Software Process Improvement: An Introduction

Hossein Saiedian, Ph.D.

EECS811: IT Project Management

Electrical Engineering & Computer Science

Spring 2013

What is Quality?

- Engineering quality software
- Quality
  - A key issue in the field of software engineering
  - Popular view: difficult to define and measure
  - Professional view: quantifiable, controllable, manageable, improvable
  - A definition: *"conformance to requirements"*

    P. Crosby, *Quality is Free: The Art of Making Quality Certain*, McGraw-Hill, 1979

Conformance to Requirements: Implications

- During the production process, measurements must continually be taken to determine conformance to those requirements:
  - *measurement model*
  - *project tracking and oversight*
  - *validation criteria*
  - *quality assurance system*
  - *plans, commitment to improvement*

  } Managerial aspects

- The use of process models is encouraged

Conformance to Requirements: Implications

- Requirements must be clearly stated such that they cannot be misunderstood:
  - *complete*
  - *unambiguous*
  - *verifiable*
  - *precise*
  - *concise*
  - *consistent*

  } Technical aspects

- The use of formalism is encouraged

Process Fundamentals

- Process: The means by which **people**, **procedures**, and **tools** are integrated to produce a product (or an end result)
- Software Process: The set of all tasks involved in the production and evolution of a software product
  - tasks are organized and sequenced
  - tasked are performed in accordance with a procedure

Process Management Principles

- The quality of a product is largely determined by the quality of the process used to build it
