MACIASZEK, L.A. (2007):
Requirements Analysis and System Design, 3rd ed.
Addison Wesley, Harlow England
ISBN 978-0-321-44036-5

Chapter 1
Software Process

Topics

- The nature of software development
- System planning
- Systems for three management levels
- The software development lifecycle
- Development models and methods
- Problem statements for case studies (separate set of slides)

1. The nature of software development

- 70% of software projects fail
(The Standish Group report, 2015) –
an exaggeration?

The essence of software development

- defined by the issues inherent in the software itself → software is a product of a creative act (not a result of a repetitive act of manufacturing)
- difficulties not amenable to breakthroughs or ‘silver bullets’ → software development **invariants** (constants):
  - complexity,
  - conformity,
  - changeability, and
  - invisibility.

The accidents of software development

- '**Accidental difficulties**' (variables) due to software production practices → amenable to human intervention
  - attributed mostly to the fact that an information system is a social system
  - the software solution must not be adding to the inherent complexity of the software product
  - adaptiveness (supportability) is the challenge
    - adaptiveness = understandability + maintainability + scalability (extensibility)
- Related to:
  - Stakeholders
  - Process
  - Modeling

Variable 1 - Stakeholders

- People who have a stake in a software project:
  - Customers (users and system owners)
  - Developers (analysts, designers, programmers, etc.)
- Information systems are social systems → developed by people (developers) for people (customers)
- The main causes of software failure can be traced to the stakeholder factor
  - on the customer end, and
  - on the developer end

