Lecture Slides for
Managing and Leading Software Projects

Chapter 1: Introduction

developed by

Richard E. (Dick) Fairley, Ph.D.

to accompany the text

Managing and Leading Software Projects

published by Wiley, 2009

Chapter 1 Topics

- Why Managing and Leading Software Projects Is
- Difficult
- The Nature of Project Constraints
- A Workflow Model for Managing Software Projects
- Organizational Structures for Software Projects
- Organizing the Project Team
- Maintaining the Project Vision and the Product Vision
- Frameworks, Standards, and Guidelines

Additional Information (1)

- Appendix 1A to Chapter 1 provides an introduction to elements of the following frameworks, standards, and guidelines that are concerned with managing software projects:
  - the SEI Capability Maturity Model® Integration CMMI-DEV-v1.2,
  - ISO/IEC and IEEE/EIA Standards 12207,
  - IEEE/EIA Standard 1058, and
  - the Project Management Body of Knowledge (PMBOK®).

Additional Information (2)

- Terms used in Chapter 1 and throughout the text are defined in Appendix A
- These presentation slides and other supporting material are available at the URL listed in the Preface to the textbook

Objectives for Chapter 1

- After reading Chapter 1 and completing the exercises, you should understand:
  - the main elements of software project management
  - the influence of project constraints
  - why managing and leading software projects is difficult
  - a workflow model for software projects
  - the work products of software projects
  - organizational structures for software projects
  - organizing a software development team
  - maintaining the project vision and product goals
  - frameworks, standards, and guidelines

What is a Project?

A project is characterized as follows:

- a one-time effort is planned
- starting and ending dates are prescribed
- a project team is assembled
- schedule and budget are allocated
- well-defined objectives are established
- roles are identified, responsibilities are assigned, and authority is delegated

Software projects are temporary organizational units

What is Management?

- Management is concerned with planning and coordinating the work activities of others so that they can achieve goals that cannot be achieved by each individual acting alone

Synergy: the combined effect is greater than the sum of the individual effects

What is Software Project Management?

Software Project Management (SPM) is the art and science of

- planning and coordinating the work of software developers and other personnel
- to develop and modify software artifacts
- that are pleasing to users and customers
- that are developed and modified in an economical and timely manner
- and that can be maintained efficiently and effectively

The Four Major Activities of SPM

1. Planning and Estimating
   - identify work activities
   - prepare a schedule
   - prepare a budget
2. Measuring and Controlling
   - requirements
   - quality and productivity
   - schedule and budget
   - product evolution
3. Leading and Communicating
   - motivating / coaching / educating project members
   - communicating with management, customers, subcontractors, other projects
4. Managing Risk
   - identifying and confronting potential problems

Managing versus Leading

- Managing is concerned with the quantitative aspects of SPM:
  - planning and estimating
  - measuring and controlling
  - quantitative risk management
- Leading is concerned with the qualitative aspects of SPM:
  - communicating and coordinating
  - inspiring and maintaining morale
  - qualitative risk management

an effective project manager is both a manager and a leader

Project Success Criteria

- The primary goal of software engineering is to develop and modify software so that:
  - the product is delivered on time & within budget
  - the product satisfies technical requirements, user needs, and customer expectations
  - the product is easy to modify and maintain
  - development milestones are achieved on time & within budget
  - staff morale is high throughout project
  - work instills pride in the developers

Q1: What are your personal success criteria?
Q2: What are most organizations' main success criteria?
Q3: What are most customers' main success criteria?

Project Manager's Success Criteria

A project manager's success criteria include, or should include:

- delivery of an acceptable product on time and within budget
  - within the limits imposed by *project constraints**
- maintaining good relations with customers, suppliers, managers, and other organizational units
- maintaining a motivated project team
- advancing the career of each project member
- advancing his or her career
- Other criteria?

*a *constraint* is an eternally imposed limitation

Why Are Software Projects Difficult?

- According to Fred Brooks* software projects are difficult because of accidental and essential difficulties
  - accidental difficulties are caused by the current state of our understanding
    - of methods, tools, and techniques
    - of the underlying technology base
  - essential difficulties are caused by the inherent nature of software

* *The Mythical Man-Month* by Fred Brooks, Addison Wesley, 1995

Essential Difficulties

- complexity,
- conformity,
- changeability, and
- invisibility of software.

Additional Difficulties

- Additional reasons software projects are difficult are:
  - intellect-intensive, team-oriented nature of the work
  - externally imposed constraints

Intellect-Intensive Teamwork

- Software is developed by:
  - teams of individuals
  - engaged in closely coordinated intellectual work activities
  - to produce various written work products

An Observation

- As Michael Jackson has observed, the entire description of a software system or product is usually too complex for the entire description to be written directly in a programming language, so we must prepare different descriptions at different levels of abstraction, and for different purposes*.
- Also, note that each of the work products listed on the following slide is a document
  - software developers and software project managers do not produce physical artifacts other than documents, which may exist in printed or electronic form.