- By extension, the quality of a software product is largely determined by the quality of the software process used for developing and maintaining it
- *To improve the quality of a software product, the process for producing it must improved*
- FACT: the majority of software problems or the causes of software crisis (e.g., budget overrun, lack of quality, late delivery) are managerial, not technical (Humphrey 1989)

Examples of Software Crisis and Problems

- A review of 17 major DoD software projects revealed that (Humphrey 1993):
  - *Average 28-month schedule was missed by 20 months*
  - *One project was not delivered for 7 years*
  - *No project was on time*
- Nine DoD contracts totaling $6.8 million
  - 47% software delivered but never used
  - 29.7% software paid for but never delivered
  - 19% software used but later reworked or abandoned
  - $\approx$ 3% software could be used after changes
  - $\approx$ 2% software could be used as delivered

Examples of Software Crisis and Problems (continued)

- Industry results are not any better (Gibbs 1994)
  - For every six new large software systems put into operation, two others are canceled
  - Average software project overruns its schedule by half; large projects do worse
  - Many projects are terminated after millions of dollars invested, e.g., CONFIRM project (over $200 M),
    AAS ($144 M), DMV ($44.3 M)
  - DIA's Baggage Handling System, delayed for more than a year at $1.1 million/day in interests and operating costs
- The Standish Group (standishgroup.com) reveals more staggering figures

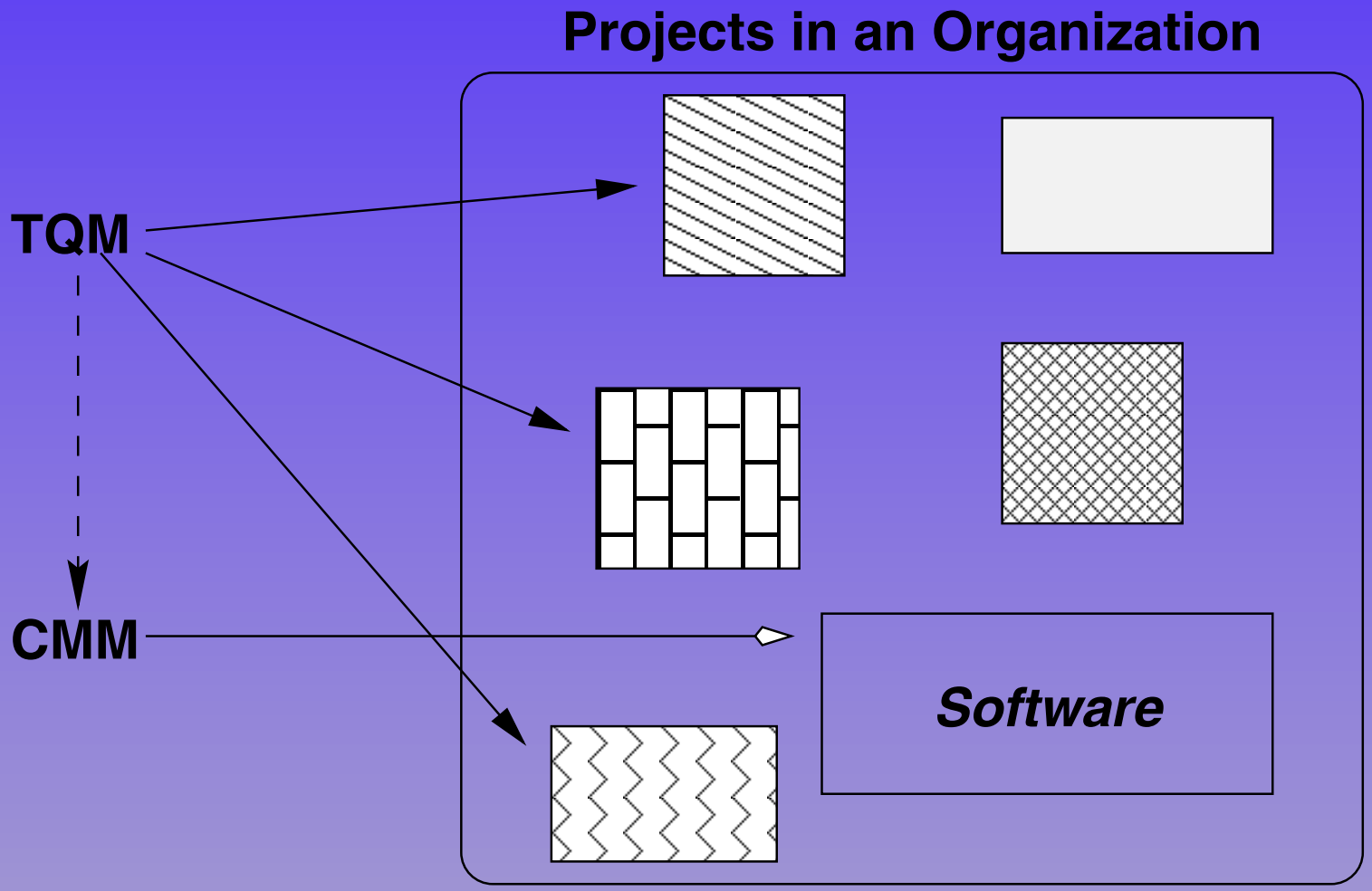
Characteristics of Immature Organizations

