Software Project Management Part 1: Organization

Introduction into Software Engineering
Lecture 19

Bernd Bruegge
Applied Software Engineering
Technische Universitaet Muenchen

Software Engineering is Problem Solving

- Analysis: Understand the nature of the problem and break the problem into pieces
- Synthesis: Put the pieces together into a large structure that prepares for the solution
- Technical aspects of the problem solving process:
  - Techniques
  - Methodologies
  - Tools
- Where does project management come in?
  - When the available resources to solve the problem are limited (time, people, budget), or
  - When we allow the problem to change.

Software Engineering: Definition

- Software Engineering is a collection of techniques, methodologies and tools that support the development of a high quality software system
  - within a given budget
  - before a given deadline
  - while change occurs.

Example: Running a rapid

A quiet river

We are on our way…
Then suddenly

How could this happen to us?

Change

- Something becomes clear ("it crystalizes")
- Something new appears ("technology enabler")
- Something becomes important ("change of requirements")
- A process can also change ("process change")
  - Examples:
    - Developers have the power to make decisions ("agile teams")
    - Work is performed where it makes sense: ("outsourcing").

Change can happen fast

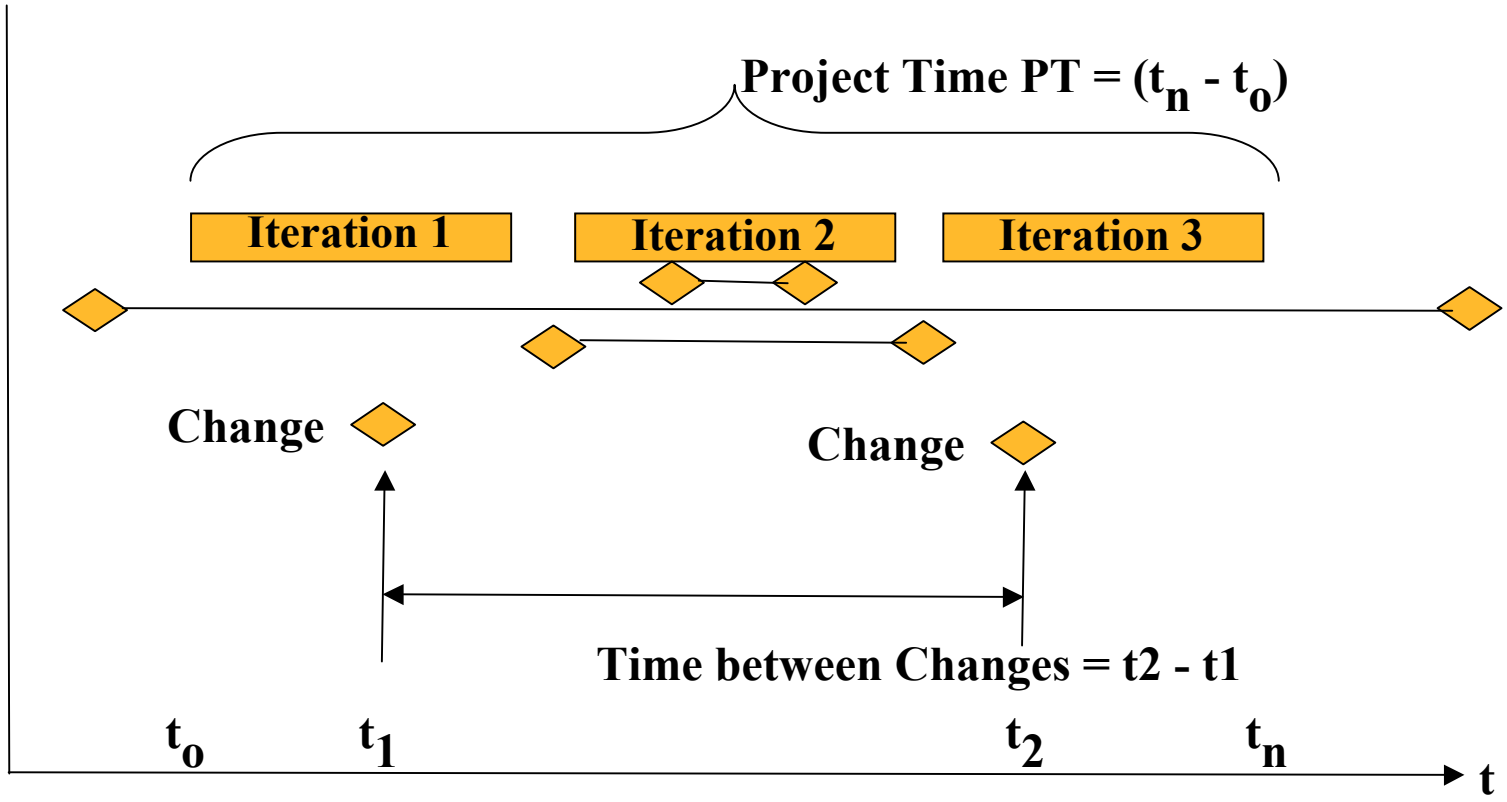
- Most of the important changes are unexpected
  - Frederick Brooks, The Mythical Man Month.
- Manager must anticipate and react to unusual technology happenings
  - Wayne Gretzky: " I go where the puck is going to be, not where it is"
  - Hammer (Reengineering): "Change is the only thing that is constant".

Changes in the Business of Software Development

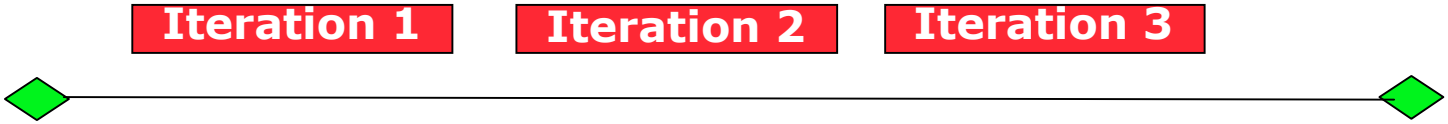
- Combination of several jobs into one job
- Developers have the power to make decisions
- Processes have multiple versions
- Checks and controls reduced
- Work performed where it makes sense
  - Outsourcing.

Project Duration vs. Rate of Change

- **PT** = Project Time
- **TBC** = Time Between Changes (Requirements, Technology)
- **MTBC** = Mean Time Between Changes



Rate of Change determines the Process

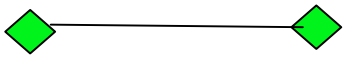


MTBC >> Project Time

Change is rare

Sequential process: Waterfall, V-model

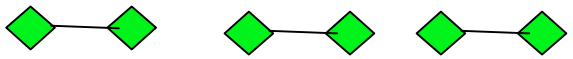
MTBC ≈ Project Time



Change will happen during the project

Iterative process: Spiral model, unified process

MTBC << Project Time



Change is frequent

Entity-based process: agile, empirical process control model.

Management vs. Project Management

Management: Getting a task done through people

Management is usually defined in terms of functions:

- Planning, organizing, directing, controlling and communicating are typical management functions
- No specific context (going on vacation, flying a plane)

- Project management: Activities in the context of a project
  - Project management tries to accomplish a specific task within a time frame and limited resources
- Software project management: Activities in the context of a software project
  - Activities to develop a software system within a given time frame and with limited resources.

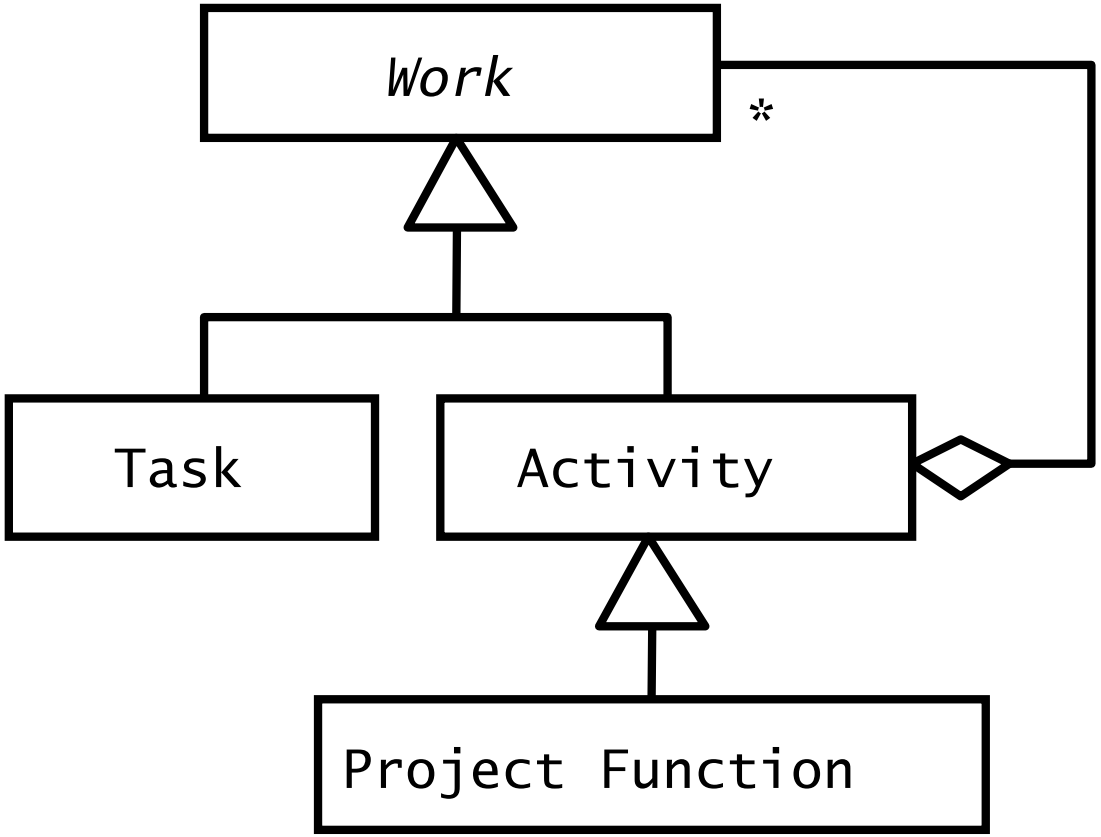
Outline of the lecture 7 04 2007

- Basic definitions: Project, Project Plan
- Status of a Project
- Software Project Management Plan
  - Project Organization
  - Managerial Processes
  - Technical Processes
  - Work Packages
- Typical Project Management Problems.