* M. Jackson, "Descriptions in Software Development," *Lecture Notes in Computer Science*, Springer Verlag GmbH, Volume 2460, 2002.

Some Work Products of Software Projects

Document	Content of the document
Project plan	Roadmap for conducting the project
Status reports	Visibility of progress, cost, schedule, and quality
Memos and meeting minutes	Issues, problems, recommendations, resolutions
e-mail messages	On-going communications
Operational requirements	User needs, desires, and expectations
Technical specification	Product features and quality attributes
Architectural design document	Components and interfaces
Detailed design specification	Algorithms, data structures, and interface details of individual modules
Source code	Product implementation
Test plan	Product validation criteria and test scenarios
Reference manual	Product encyclopedia
Help messages	Guidance for users
Installation instructions	Guidance for operators
Release notes	Known issues; hints and guidelines
Maintenance guide	Guidance for maintainers

Note

- Note that the work products generated by software engineers exist in graphical, iconic, and textual forms
  - software engineers do not design or fabricate artifacts made of physical materials
  - our work products are generated from our thought processes

A Simile

- A team that writes software together is like a team that writes a book together
  - the team may pursue a “plan-driven” approach
  - or an “agile” approach

Plan-Driven Development

- Plan-driven development involves:
  - defining the product requirements
  - developing an architectural structure for the product
  - allocating the work among teams
  - measuring progress and making corrections
  - refining and revising the work products as necessary
    - preferably in an iterative manner

Agile Development

- When pursuing an agile approach, the team members must:
  - develop an understanding of the nature of the desired product to be delivered,
  - develop continuous, ongoing relationship with a knowledgeable user representative
  - establish a shared design metaphor,
  - adopting a version of agile development, and
  - determine the constraints on schedule, budget, resources, and technology that must be observed.

most successful software projects incorporate aspects of both planning and agility

Additional Difficulties

- Additional reasons software projects are difficult are:
  - intellect-intensive nature of the work
  - externally imposed constraints

Engineering Constraints

- Engineering is concerned with applying science and technology to develop products for use by society within the *constraints* of:
  - product requirements: features and quality attributes
  - project scope: work activities to be accomplished
  - time: scheduled dates for progress
  - resources: assets available to conduct a project
  - budget: money used to acquire resources

Additional Constraints

- Additional limitations imposed on software projects include:
  - platform technology: software tools and hardware/software base
  - domain technology: the realm of the user domain
  - process standards: ways of conducting work activities
  - scientific knowledge: solution methods
  - business considerations: profit, stability, growth
  - mission needs: safety and security of citizens
  - ethical considerations: serving best interests of humans and society
- Others?

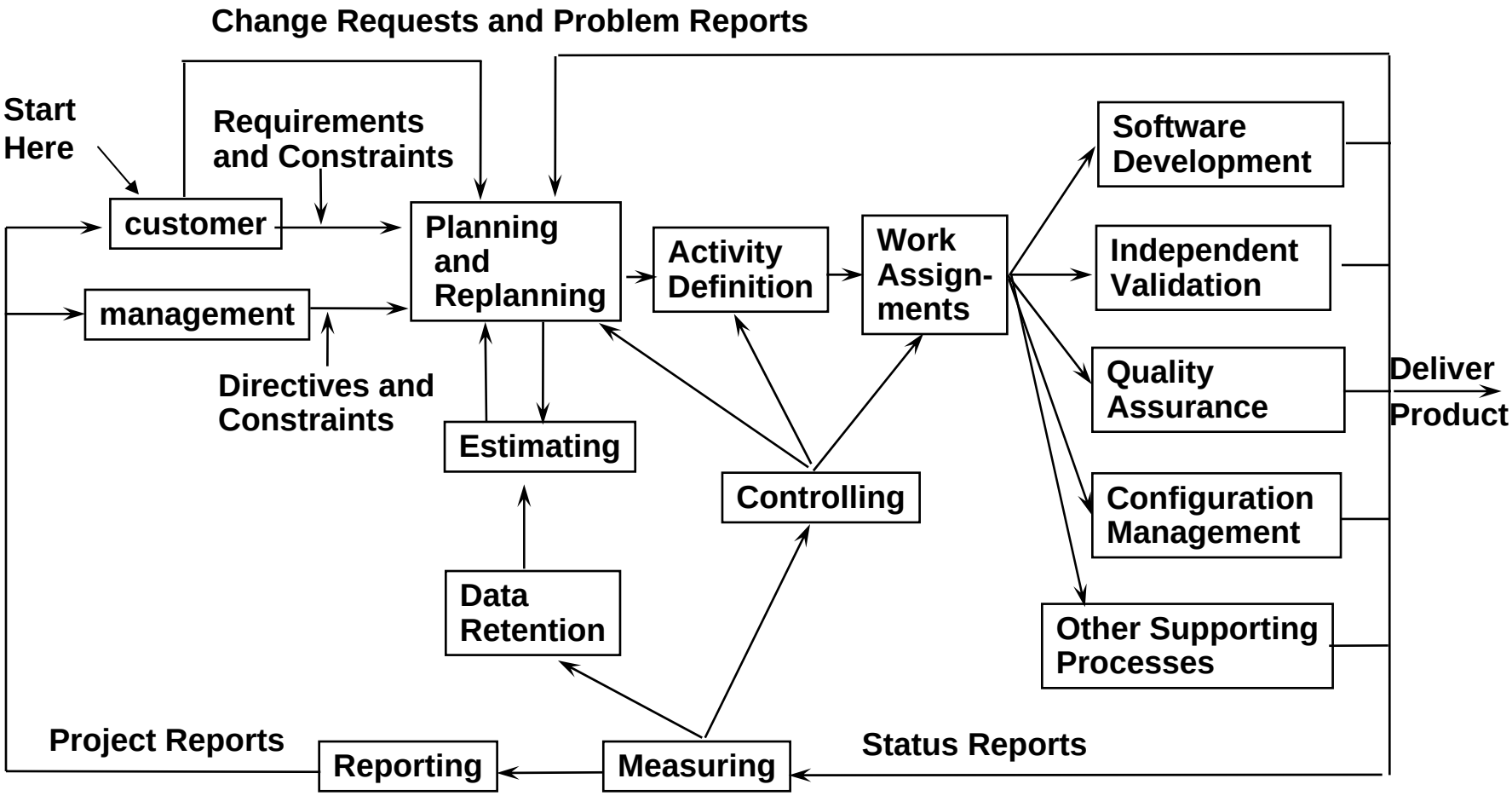
Useful Constraints and Inhibiting Constraints