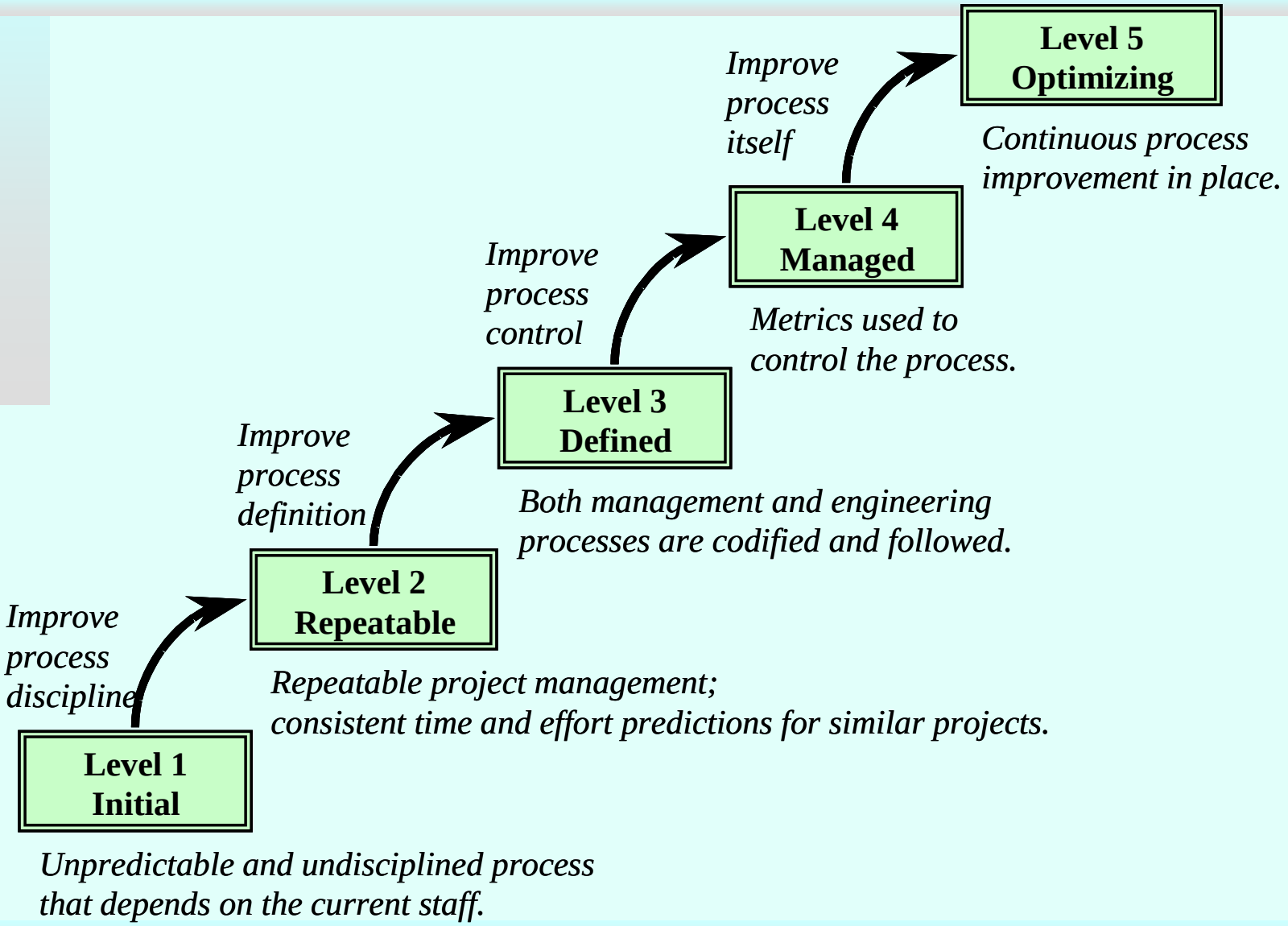
Variable 2 - Software process

- Defines activities and organizational procedures used in software production and maintenance
- A process model:
  - states an order for carrying out activities;
  - specifies what development artifacts are to be delivered and when;
  - assigns activities and artifacts to developers;
  - offers criteria for monitoring a project's progress, for measuring the outcomes, and for planning future projects.
- Is not susceptible to standardization

Iterative and incremental process

- 'An **iterative** process is one that involves managing a stream of executable *releases*. An **incremental** process is one that involves the continuous *integration* of the system's architecture to produce these releases, with each new release embodying incremental improvements over the other.' (RUP)
- Some examples:
  - the spiral model
  - the Rational Unified Process (RUP)
  - Model Driven Architecture (MDA)
  - the agile development process
  - aspect-oriented software development
- Iterative and incremental development
  - must be planned and controlled, and
  - must conform to a pre-defined *architectural design framework* (meta-architecture)