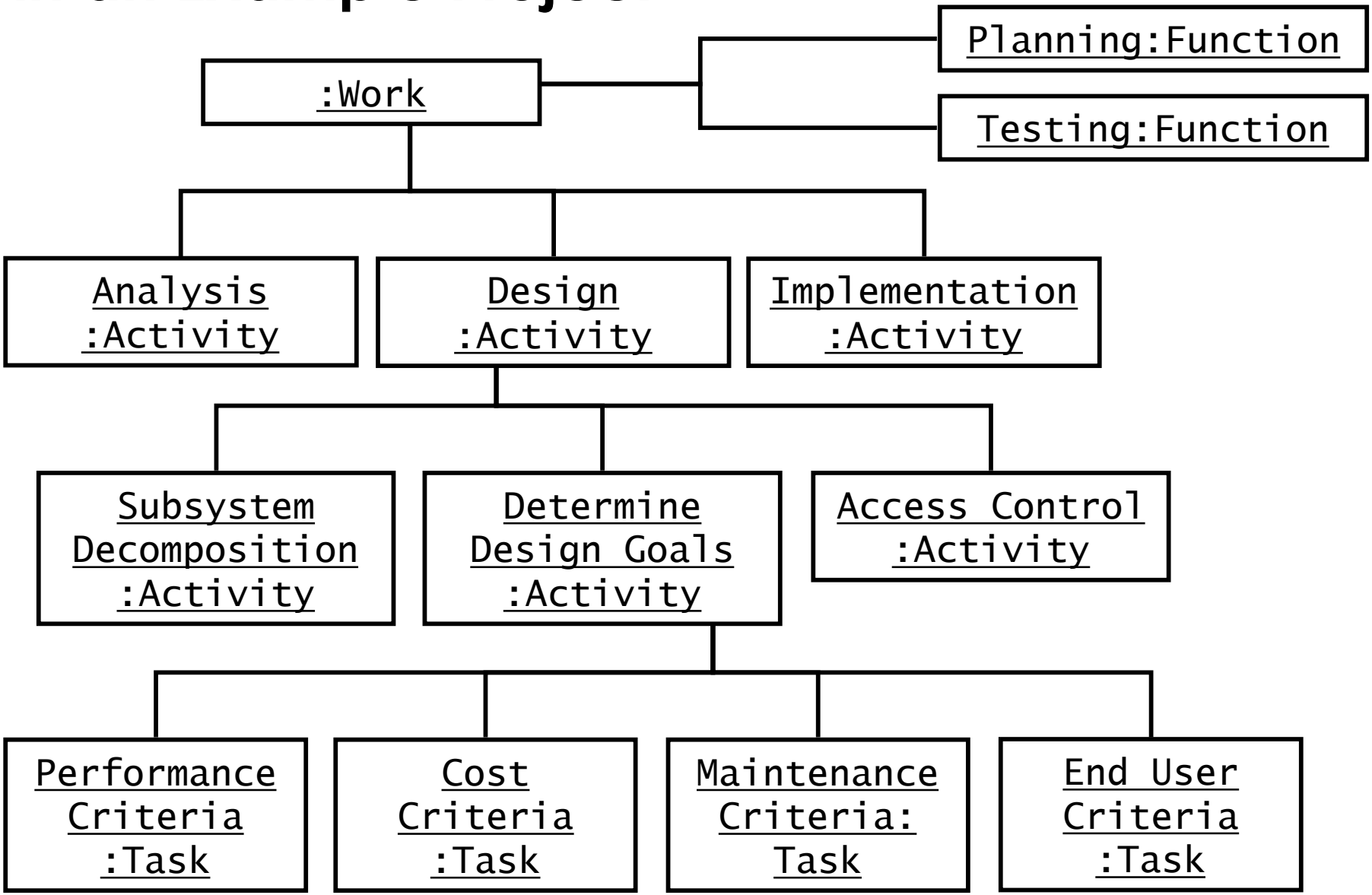
Basic Definitions

- Software Project:
  - All technical and managerial activities required to produce a set of deliverables for a client
  - A software project has a specific duration, consumes resources and produces work products
  - Management categories to produce work products in a software project:
    - Tasks, Activities, Project Functions
- Software Project Management Plan (SPMP):
  - The controlling document for a software project
  - Specifies the technical and managerial approaches to develop the software product
  - Companion document to requirements analysis document
    - Changes in either may imply changes in the other document.

Work Categories in a Software Project: Functions, Activities and Tasks

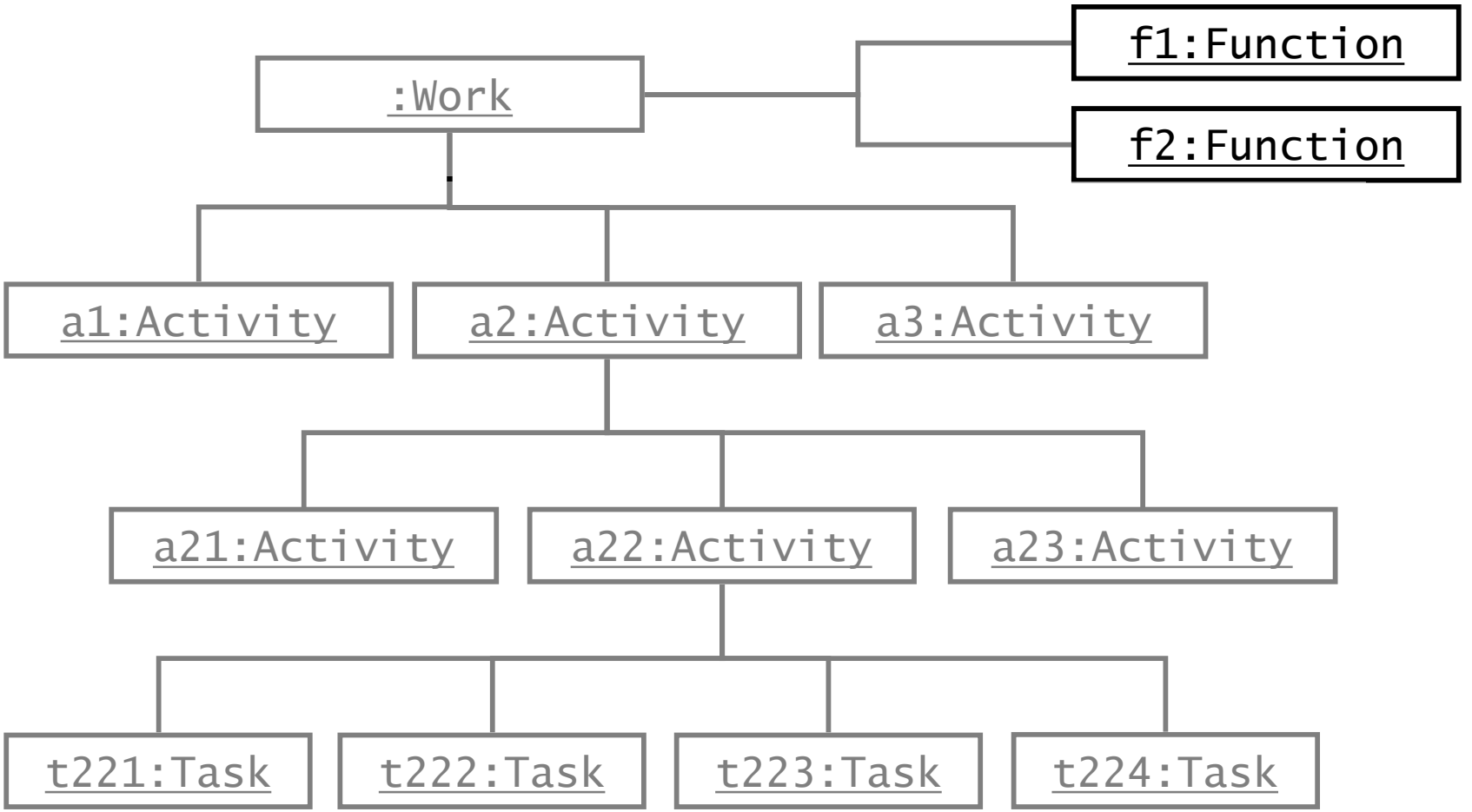


Instance Diagram for Work in an Example Project



Project Function

- **Definition** Project Function: An activity or set of activities that span the duration of the project