- Useful constraints provide guidance:
  - for example, well-defined requirements are the basis for planning, estimating, and establishing success criteria
- Inhibiting constraints inhibit the ability to achieve success criteria:
  - for example, excessive schedule pressure may inhibit the ability to delivery a product of high quality

The Challenges of Software Project Management

- Some of the most difficult problems you will encounter in managing software projects arise from establishing and maintaining a balance among the constraints on project scope, budget, resources, technology, and the scheduled delivery date:
  - scope: the work to be done,
  - budget: the money to acquire resources,
  - resources: the assets available to do the job,
  - technology: methods and tools to be used, and
  - delivery date: the date on which the system must be ready for delivery.

A Workflow Model for Software Projects



Some Elements of the Model

- Customers and managers
- Requirements
- Directives and constraints
- Planning and re-planning
- Estimating
- Identifying the work activities and work assignments
- Conducting the work activities
- Measuring and reporting status
- Controlling the project
- Retaining status data
- Handling change requests and problem reports
- Supporting processes

Some Supporting Processes for Software Projects

Supporting Process	Purpose
Configuration Management	Change control; baseline management; product audits; product builds
Verification	Determining the degree to which work products satisfy the conditions placed on them by other work products and work processes
Validation	Determining the degree of fitness of work products for their intended use in their intended environments
Quality Assurance	Assuring conformance of work processes and work products to policies, plans, and procedures
Documentation	Preparation and updating of intermediate and deliverable work products
Developer Training	Maintaining adequate and appropriate skills
User and Operator Training	Imparting skills needed to effectively use and operate systems

Eight Supporting Processes in ISO & IEEE Standards 12207

- Documentation
- Configuration management
- Quality assurance
- Verification
- Validation
- Joint review
- Audit
- Problem resolution

A Note on Terminology

- In many organizations the term “software quality assurance (SQA)” is used to mean independent testing
- In the 12207 standards quality assurance is concerned with:

  “providing adequate assurance that the software products and processes in the project life cycle conform to their specified requirements and adhere to their established plans.”
- Testing is in the realm of Verification and Validation
  - independent testing should not be termed “QA” or “SQA”

CMMI-DEV-v1.2 uses terminology similar to 12207 more later

Managing The Supporting Processes

- In a project-format organization all of the supporting processes (except SQA) are managed by the project manager
- In a functional-format organization all of the supporting processes (including SQA) are managed by the functional managers
- In a matrix-format organization most of the supporting functions (except SQA) are managed by the project manager
- In a hybrid-format organization, some or all of the supporting processes (including SQA) are provided by functional groups within the organization
- QA is always (or should be) conducted by a separate group

