprint, with all other processes supporting the sprint

Topic learning outcomes revisited

After completing this topic you should be able to:

- Compare the underlying assumptions and uses of a predictive and an adaptive system development life cycle (SDLC)
- Explain what makes up a system development methodology, including the SDLC as well as models, tools, and techniques
- Describe the key features of Agile development
- Understand and briefly describe the key features of the Unified Process, Extreme Programming, and Scrum Agile system development methodologies

Murdoch UNIVERSITY