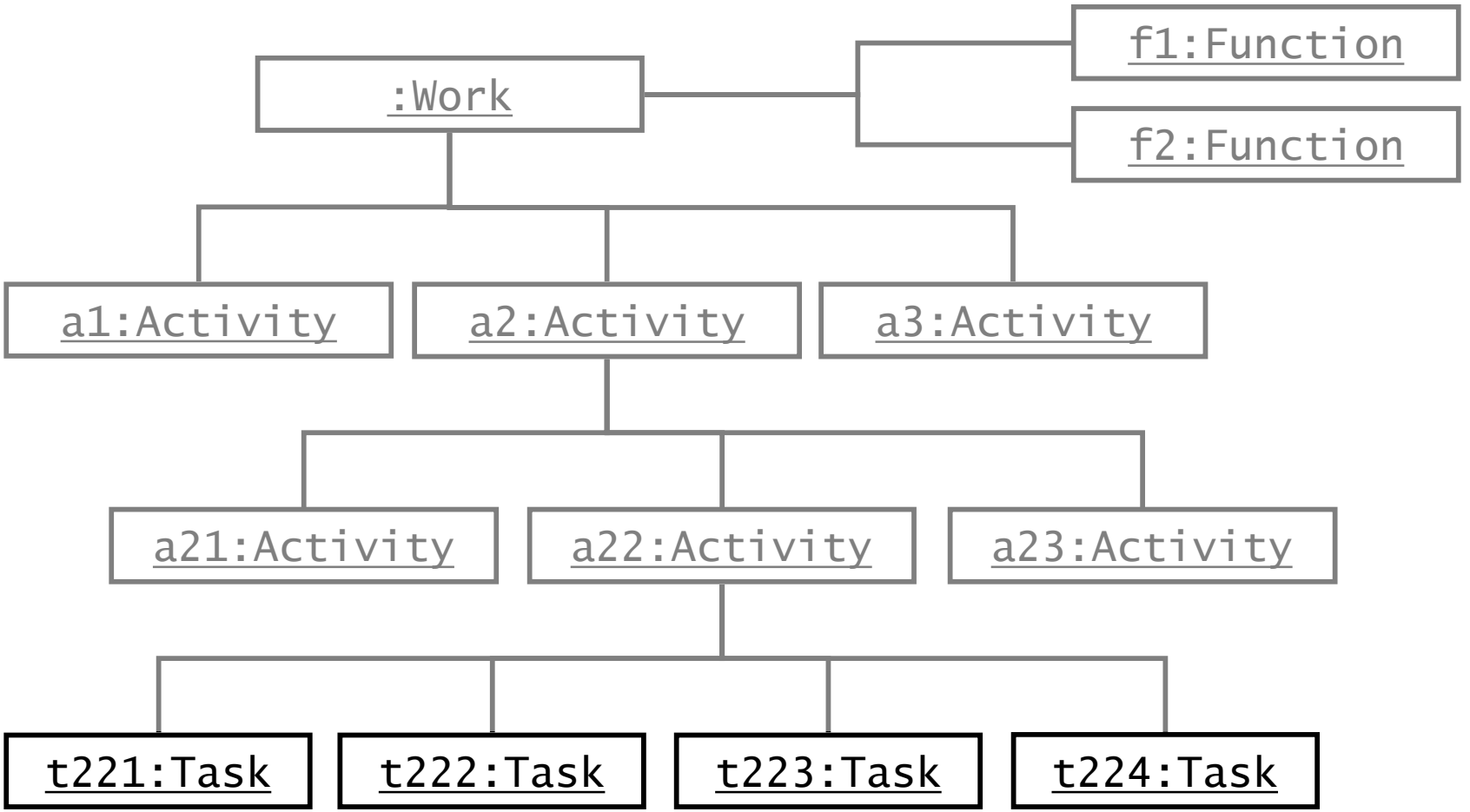
Examples of Project Functions

- Project function is the official name in the IEEE 1058 standard.Different names:
  - Integral processes (in the IEEE 1074 standard)
  - Sometimes also called cross-development processes.
- Examples:
  - Configuration Management
  - Documentation
  - Quality Control (V&V: Verification and validation)
  - Training
  - Testing
  - Project management activities

Slide 48-49
Examples of Management activities

Tasks

- Smallest unit of work subject to management
- Small enough for adequate planning and tracking
- Large enough to avoid micro management



Tasks

- Smallest unit of management accountability
  - Atomic unit of planning and tracking
  - Tasks have finite duration, need resources, produce tangible result (documents, code)
- Specification of a task: Work package
  - Name, description of work to be done
  - Preconditions for starting, duration, required resources
  - Work product to be produced, acceptance criteria for it
  - Risk involved
- Completion criteria
  - Includes the acceptance criteria for the work products (deliverables) produced by the task.

Determining Task Sizes

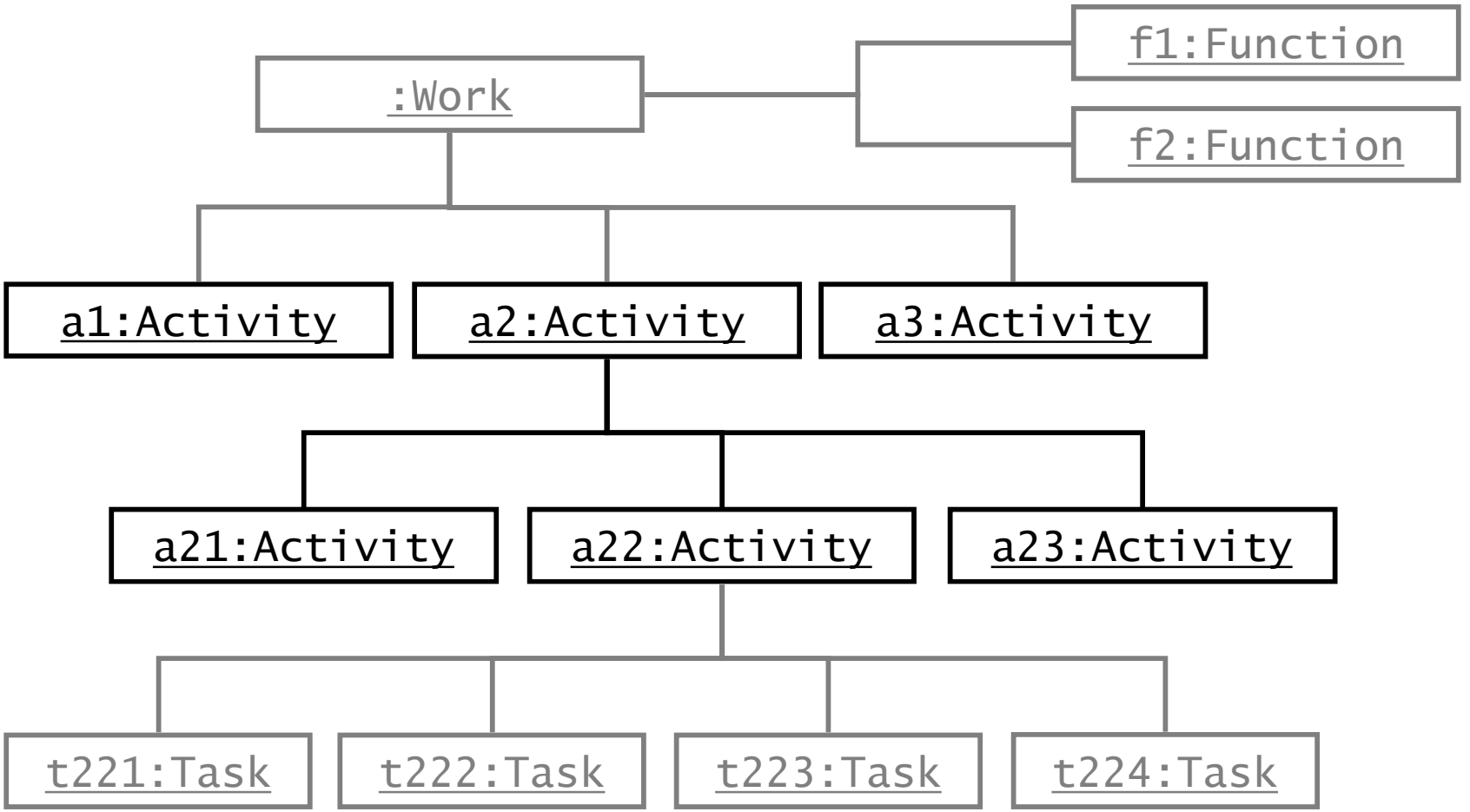
- Finding the appropriate task size is difficult:
  - During initial planning a task is necessarily large
  - You may not know how to decompose the problem into tasks at first
  - Each software development activity identifies more tasks and modifies existing ones
  - ☺Reuse Todo lists from previous projects
- Tasks must be decomposed into sizes that allow monitoring
  - Decomposition depends on nature of work and how well the task is understood.
  - Should correspond to a well defined work assignment for one participant for a week
    - ➢ Action item.

Action Item

- **Definition** Action Item: A task assigned to a person to be done by a certain time
  - What?, Who?, When?
  - Heuristics for duration: One week
- Definition Todo: An action item that is missing either the person or the deadline
- Examples of Todos:
  - Unit test class Foo, develop project plan
- Example of Action Items:
  - Bob posts the next agenda for the context team meeting before Sep 10, 12 noon
  - The testing team develops the test plan by Oct 21.

Activities

- Major unit of work with precise dates
- Consists of smaller activities or tasks
- Culminates in project milestone.