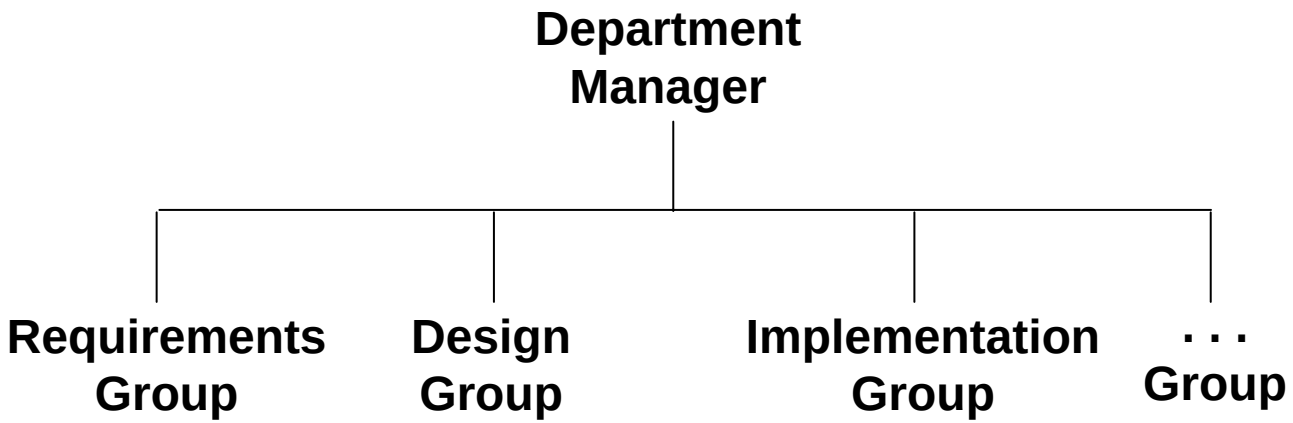
Chapter 1 Topics

- Why Managing and Leading Software Projects Is
- Difficult
- The Nature of Project Constraints
- A Workflow Model for Managing Software Projects
- *Organizational Structures for Software Projects*
- Organizing the Project Team
- Maintaining the Project Vision and the Product Vision
- Frameworks, Standards, and Guidelines

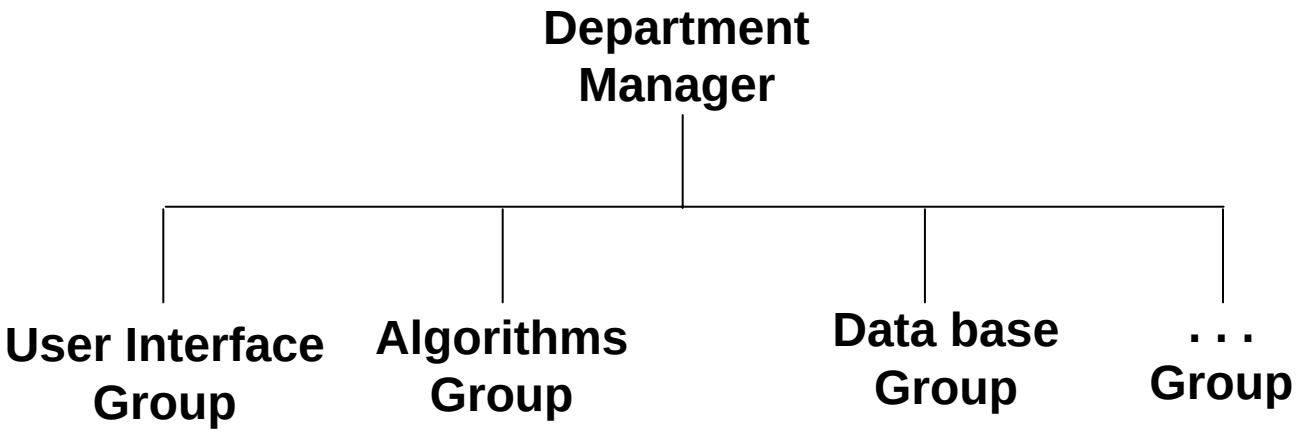
Organizational Structures

- Organizations that conduct engineering projects, including software projects, are typically organized in one of four ways:
  - functional structure,
  - project structure,
  - matrix structure, or
  - hybrid structure.

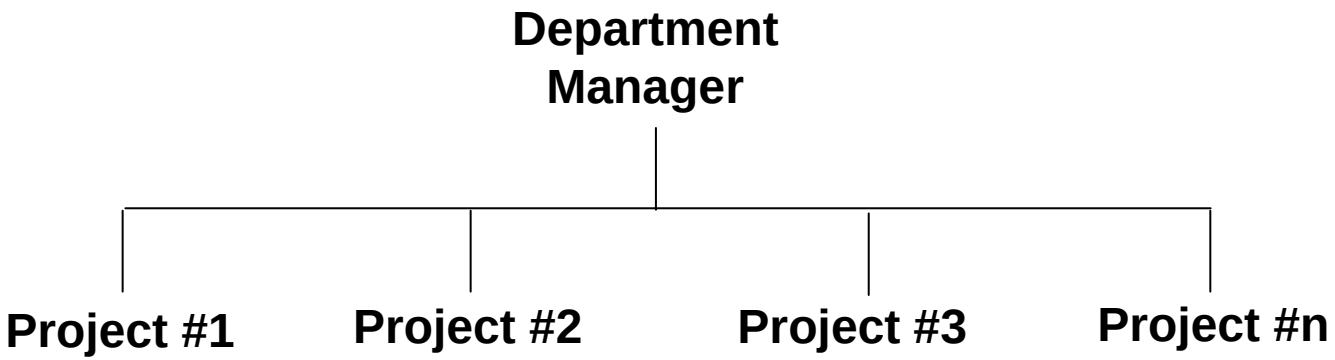
A Process-Structured Functional Organization