Capability maturity model



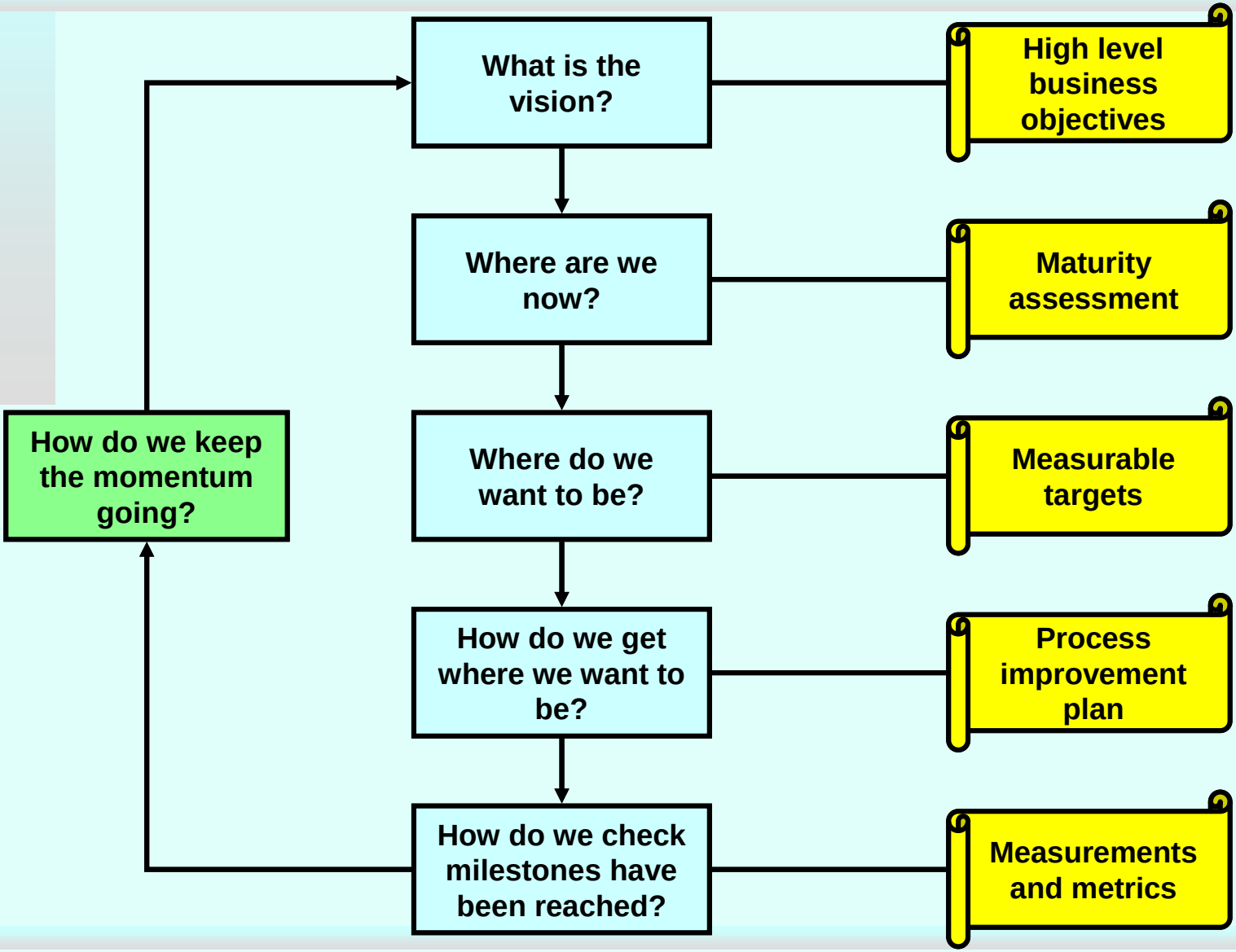
ISO 9000 family of quality standards

- Developed by the International Organization for Standardization
- The ISO standards
  - apply to the quality management and the process to produce a quality product
  - apply to any industry and all types of businesses, including software development
- The main premise
  - if the process is right then the process outcome (product or service) will also be right
  - but... the ISO standards do not enforce or specify processes → the standards provide models of **what** must be accomplished, **not how** activities must be performed

ITIL framework

- IT as commodity
- Software becomes merely a service enabling business solution → *solution (service) delivery*
- IT Infrastructure Library (ITIL) – the most widely used and accepted framework of best practices for IT service management
  - Efficient and effective use of the *four Ps*
    - people, processes, products (tools and technology) and partners (suppliers, vendors and outsourcing organizations).
  - Next slide - solution management as a Continuous Service Improvement Programme (CSIP)

CSIP


What is the vision?
High level business objectives
Where are we now?
Maturity assessment
Where do we want to be?
Measurable targets
How do we get where we want to be?
Process improvement plan
How do we check milestones have been reached?
Measurements and metrics
How do we keep the momentum going?

COBIT framework

- ITIL addresses the operational side of the solution delivery and management
  - ITIL, CMM and ISO 9000 are ***process standards***
- COBIT (Control OBjectives for Information and related Technology) is a compliance framework and addresses the control side of the solution management
  - COBIT is rather a ***product standard***
- COBIT groups IT-related efforts into four *domains*:
  - Plan and Organize,
  - Acquire and Implement,
  - Deliver and Support, and
  - Monitor.
- The domains are assigned *control objectives*
  - 34 high-level control objectives
  - 318 recommended detailed control objectives

Variable 3 - Modeling

- Modeling *artifacts* have to be
  - communicated (language) and
  - documented (tools)
- 'The **Unified Modeling Language (UML)** is a general-purpose visual modeling language that is used to specify, visualize, construct, and document the artifacts of a software system.'
- **Computer-Assisted Software Engineering (CASE)** tool enables storage and retrieval of models in a central repository and graphical and textual manipulation of models on a computer screen

UML

- **UML** is independent of
  - any software development process
    - A process that adopts UML must support an object-oriented approach to software production
  - implementation technologies (as long as they are object-oriented)
    - This makes UML somewhat deficient in supporting the detailed design phase of the development lifecycle
- The **UML models** can be categorized into three groups:
  - **State models**
    - describe the static data structures
  - **Behavior models**
    - describe object collaborations
  - **State change models**
    - describe the allowed states for the system over time

CASE and process improvement

- **Process improvement** is much more than the introduction of new *tools*, *methods* and *techniques*
  - the introduction of new methods and techniques to organization at a low level of process maturity can bring more harm than good
- An integrated **CASE** tool can allow multiple developers to collaborate and share design information in order to produce new design artifacts → the tool imposes processes on the development team → in “immature” organizations processes will not be followed (creating more mess than before)
- However, a CASE tool would always bring personal productivity and quality improvements to individual developers

Development or integration?

- Application development
  - Stand-alone
  - From-scratch
- Integration development
  - Value-added or
  - Brand new application that (also) requires integration of existing apps
- Integration approaches
  - Information- and/or portal-oriented
  - Interface-oriented
  - Process-oriented

Review Quiz 1.1

1. Do “accidents” of software development define the software development invariants?
2. What are the two main groups of stakeholders in software projects?
3. Does each incremental release within an iteration add a new functionality to the software product under development?
4. Is COBIT a product or a process standard?
5. Is portal-oriented integration a special kind of interface-oriented integration?