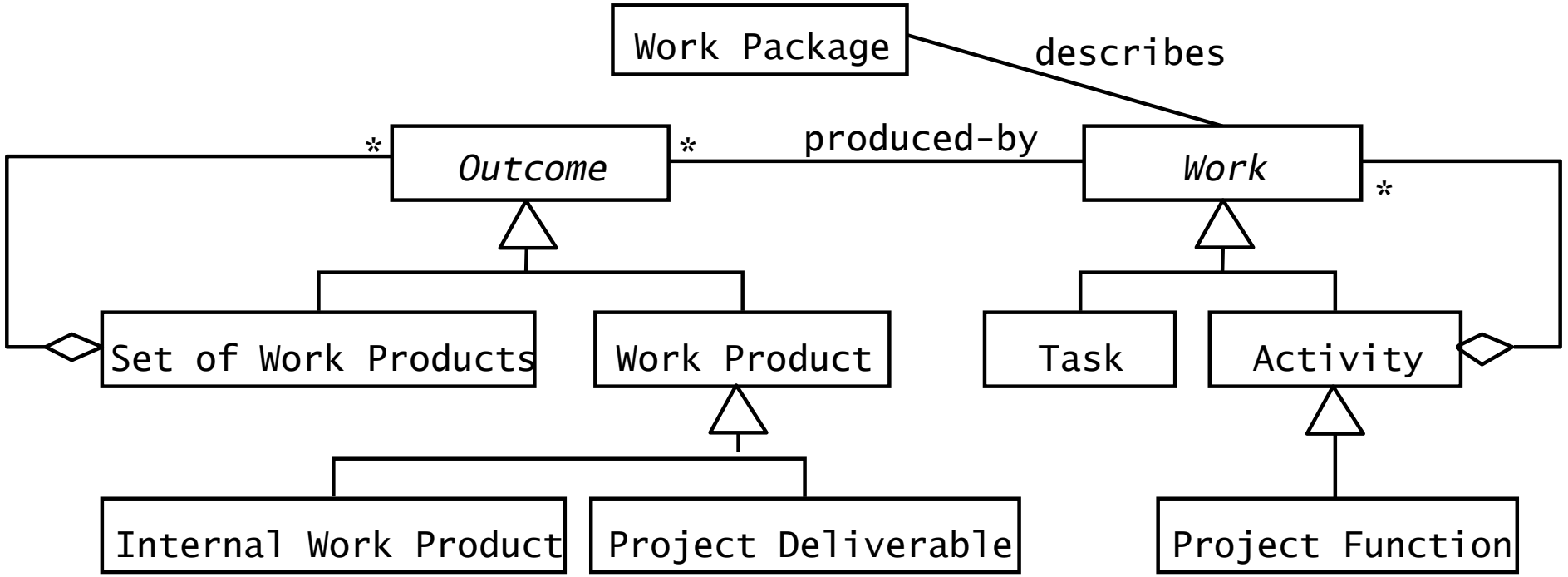
Activities

- Major unit of work 
- Culminates in major project milestone:
  - Internal checkpoint should not be externally visible
  - Scheduled event used to measure progress
- Milestone often produces project baselines:
  - formally reviewed work product
  - under change control (change requires formal procedures)
- Activities may be grouped into larger activities:
  - Establishes hierarchical structure for project (phase, step, ...)
  - Allows separation of concerns
  - Precedence relations often exist among activities.

Work package, Product, Deliverable, Baseline

- Work Package:
  - A description (specification) for the work to be accomplished in an activity or task
- Work Product:
  - Any tangible item that results from a project function, activity or task.
- Deliverable:
  - A work product to be delivered to the customer
- Baseline:
  - A work product that has been formally reviewed and agreed upon
  - A project baseline can only be changed through a change request and a formal change procedure.

Work package, Product, Deliverable



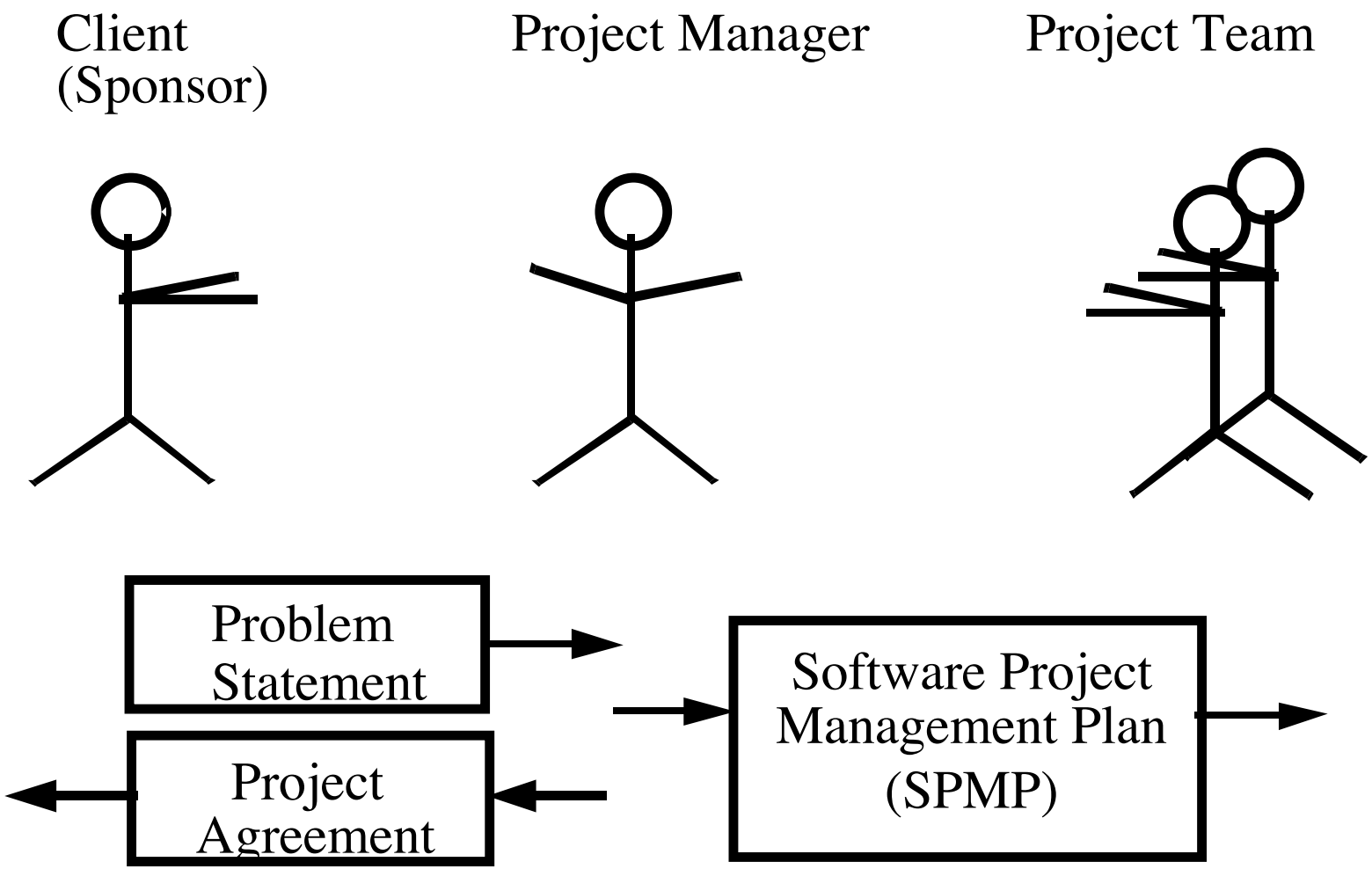
Project Agreement

- Project Agreement: Document written for a client that defines:
  - the scope, duration, cost and deliverables for the project
  - the exact items, quantities, delivery dates, delivery location.
- The form of a project agreement can be a contract, a statement of work, a business plan, or a project charter.

Software Project Management Plan (SPMP)

- IEEE Std 1058
- What it does:
  - Specifies the format and contents of software project management plans
  - Provides a standard set of abstractions for a project manager or a whole organization for developing software project management plans
    - Abstractions: Project, Function, Activities, Tasks
- What it does not do:
  - It does not specify the procedures or techniques to be used in the development of the plan.

Problem Statement, SPMP, Project Agreement



Software Project Management Plan

0. Front Matter
1. Introduction
2. Project Organization
3. Managerial Process
4. Technical Process
5. Work Elements, Schedule, Budget

Optional Inclusions

SPMP Part 0: Front Matter

- Title Page
- Revision sheet (update history)
- Preface: Scope and purpose
- Tables of contents, figures, tables

SPMP Part 1: Introduction

1.1 Project Overview

- Executive summary: description of project, product summary

1.2 Project Deliverables

- All items to be delivered, including delivery dates and location

1.3 Evolution of the SPMP

- Plans for anticipated and unanticipated change

1.4 Reference Materials

- Complete list of materials referenced in SPMP

1.5 Definitions and Acronyms

SPMP Part 2: Project Organization

2.1 Process Model

- Relationships among project elements

Example Slide 61

2.2 Organizational Structure

- Internal management, organization chart

Example Slide 52

2.3 Organizational Interfaces

- Relations with other entities (subcontractors, commercial software)

2.4 Project Responsibilities

- Description of major functions and activities; nature of each; who's in charge
- Matrix of project functions/activities vs. responsible individuals.

SPMP Part 3: Managerial Process

3.1 Management Objectives and Priorities

- Describes management philosophy, priorities among requirements, schedule and budget

3.2 Assumptions, Dependencies and Constraints

- External events the project depends on, constraints under which the project is to be conducted

3.3 Risk Management

- Identification and assessment of risk factors, mechanism for tracking risks, implementation of contingency plans

Examples
See Slide 53.

3.4 Monitoring and Controlling Mechanisms

- Frequency and mechanisms for reporting

3.5 Staffing Plan

- Number and types of personnel required for the project

SPMP Part 4: Technical Process

2.1 Methods, Tools and Techniques

- Specify the methods, tools and techniques to be used on the project

2.2 Software Documentation

- Describe the documentation plan

2.3 Project Support Functions

- Plans for (at least) the following project support functions.
  - Quality assurance
  - Configuration management (IEEE Std 1042)
  - Verification and validation
- The plans can be included in this section or there is a reference to a separate document.