- Processes are generally improvised during the project
- Specific processes are not rigorously followed
- Schedules and budgets are routinely exceeded
- The organization is reactionary
- Product quality suffers

Software Process Improvement Models

- Objective: to provide a framework for applying process management and quality improvement concepts to software development and maintenance
- Examples: Capability Maturity Model (CMM), AMI, SPICE, Bootstrap, Trillium, ISO 9000-3 Standards; also PSP and TSP
- Based on process management concepts from the TQM movement
  - *Enabling quality improvement is a management responsibility*
  - *Quality improvement focuses on fixing processes not people*
  - *Quality improvement must be measured*
  - *Quality improvement is a continuous process*
- TQM applied to software projects

CMM: TQM Applied to Software



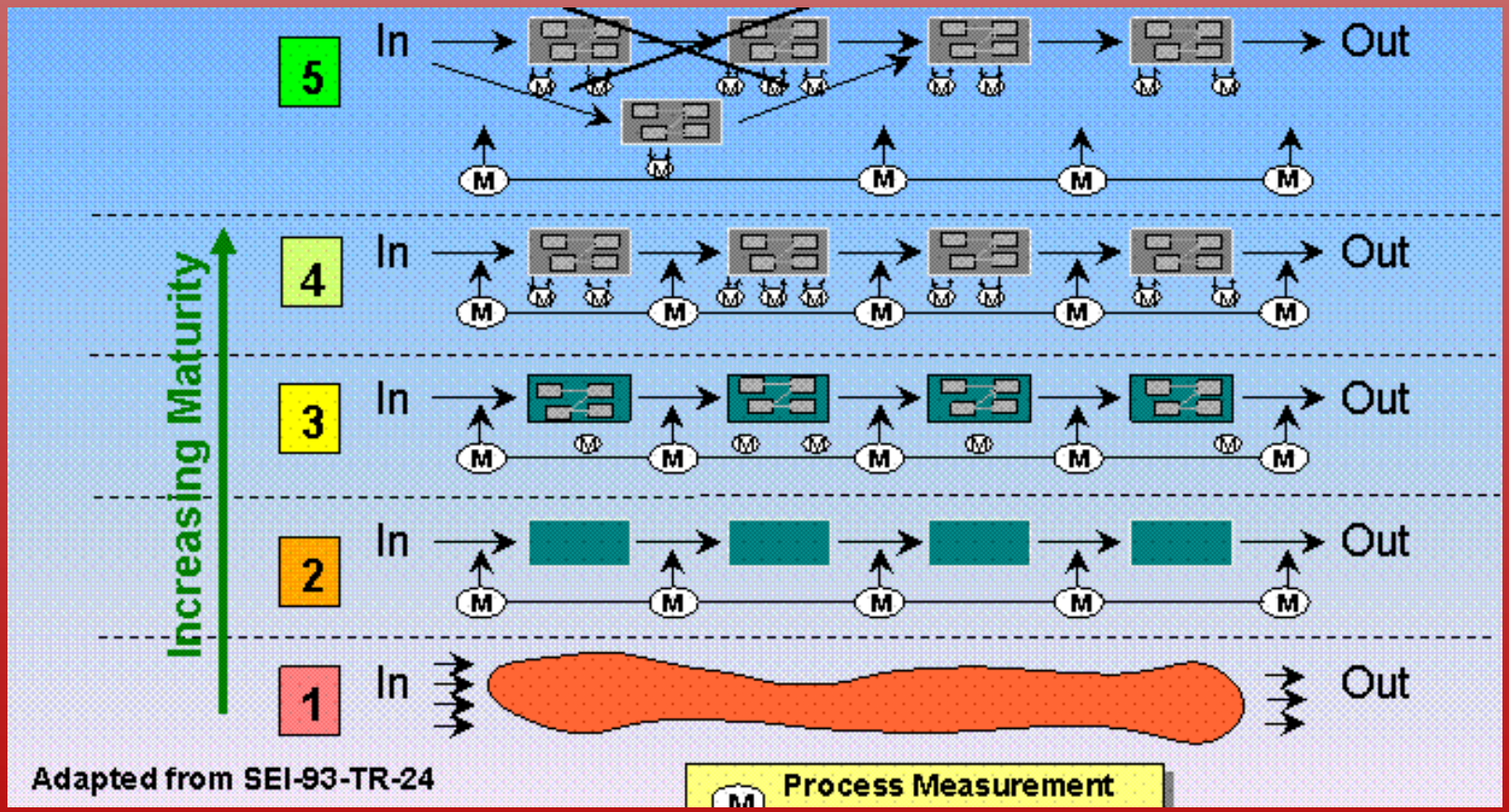
CMM's Five-Level Framework

- **Initial:** unpredictable, poorly controlled
- **Managed (Repeatable):** basic process management practices are established; organization can repeat previously mastered tasked
- **Defined:** the software process for both management and engineering activities is documented and well understood
- **Quantitatively Managed:** detailed measures of the software process and product quality are collected; both are understood and controlled
- **Optimizing:** focus on process improvement; feedback from piloting innovative ideas and technologies

How Maturity Affects Project Results?

- Level 1: Initial Schedule and cost targets are typically overrun
- Level 2: Managed (Repeatable) Plans based on past performances are more realistic
- Level 3: Defined With well-defined processes, performance improves
- Level 4: Quantitatively Managed Based on quantitative understanding of process and product, performance continues to improve
- Level 5: Optimizing Performance continuously improves

How Maturity Affects Project Visibility?