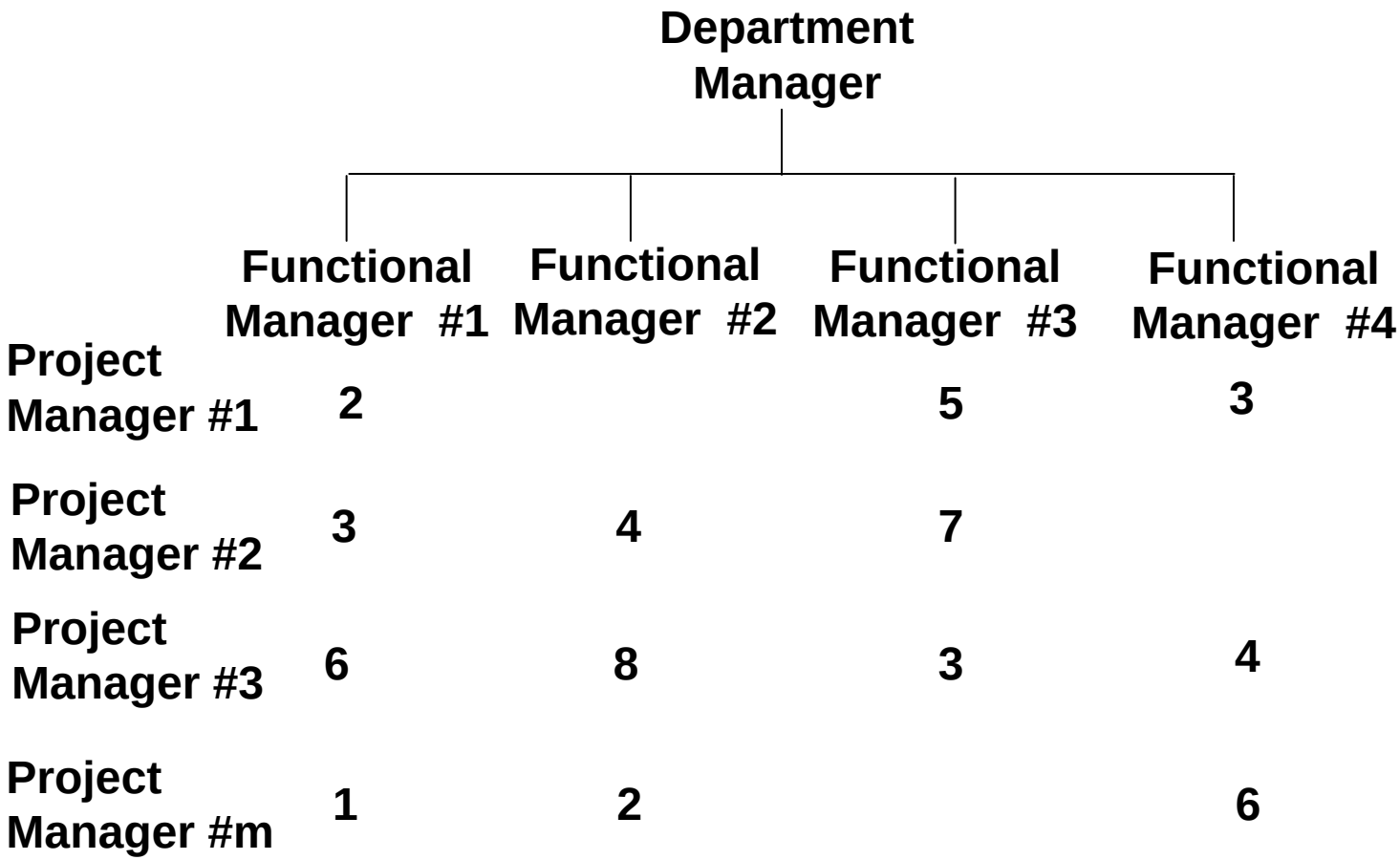
A Product-Structured Functional Organization