SPMP Part 5: Description of Work Packages

5.1 Work Breakdown Structure (WBS)

Example
Slides 46-47

- Hierarchical decomposition of the project into activities and tasks

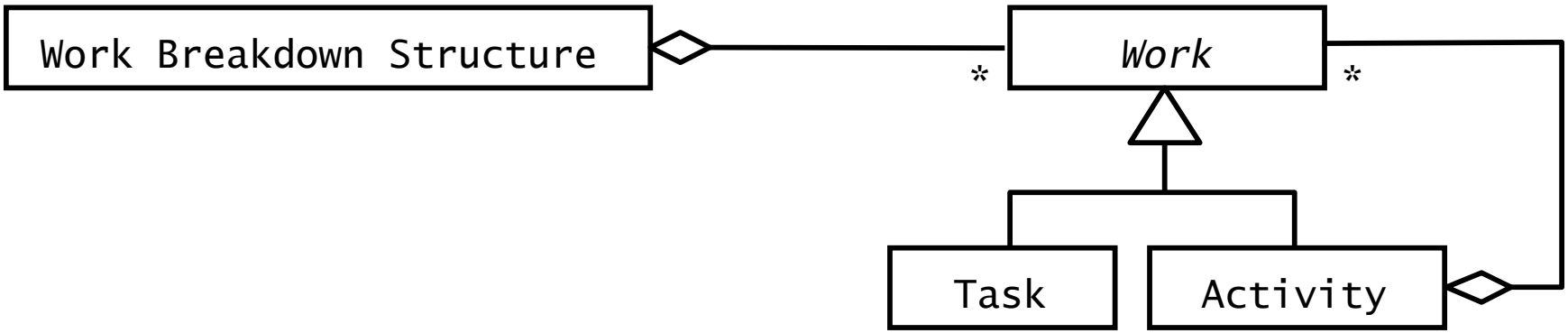
5.2 Dependencies between tasks

- Temporal relationship between tasks: "must be preceded by"
- Structural relationships

- A *dependency graph* visualizes the temporal dependencies
  - Nodes are activities
  - Lines represent temporal dependencies

Example
Slide 58.

Work Breakdown Structure



Work Breakdown Structure: The aggregation of all the work to be performed in a project.

Time estimates for establishing a WBS

- Establishing an WBS in terms of percentage of total effort:
  - Small project (7 person-month): **at least 7% or 0.5 Person Months (PM)**
  - Medium project (300 person-month): **at least 1% or 3 PMs**
  - Large project (7000 person-month): **at least 0.2 % or 15 PMs**

Source: Barry Boehm, Software Economics.

Creating Work Breakdown Structures

- Two major approaches
  - Activity-oriented decomposition („functional decomposition“)
    - Write the book, get it reviewed, do the suggested changes, get it published
  - Result-oriented decomposition („Object-oriented decomposition“)
    - Chapter 1, Chapter 2, Chapter 3
- Which one is better? Depends on project type:
  - Development of a prototype
  - Development of a product
  - Project team consist mostly of inexperienced beginners
  - Project team consists of experienced developers

Should you mix the WBS Approaches?

- Consider a WBS for the activity „Prepare report“
- Activity-oriented approach:
  - Write draft report (Joe)
  - Review draft report (Ann)
  - Write final report (Joe)
- Result-oriented approach:
  - Chapter 1 (Joe)
  - Chapter 2 (Ann)
- Mixed approach:
  - Chapter 1 (Joe)
  - Chapter 2 (Ann)
  - Review draft report (Ann)
  - Write final report (Joe)

Why is this bad?

“Write the final version of Chapter 2” can be included Ann’s task: “Chapter 2” or in Joe’s task “Write final report”.

SPMP 5: Description of Work Packages (cont'd)

5.3 Resource Requirements

- Estimates of the resources required to complete the project
  - Numbers and types of personnel
  - Computers, office and laboratory facilities, travel
- Maintenance and training requirements

5.4 Budget

5.5 Schedule

- Estimates the duration of each task
- Often used notation: PERT Chart (Performance Evaluation Review Technique, invented in 1958 for managing the development of the Polaris rocket) Example Slide 61
  - Dependency graph labeled with time estimates
  - Allows computation of critical paths.

How much planning should you do?

- Two styles of navigation [Gladwin 1964]
  - "European navigation"
    - Current Location and Desired Location
    - Planned Route
    - Route Deviation and Route Correction 
  - "Polynesian navigation"
    - Goal
    - Reaction to unexpected: Change the route 

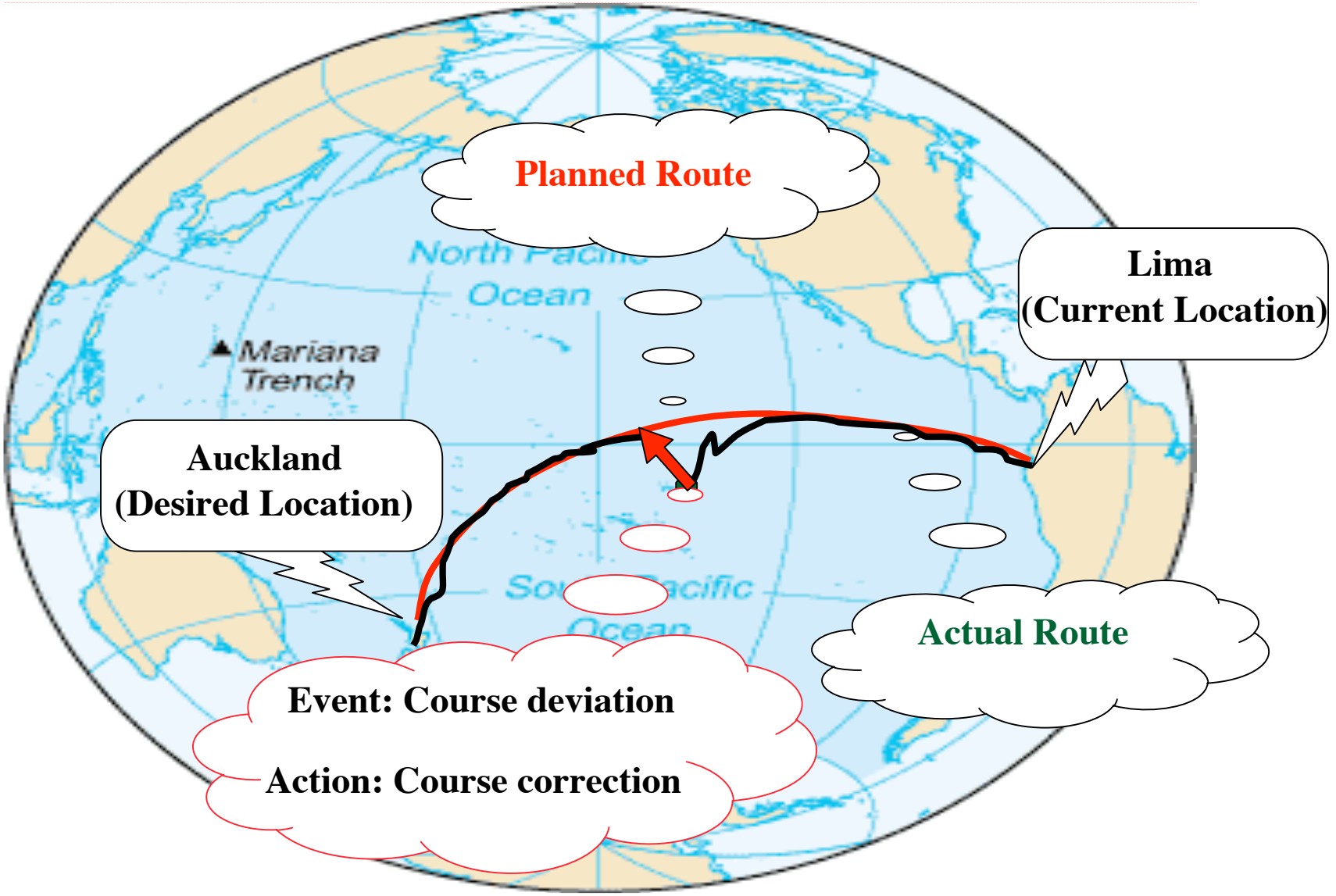
The main difference is the reaction to events
This leads us to the notion of situated action

Situated action [Suchman 1990]

- Situation Action: Selection of action depends on type of event, situation and skill of developer
  - Events: Course deviation, Birds seen, Clouds seen
- European Navigation is context independent:
  - Event: "Course deviation in the morning"
    => Action: "Course correction towards planned route"
  - Event: "Course deviation in the evening"
    => Action: "Course correction towards planned route"
- Polynesian Navigation is context dependent:
  - Event: "Birds seen", Context: It is morning
    => Action: "Sail opposite to the direction the birds are flying"
  - Event: "Birds seen", Context: It is evening
    => Action: "Sail in the direction the birds are flying.