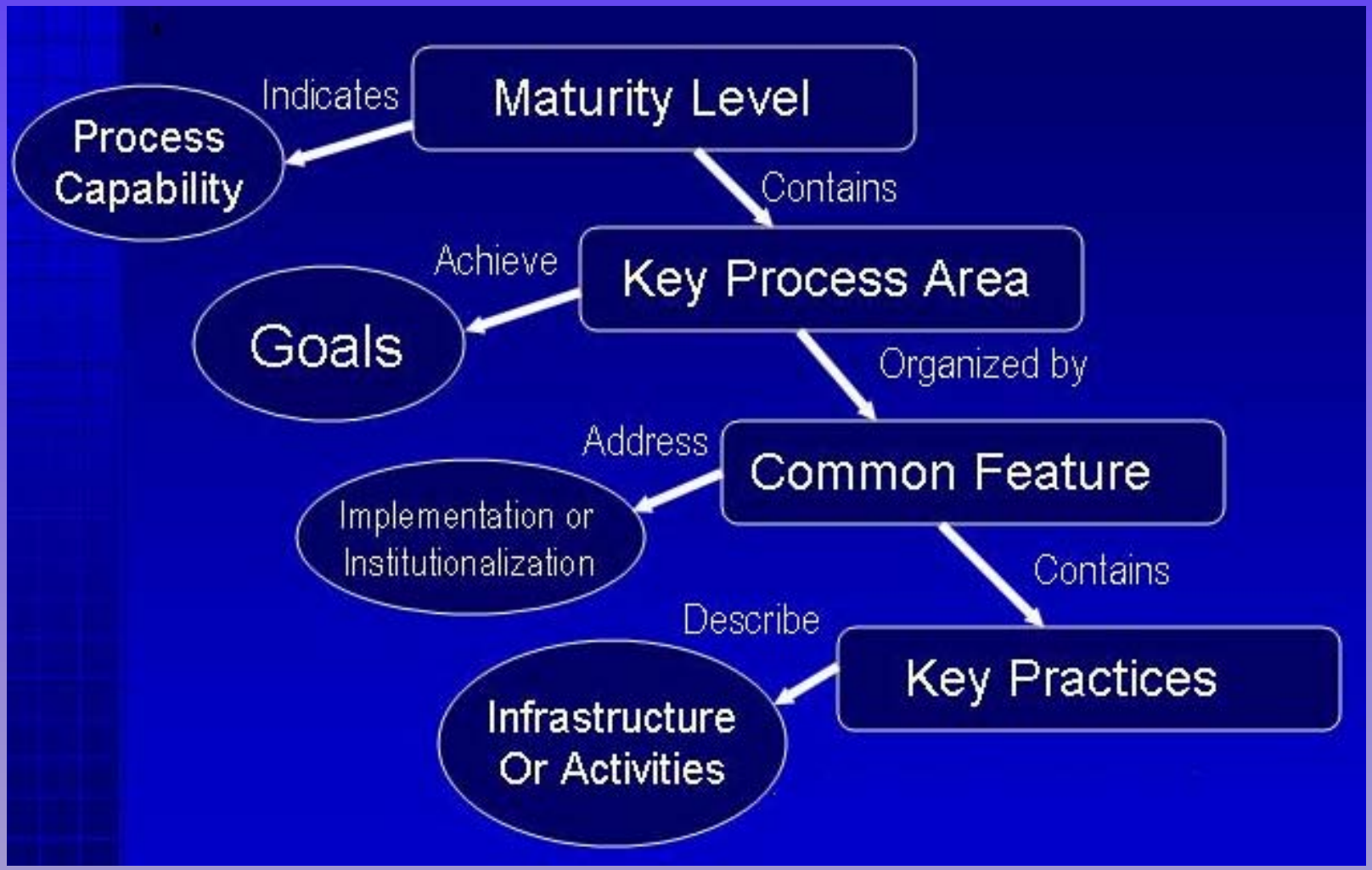
Components of the CMM

- Maturity Levels: indicate process capability
- Key Process Areas: achieve goals
- Common Features: implementation/institutionalization criteria
- Key Practices: low level activities