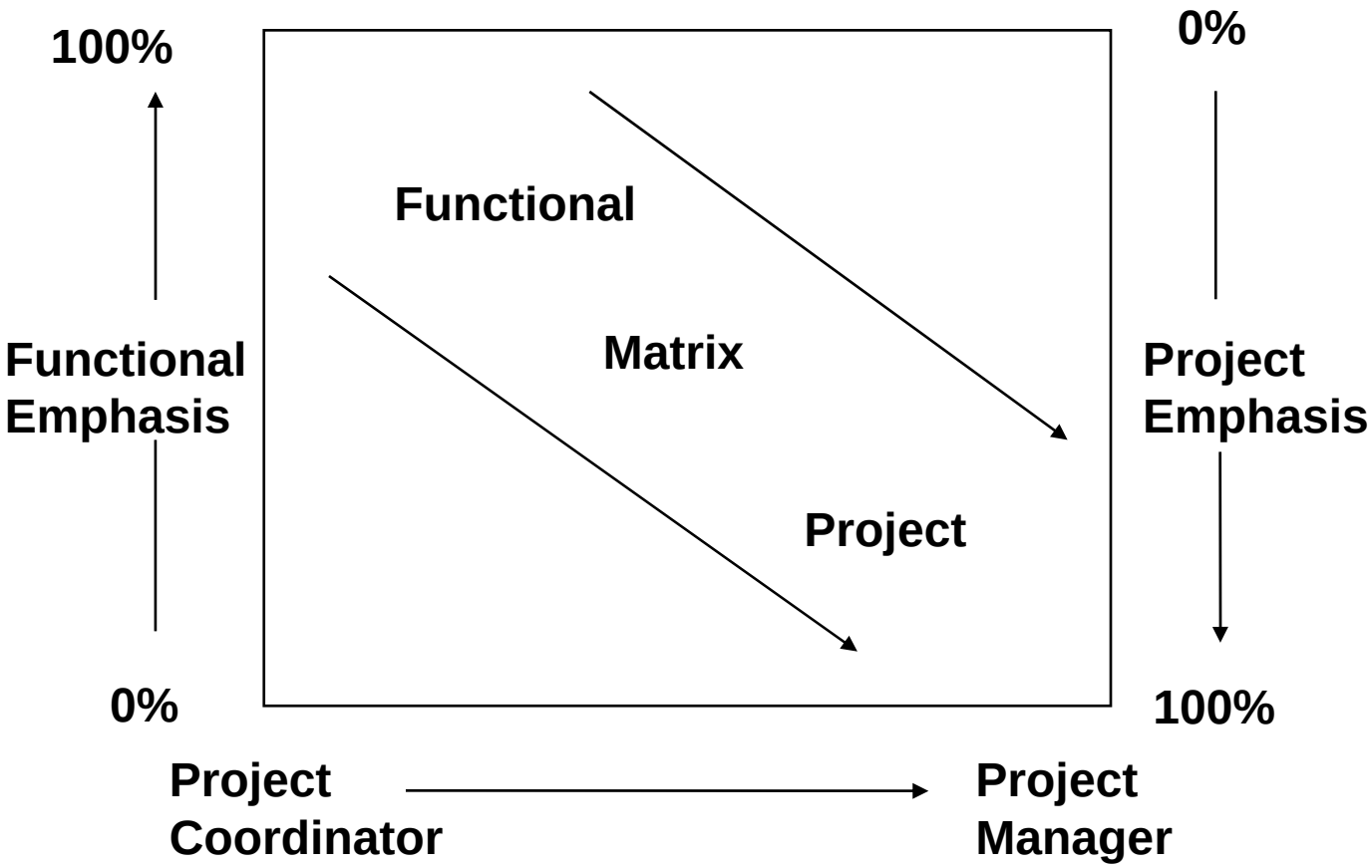
A Project-Structured Organization



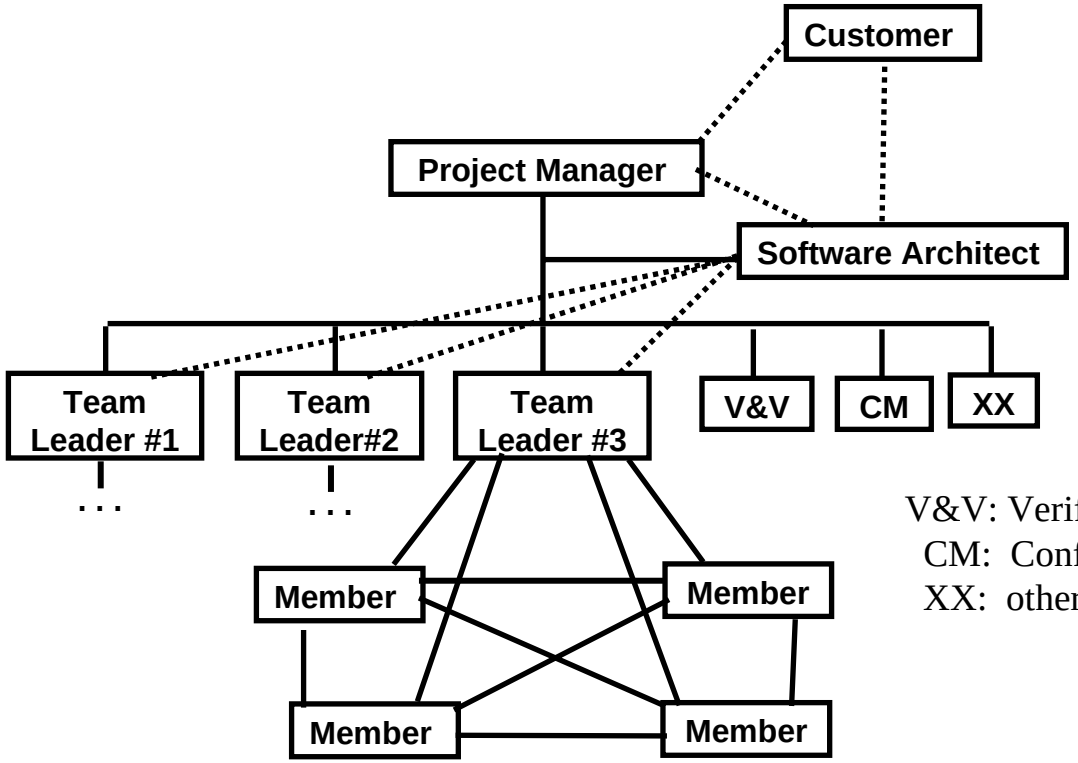
A Matrix-Structured Organization



The Organizational Continuum



An Organizational Model for Software Projects



V&V: Verification and Validation
CM: Configuration Management
XX: other supporting processes