46

"European Navigation"



Auckland Project Plan (European Navigation)

Project Goal: Auckland

Desired Outcome: Auckland is found

Team: Captain and 50 sailors

Organization: Flat hierarchy

Tools: Compass, speed meter, map

Methods: Determine course and write it before departure. Example: Start Lima. Sail West, keep the compass constantly at 97 degrees, stay at latitude 20 degrees

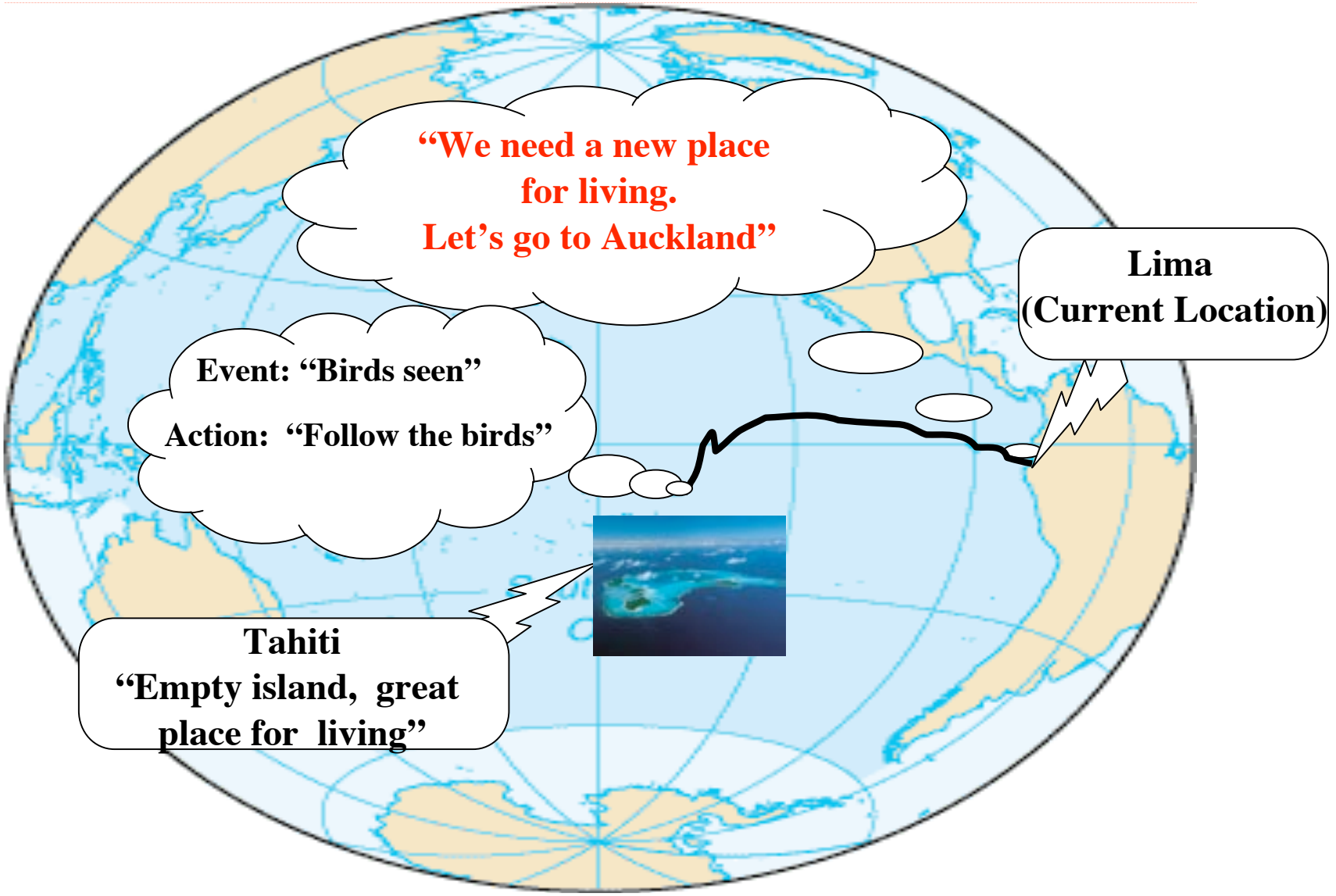
Work breakdown structure:

- Task T1 (Check direction): Determine current direction of ship
- Task T2 (Compute deviation): Determine deviation from plan
- Task T3 (Course Correction): Bring ship back on course

Process: Execute T1 and T2 hourly. If there is a deviation, execute T3 to bring the ship back on the planned course.

Schedule: With good wind 50 days; with doldrums 85 days.

Polynesian Navigation



Auckland Project Plan (Polynesian Navigation)

Project Goal: Auckland Desired Outcome: A new place

Team: Captain and 50 sailors Organization: Flat hierarchy

Tools: Stars for navigation, hand for measuring temperature

Methods: A set of event-action rules. When an event occurs, determine action in the current context.

Work breakdown structure:

- Task T1 (Set direction): Determine new course for ship
- Task T2 (Check Clouds): Look for clouds in the distance
- Task T3 (Check Birds): Look for birds, determine their direction
- Task T4 (Change course): Change direction to follow new course

Process:

- Start with T1 and T4. Then execute T2 and T3 regularly. Interpret task results (cloud detected, birds detected) in the current context. If the interpretation makes a new course more promising, execute tasks T1 and T4.

Schedule: None

Pros and Cons of Project Plans

- Advantages:
  - Very useful to kick off a software project (establish goals, organize teams, and start with development)
  - Also useful if the outcome is predictable or if no major change occurs
- Disadvantages:
  - Of limited value
    - when outcome is unpredictable
    - when events occur that change the project context
- Examples of unexpected events:
  - Appearance of new technology during the project
  - A visionary scenario turns out to be not implementable
  - Change of requirements

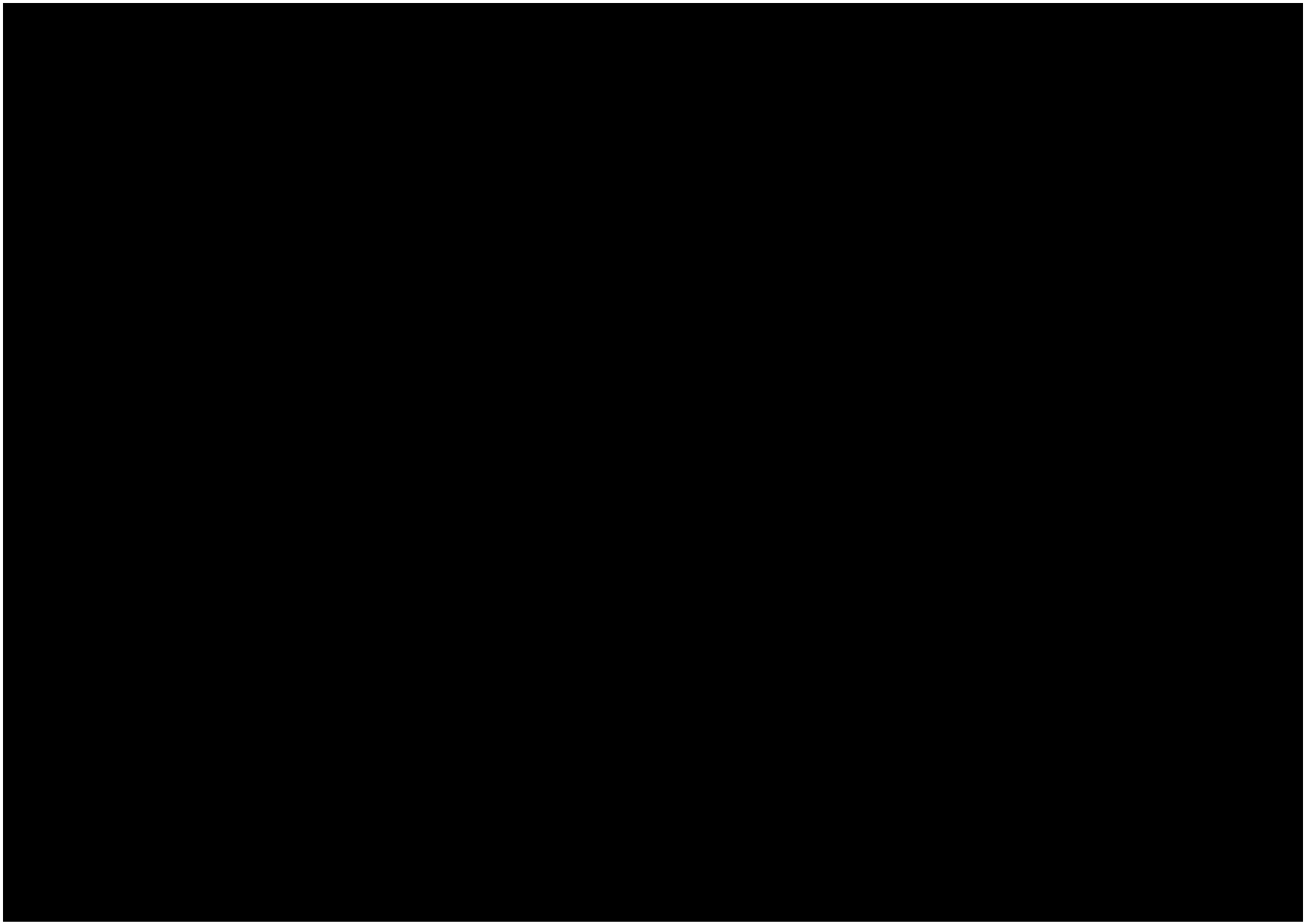
➢ Lecture on Agile Methods.