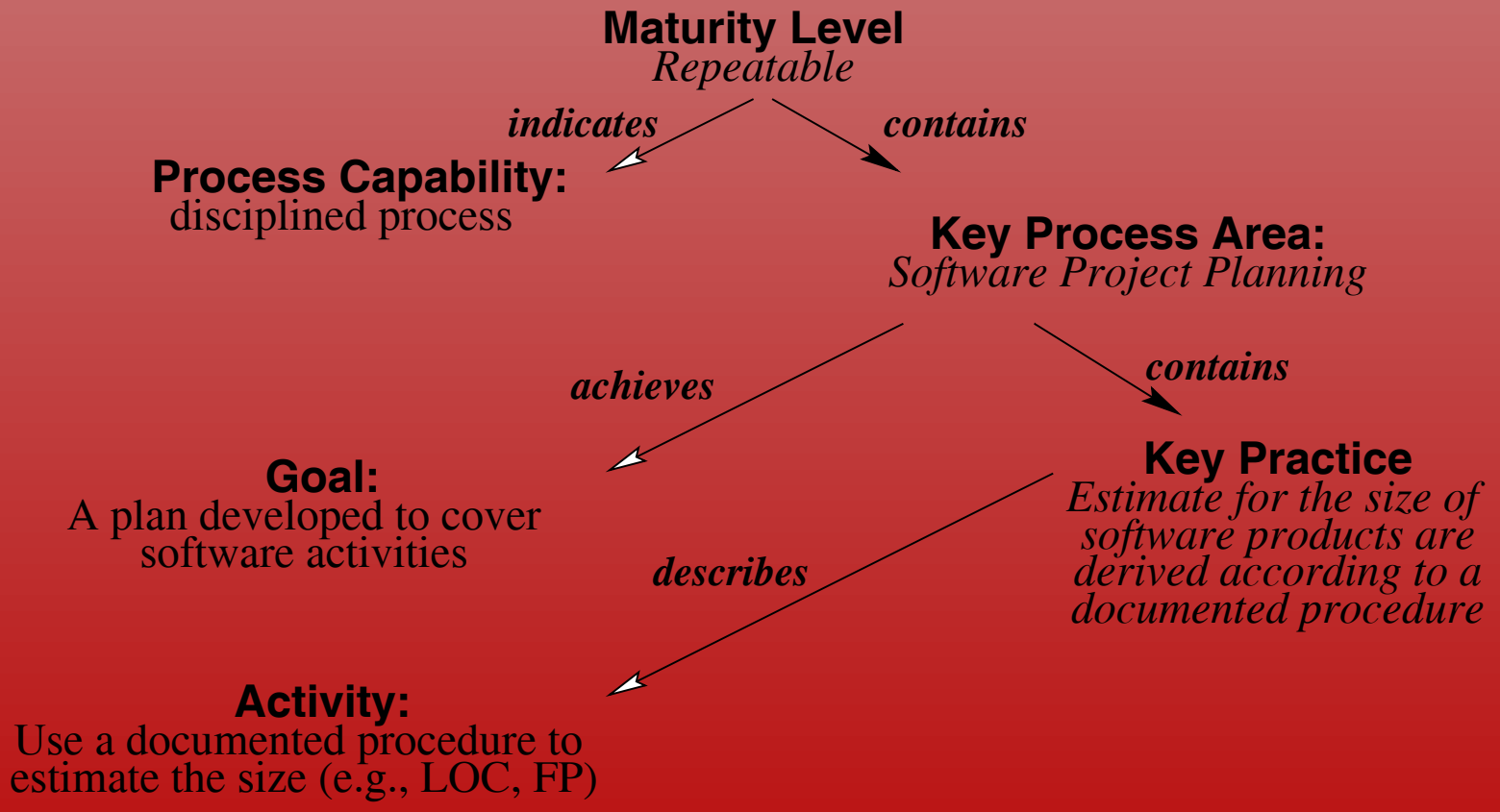
Key Process Area: Definition

A cluster of related activities (*key practices*) that, when performed collectively, achieve a set of goals considered important for establishing process capability

Pictorial Representation of the CMM Components



Pictorial Representation: An Example



Key Process Areas by Levels

- Level 1: None
- Level 2: 6 KPAs (e.g., Software Project Planning)
- Level 3: 7 KPAs (e.g., Peer Reviews)
- Level 4: 2 KPAs (e.g., Quantitative Process Management)
- Level 5: 3 KPAs (e.g., Process Change Management)

KPAs of Level 2 (Managed or Repeatable)

- Requirements Management
- Software Project Planning
- Software Project Tracking and Oversight
- Software Subcontract Management
- Software Quality Assurance
- Software Configuration Management

KPA: Software Quality Assurance

- Objective: to provide management with appropriate visibility into the process being used by the software project and of the products being built
- Involves reviewing and auditing the software products and activities to verify that they comply with the applicable procedures and standards and providing the software project and other appropriate managers with the results of these reviews and audits

SQA KPA Goals

1. Software quality assurance activities are planned
2. Adherence of software products and activities to the applicable standards, procedures, and requirements is verified objectively
3. Affected groups and individuals are informed of software quality assurance activities and results
4. Noncompliance issues that cannot be resolved within the software project are addressed by senior management

Goal: SQA Activities Are Planned — Common Features

Commitment to Perform: — the project follows a written organizational policy for implementing SQA

Ability to Perform — a trained SQA group that is responsible for coordinating and implementing for the project exists; adequate resources are provided

Activities Performed — a SQA plan is prepared for the software project according to a documented procedure; activities performed in accordance with the plan

Measurement and Analysis — measurements are made and used to determine the cost and schedule status of the SQA activities

Verifying Implementation — the SQA activities are reviewed with the project manager on both a periodic and event-driven basis