Each team has 2 to 5 members plus a team leader

A Note

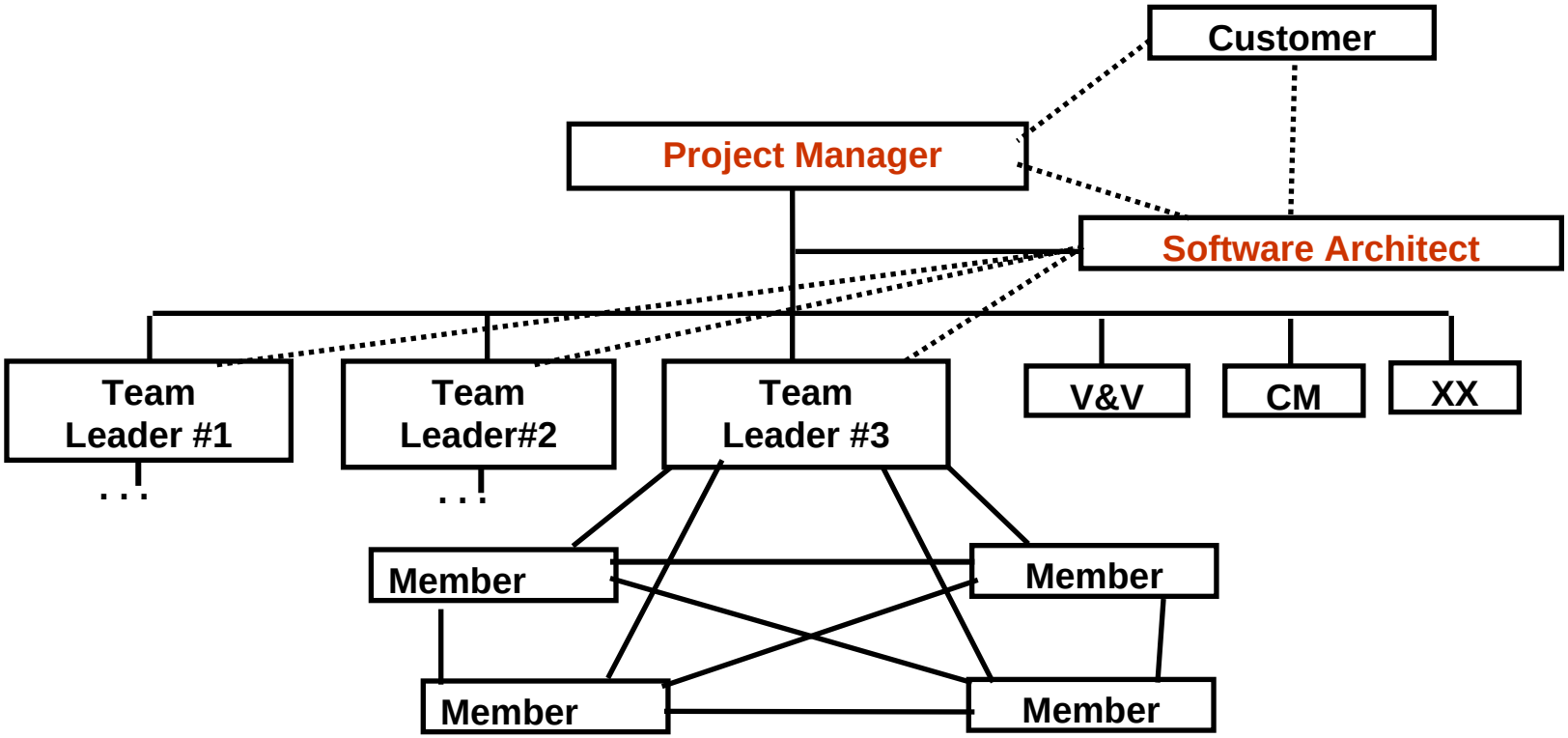
- A complex system is composed of:
  - hardware (computers and others)
  - software (newly developed and reused)
  - people (operators, maintainers)
- A software project may be one of a collection of projects
  - under the technical direction of a system engineering team