Additional Readings

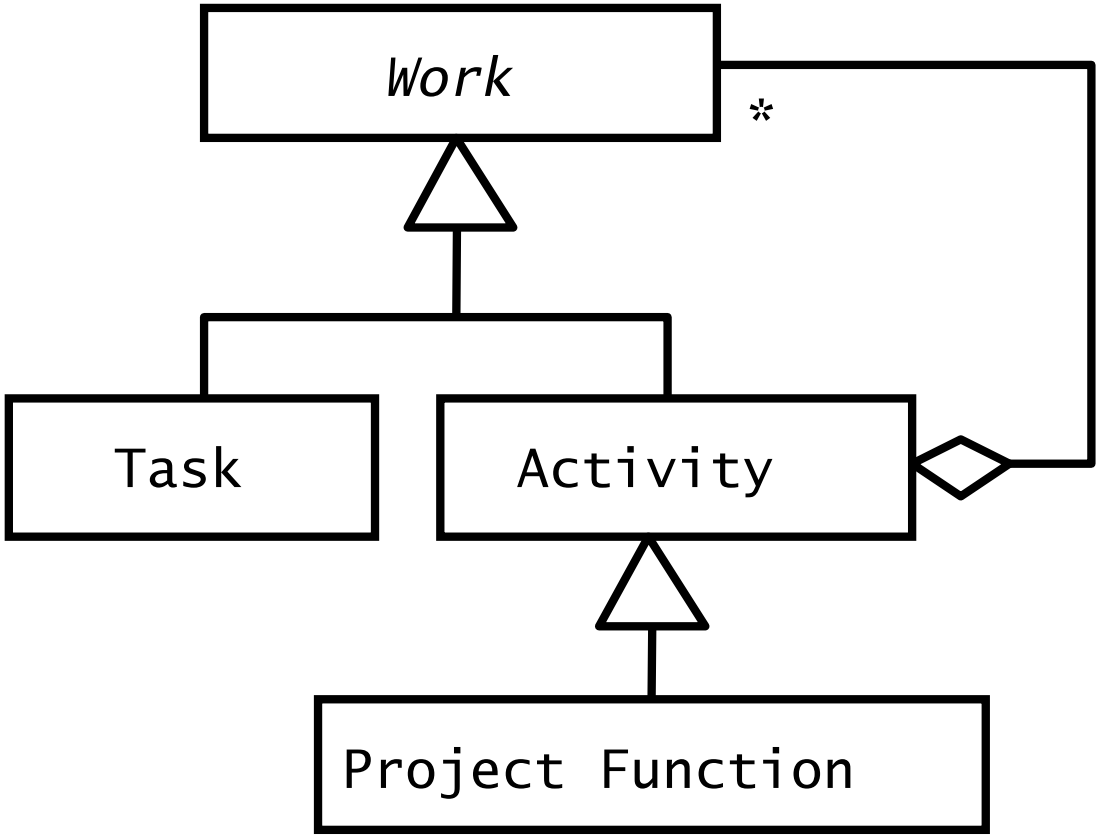
- [IEEE Std 1058]
  - Standard for Software Project Management Plans
- Frederick Brooks
  - No Silver Bullet, IEEE Computer, 20, 4 10-19, April 1987
- L.A. Suchman
  - Plans and Situated Actions: The Problem of Human Machine Communication, Cambridge University Press, 1990.
- Barry W. Boehm
  - *Software Engineering Economics*, Prentice Hall, 1981.
- T. Gladwin
  - "Culture and logical process", in W. Goodenough (ed), Explorations in Cultural Anthropology: Essays Presented to George Peter Murdock, McGraw-Hill, New York, 1964.

Summary

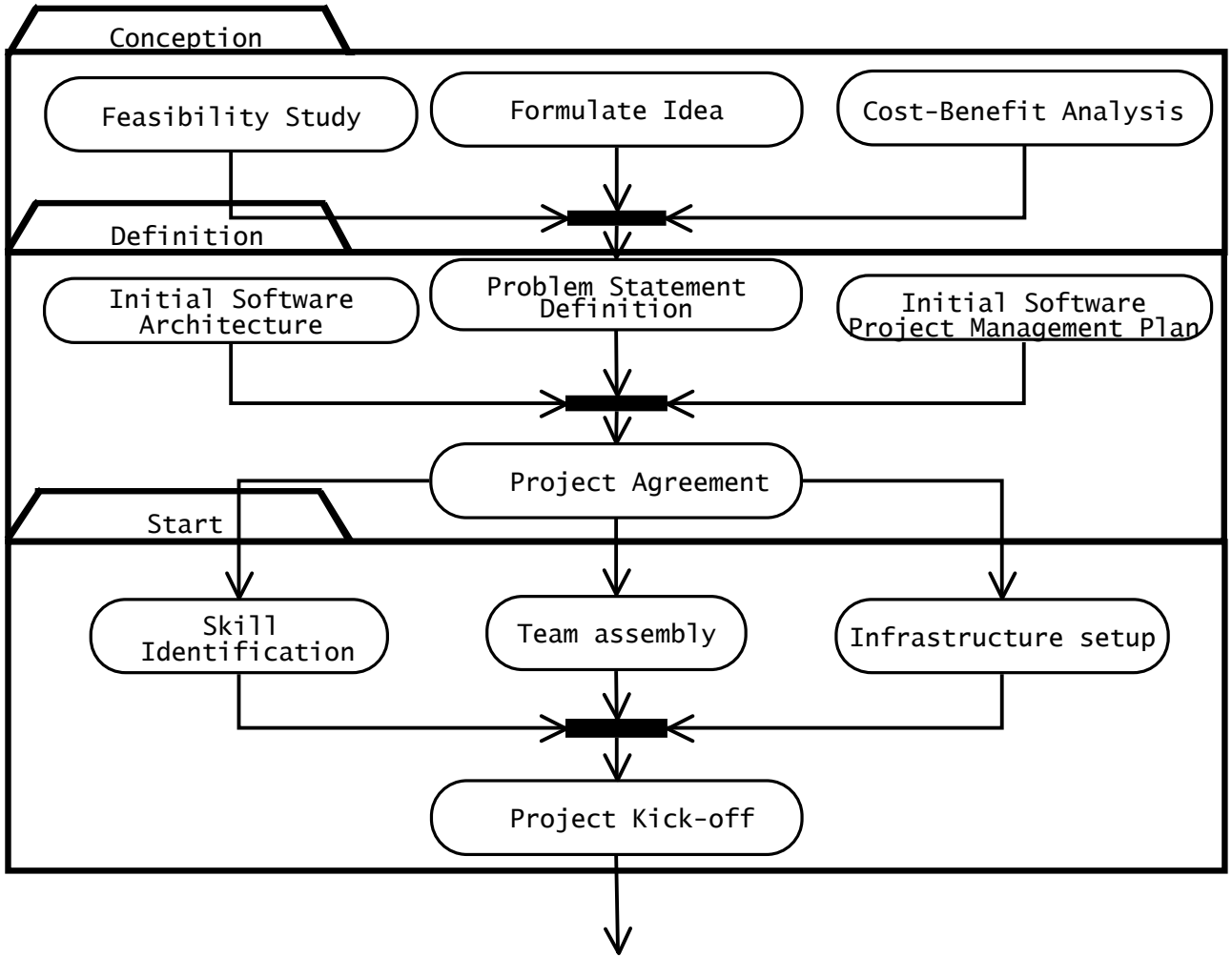
- Software engineering is a problem solving activity
  - Developing quality software for a complex problem within a limited time while things are changing
- The system models addresses the technical aspects:
  - Object model, functional model, dynamic model
- Other models address the management aspects
  - SPMP, WBS, Schedule are examples
  - Other models: Issue models, Cost models
- Technical terms introduced in this lecture:
  - Project, Activity, Function, Task, WBS, SPMP.



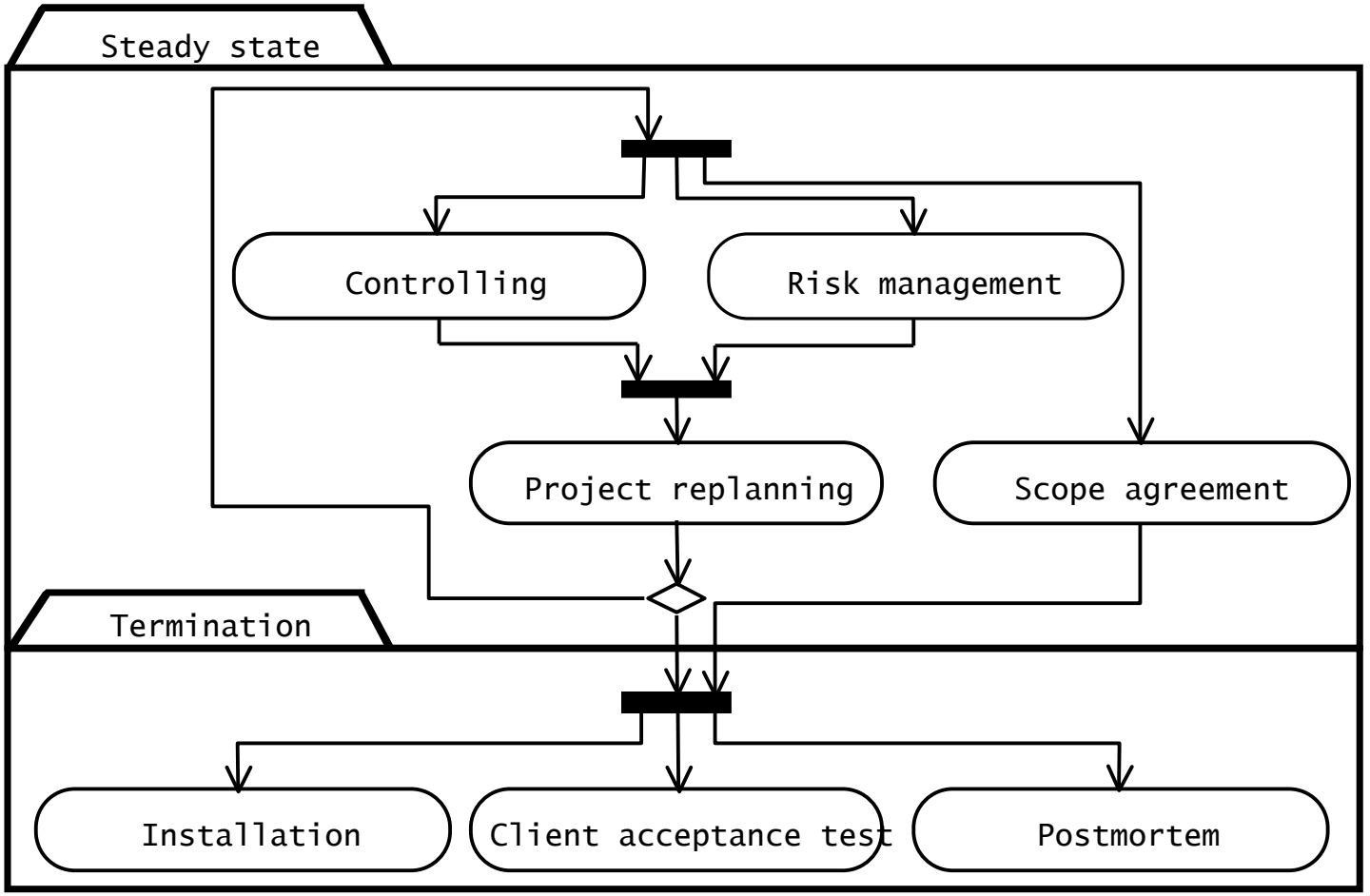
Tasks, Activities and Project Functions (UML Class Diagram)