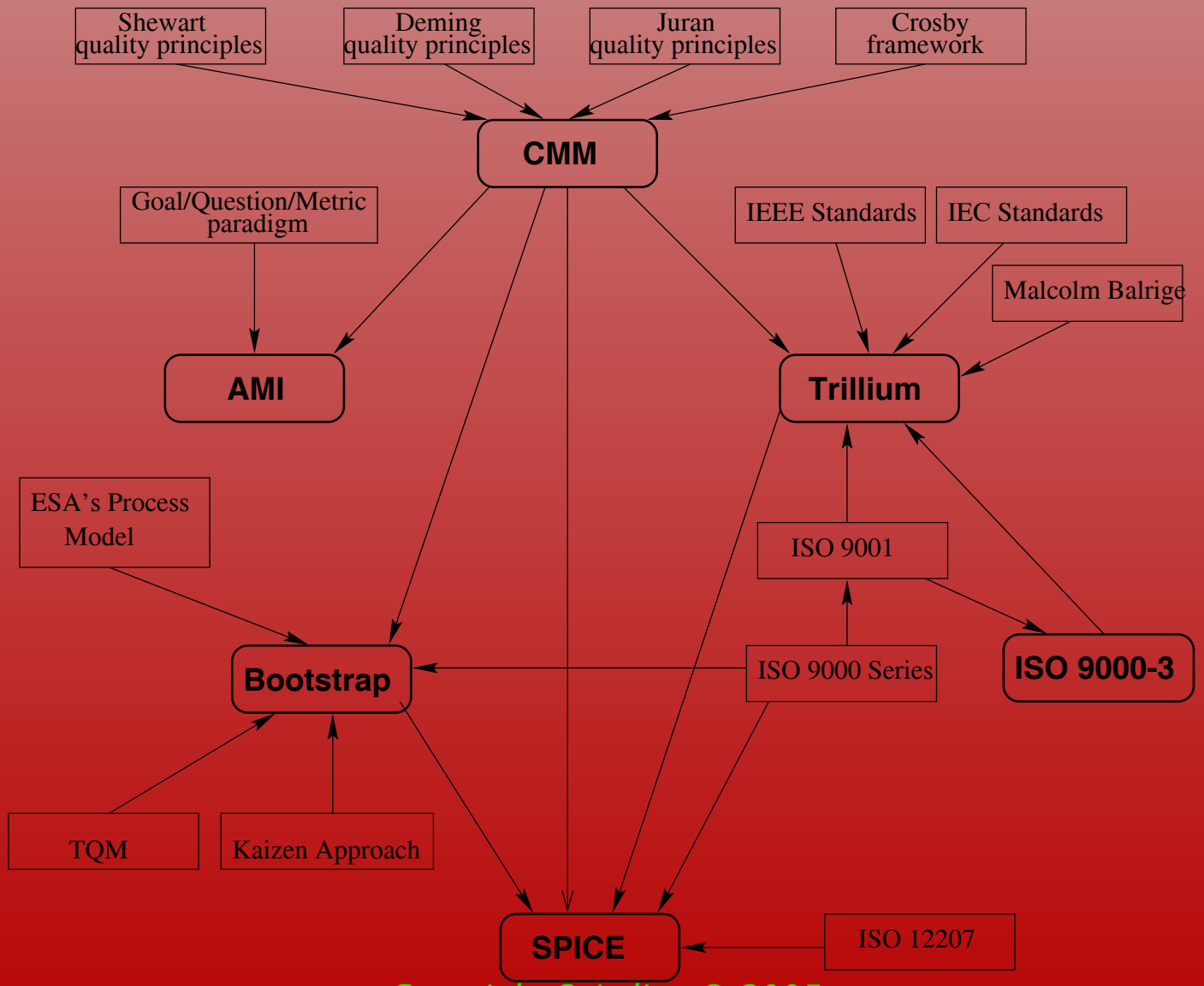
Uses of the CMM

- **CMM-Based Appraisal for Internal Process Improvement**
  - CBA-IPI is used by an organization to assess current practices and to improve software process
  - Results to organization only
  - As a catalyst for process improvement; provides input to improvement and action plan
  - Collaborative: organization members on the team
  - Apply to overall organization, not individual projects

Uses of the CMM (continued)

- **Software Capability Evaluations:**
  - Used by the acquisition organization for source selection and contract monitoring
  - Results to organization and acquire
  - Substantiates current practices
  - Analyze contract performance potentials
  - Independent evaluation: no organization members on team
  - Apply to a particular contract/project

CMM and Other SPI Models



Assessment Teams

- Independent or internal
- Qualification
  - Experienced
  - Respected
  - Good communication
  - Trained
- Confidentiality
- Conclude with a written report

Moving Up Levels

- Assessment team meets with management, technical people
- Meetings last about a week
- Assess, not audit
- Ask the right people the right questions
- Compliance is checked
- Level is *lowest* of all KPAs

Case Studies of Applying the CMM

- Studies based on 13 organizations (Herbsleb et al 1995)
- Organizations involved:
  - DoD contractors (e.g., Hughes Aircraft)
  - Commercial organizations (e.g., HP, AT&T, Bull HN, Schlumberger, TI)
  - Military organizations (e.g., OC-ALC)
- Data collected on organizational characteristics, SPI efforts, results of SPI efforts, other elements

Results of Case Studies

Category	Range	Median
Total yearly cost of SPI activities	$49,000–$1.2 M	$245,000
Years engaged in SPI	1–9	3.5
Cost of SPI per software engineer	$490–$2,004	$1,375
Productivity gain per year	9%–67%	35%
Early error detection gain per year	6%–25%	22%
Yearly reduction in time to market	15%–23%	19%
Yearly reduction in post-release	10%–94%	39%
Return-on-investment in SPI	4.0–8.8	5.0

A Specific Case: The Hughes Aircraft

- Organization description
  - Software Engineering Division, 500 Employees
  - US DoD Contracts
- SEI Efforts
  - First assessment in 1987 — Level 2
  - Second assessment in 1990 — Level 3
- Costs
  - Level 2 to Level 3: 75 person-months
  - $\approx$\$450,000 (\$400,000 + \$45,000)
- ROI: First year benefits: \$2,000,000

Lessons Learned

- Substantial increase in productivity (as much as 67%)
- Substantial improvement in quality
- High ROI (as high as 8 to 1)
- Management involvement is important
- Developing action plan is essential
- Full time personnel in SEPG is vital
- SEPG leadership is key to success
- Many intangible benefits (less stress, higher morale, few crisis)