The System Engineering Team

- The responsibilities of systems engineers include:
  - defining operational requirements,
  - specifying system requirements,
  - developing the system design,
  - allocating system requirements to system components,
  - integrating the system components as they become available,
  - verifying that the system to be delivered is correct, complete, and consistent with respect to its technical specifications, and
  - validating operation of the system with its intended users in its intended operational environment.

for “software only” projects the people who perform these functions are termed “software system engineers”

An Organizational Structure for Software Projects



V&V: Verification and Validation
CM: Configuration Management
XX: other supporting processes

Each team has 2 to 5 members plus a team leader

Maintaining the Project and Product Visions

- The project manager is the the keeper of the process vision
  - which is documented in the project plan
    - and is updated as the project evolves
- The software architect is the keeper of the product vision,
  - which is documented in the requirements and architectural design specifications
    - and is updated as the product evolves

Another Simile

- The project manager is like a movie producer and the software architect to a movie director.
  - The producer (project manager) has overall responsibility for schedules, budgets, resources, customer relations, and delivery of a satisfactory product on time and within budget.
- The director (software architect) is responsible for the content of the product.

Producer and director must work together to maintain and constantly communicate the process vision and the product vision to the cast of developers and supporting personnel as well as other project stakeholders

Frameworks, Standards, and Guidelines (1)

- A *process framework* is a generic process model that can be tailored and adapted to fit the needs of particular projects and organizations.
- An *engineering standard* is a codification of methods, practices, and procedures that is usually developed and endorsed by a professional society or independent agency.
- *Guidelines* are pragmatic statements of practices that have been found to be effective in many practical situations.

Frameworks, Standards, and Guidelines (2)

- Some well known frameworks, standards, and guidelines for software engineering and the associated URLs are:
  - the Capability Maturity Model® Integration for development (CMMI-DEV-v1.2) [www.sei.cmu.edu/cmmi/models]
  - ISO/IEC and IEEE/EIA Standards 12207 [www.iso.org]; [standards.ieee.org/software]
  - IEEE/EIA Standard 1058 [standards.ieee.org/software]
  - the Project Management Body of Knowledge (PMBOK®) [www.pmibookstore.org]
- Elements of these models that are relevant to managing and leading software projects are presented in appendices to the chapters of this text, including Appendix 1A to this chapter.

The Main Points of Chapter 1 (1)

- A project is a coordinated set of activities that occur within a specific timeframe to achieve specific objectives
- The primary activities of software project management are planning and estimating; measuring and controlling; leading, communicating, and coordinating; and managing risk
- Software projects are inherently difficult because software is complex, changeable, conformable, and invisible
- Software projects are conducted by teams of individuals who engage in intellect-intensive teamwork
- Project constraints are limitations imposed by external agents on some or all of the operational domain, operational requirements, product requirements, project scope, budget, resources, completion date, and platform technology
- A workflow model indicates the work activities and the flow of work products among work activities in a software project

The Main Points of Chapter 1 (2)

- The entire description of a software system or product is usually too complex for the entire description to be written directly in a programming language, so we must prepare different descriptions at different levels of abstraction, and for different purposes
- Organizations that conduct software projects use functional, project, weak matrix, and strong matrix structures
- Software projects organized in a hierarchical manner provide well-defined work activities, roles, authorities, and responsibilities at each level in the hierarchy; hierarchies can expand and shrink to fit the needs of each project
- Requirements must be allocated and the design structured so that the work of each small team can proceed concurrently with the work of other teams

The Main Points of Chapter 1 (3)

- The project manager maintains the project vision, as documented in the project plan, and the software architect maintains the product goals, as documented in the requirements and architectural design
- A software process framework is a generic process model that can be tailored and adapted to fit the needs of particular projects and organizations.
- A software engineering standard is a codification of methods, practices, and procedures, usually developed and endorsed by a professional society or independent agency.
- Guidelines are pragmatic statements of practices that have been found to be effective in many practical situations.
- SEI, ISO, IEEE, and PMI provide process frameworks, standards, and guidelines that contain information relevant to managing software projects (see Appendix 1A to Chapter 1)