Process improvement pays off and is cost-effective

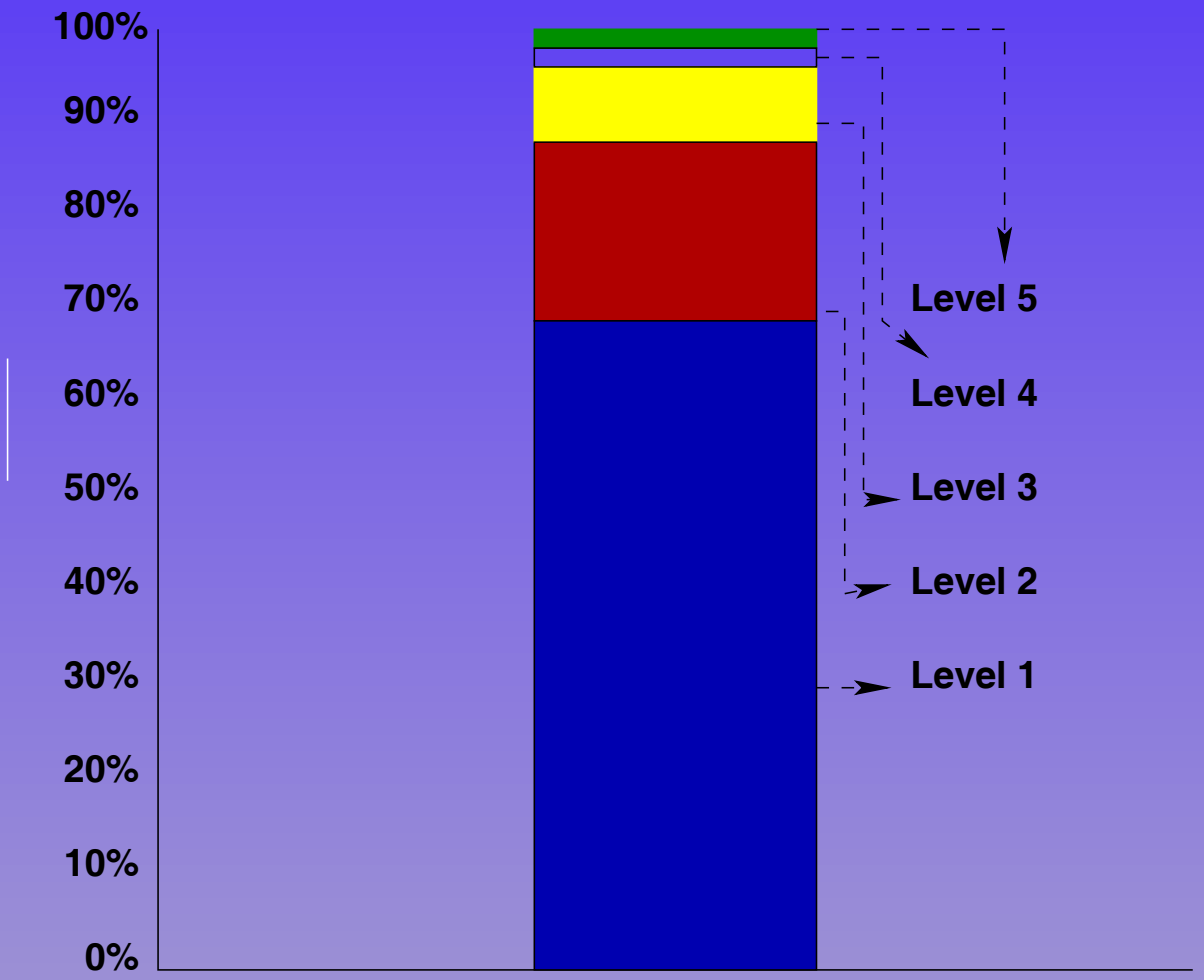
Industry Concerns

- High cost of assessment and re-assessment; small companies are at a disadvantage
- SCE's can be viewed as intrusion into previously "private" environment
- SCE results can vary
- CBA-IPI's and SCE's do not agree; CBA-IPI's rates higher
- Failures unlikely to be reported
- Real motivation: long-term process improvement or short-term business profits?
- Corporate policies, political reasons of obtaining a lower CMM rating (sand-bagging)

Levels Below 1?!?

- Level 0: Negligent, uncaring
- Level -1: Obstructive, resisting improvement
- Level -2: Contemptuous
- Level -3: Undermining; threatened heroes

Early 2000s Status



SPI/CMM Common Myths

- CMM is for high budget projects in large organizations only
- Maturity levels cannot be skipped. Each level must be successively achieved
- The CMM is usable for any organization as-is and does not need any customization
- CMM is only beneficial for long term improvements
- The CMM serves as a replacement for existing metrics models
- Once an SPI model is implemented, it is safe to trust that software quality will increase
- A part-time SEPG will be sufficient
- The CMM will eliminate people from the software development equation

Process Improvement Challenges

- Sponsorship from senior management; key champion from the lowest level of management
- Benefit realization: the time it takes to observe the benefits of process improvement
- SEPG: must be established based on the concept of *continuous improvement* (and provided the authority to make a difference)
- Cultural change

  "Someone tell me how do to do my job better?"

  "Document my job so that anyone can do it?"

  "Do a lot more work on my tight schedule?"

  "Process improvement is just a side job to fill a square"

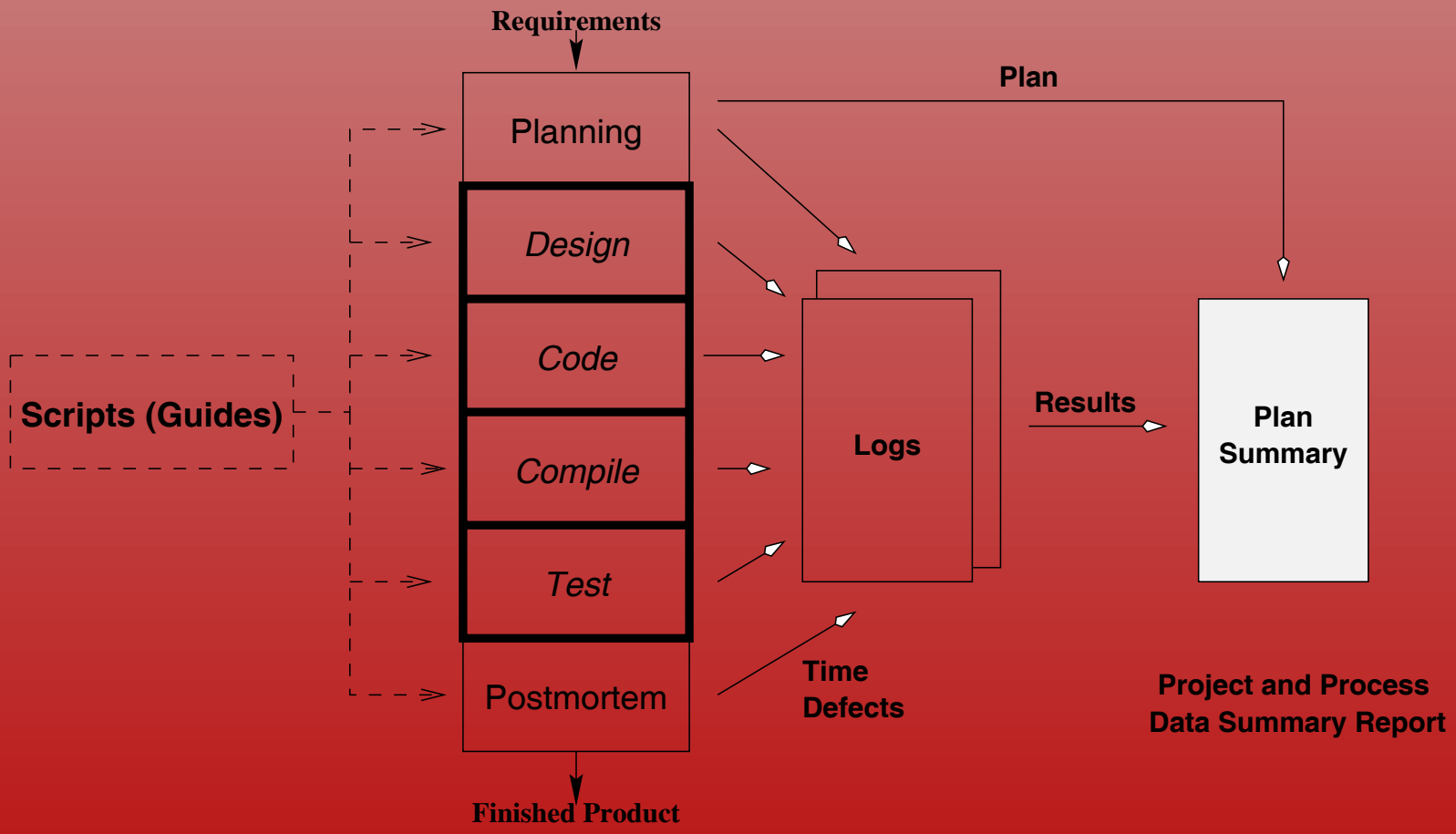
What is CMMI?

- Capability Maturity Model Integration
- Mix and match: Software (SW), System (SE), Supplier Sourcing (SS), People (P)
- Capability levels:
  - Incomplete (0)
  - Performed (1)
  - Managed (2)
  - Defined (3)
  - Quantitatively managed (4)
  - Optimizing (5)

Alternatives/Other Models

- ISO 9000 Quality Standards (ISO 9003 for Software)
- SPICE (Software Process Improvement and Capability dEtermination) (better known as ISO/IEC 15504)
- ITIL (IT Infrastructure Library): provides a framework for identifying, planning, delivering and supporting IT services
- Personal Software Process (PSP)
  - A self-improvement process designed for individual software engineers; helps control, manage, and improve individual work
  - Four levels of maturity (PSP0 through PSP3); Twelve KPAs

PSP Process Flow and Elements



Summary

- Absolutely essential to improve the quality of software
- To improve software quality, the software process must be improved
- To improve the process, we need process models and improvement models
- There are a number of such models, e.g., the SEI CMM
- Such models are based on traditional quality models
- Process improvement may be costly but its ROI is very high
- Many exciting opportunities