Project Management Activities in a Software Project



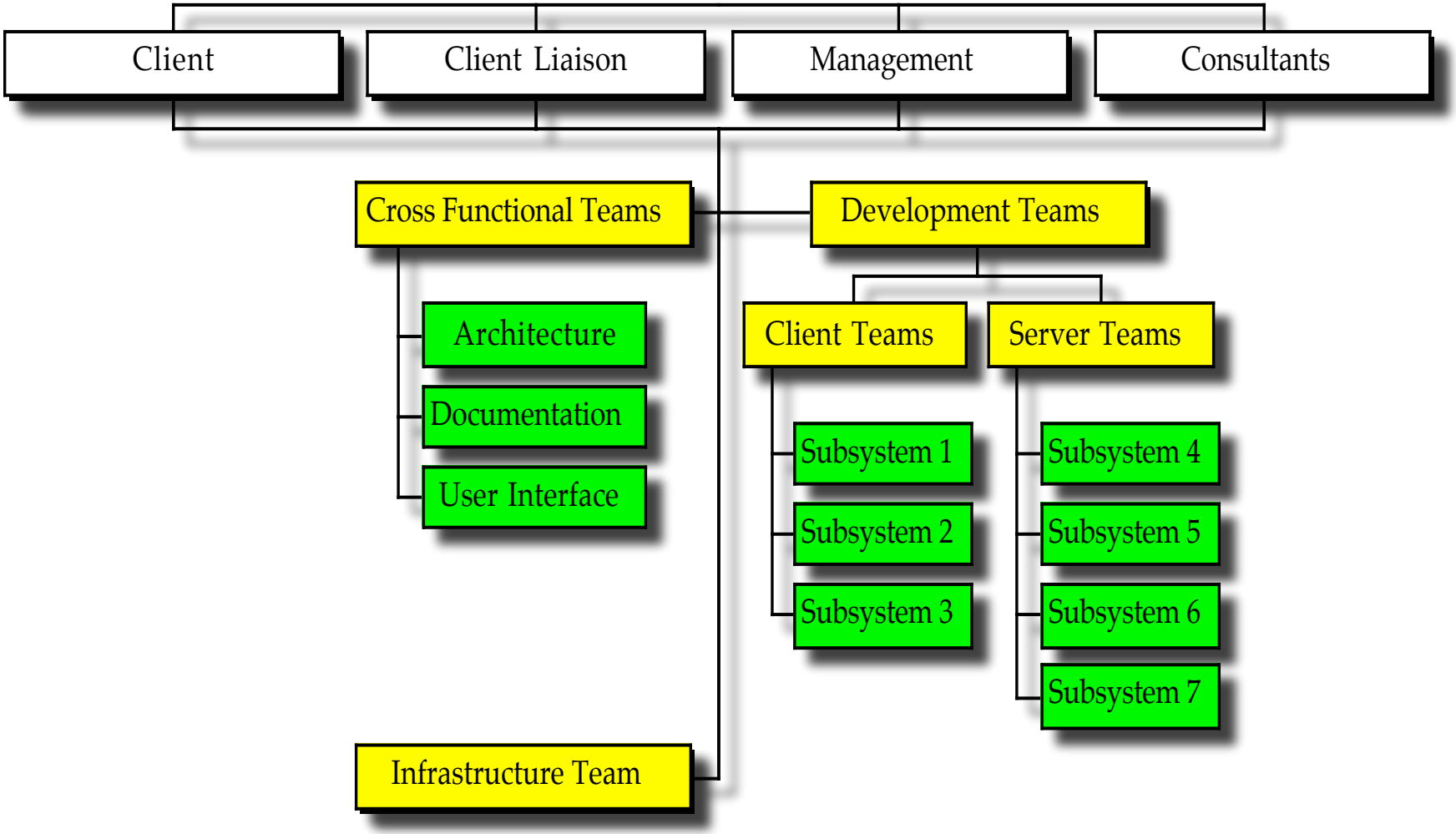
Management activities in a software project (cont'd)



"Laws" of Project Management

- Projects progress quickly until they are 90% complete. Then they remain at 90% complete forever.
- When things are going well, something will go wrong.
- When things just can't get worse, they will.
- When things appear to be going better, you have overlooked something.
- If project content is allowed to change freely, the rate of change will exceed the rate of progress.
- Project teams detest progress reporting because it manifests their lack of progress.

Organizational Structure Example



Examples of Risk Factors

- Contractual risks
  - What do you do if the customer becomes bankrupt?
- Size of the project
  - What do you do if you feel the project is too large?
- Complexity of the project
  - What do you do if the requirements are multiplying during analysis? („requirements creep")
- Personal fluctuation
  - How do you hire people? Is there a danger of people leaving the project?
- Customer acceptance
  - What do you do, if the customer does not like the developed prototype?

Example: Activities to Build a House

- Surveying
- Excavation
- Request Permits
- Buy Material
- Lay foundation
- Build Outside Wall
- Install Exterior Plumbing
- Install Exterior Electrical
- Install Interior Plumbing
- Install Interior Electrical
- Install Wallboard
- Paint Interior
- Install Interior Doors
- Install Floor
- Install Roof
- Install Exterior Doors
- Paint Exterior
- Install Exterior Siding
- Buy Pizza

These activities are known by a good contractor

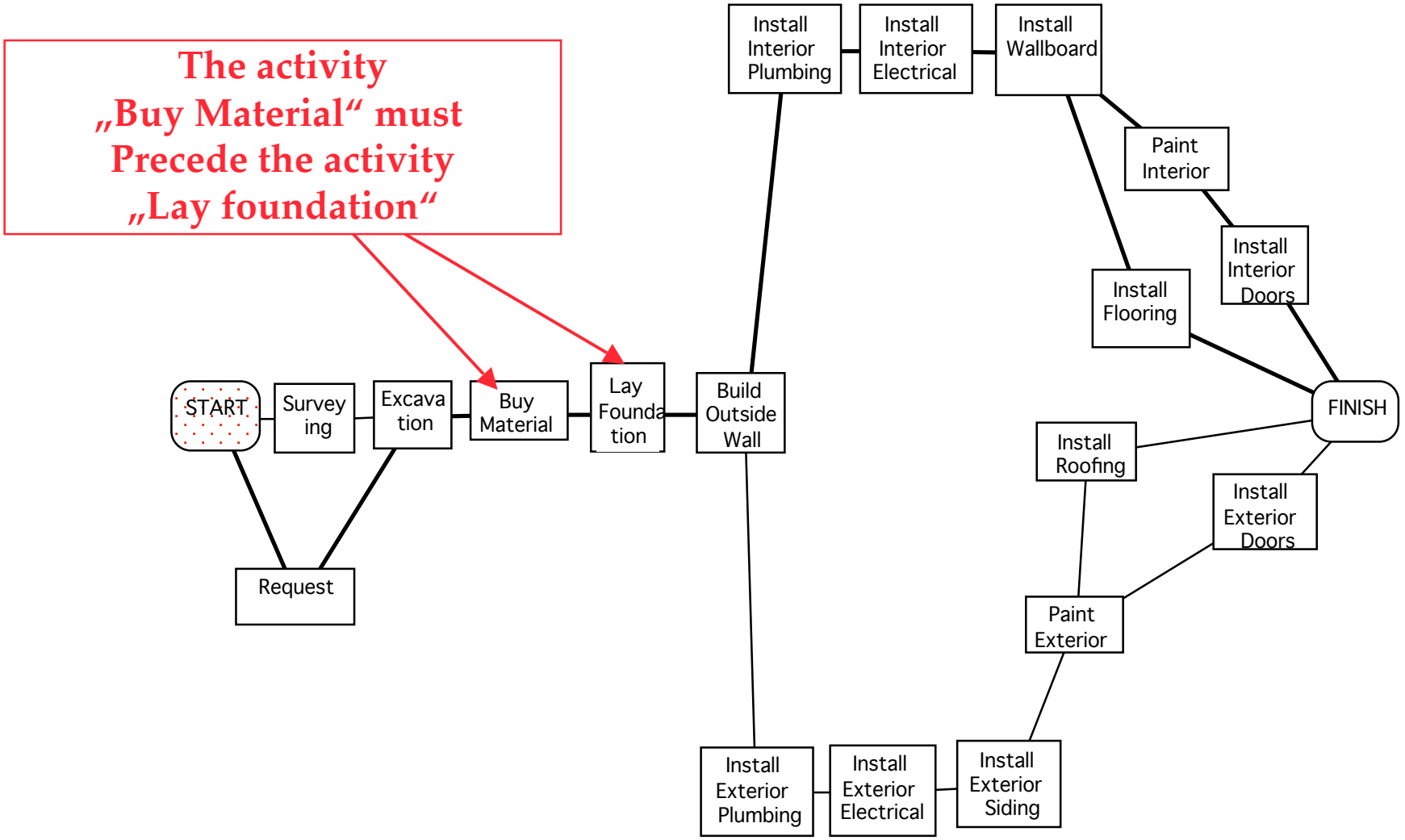
Finding these activities may also require brainstorming
It then requires similar activities used during analysis (use case modeling).

Hierarchical Organization of the Activities (Top-Level Use Cases)

- Building the house consists of
  - Prepare the building site
  - Building the Exterior
  - Building the Interior

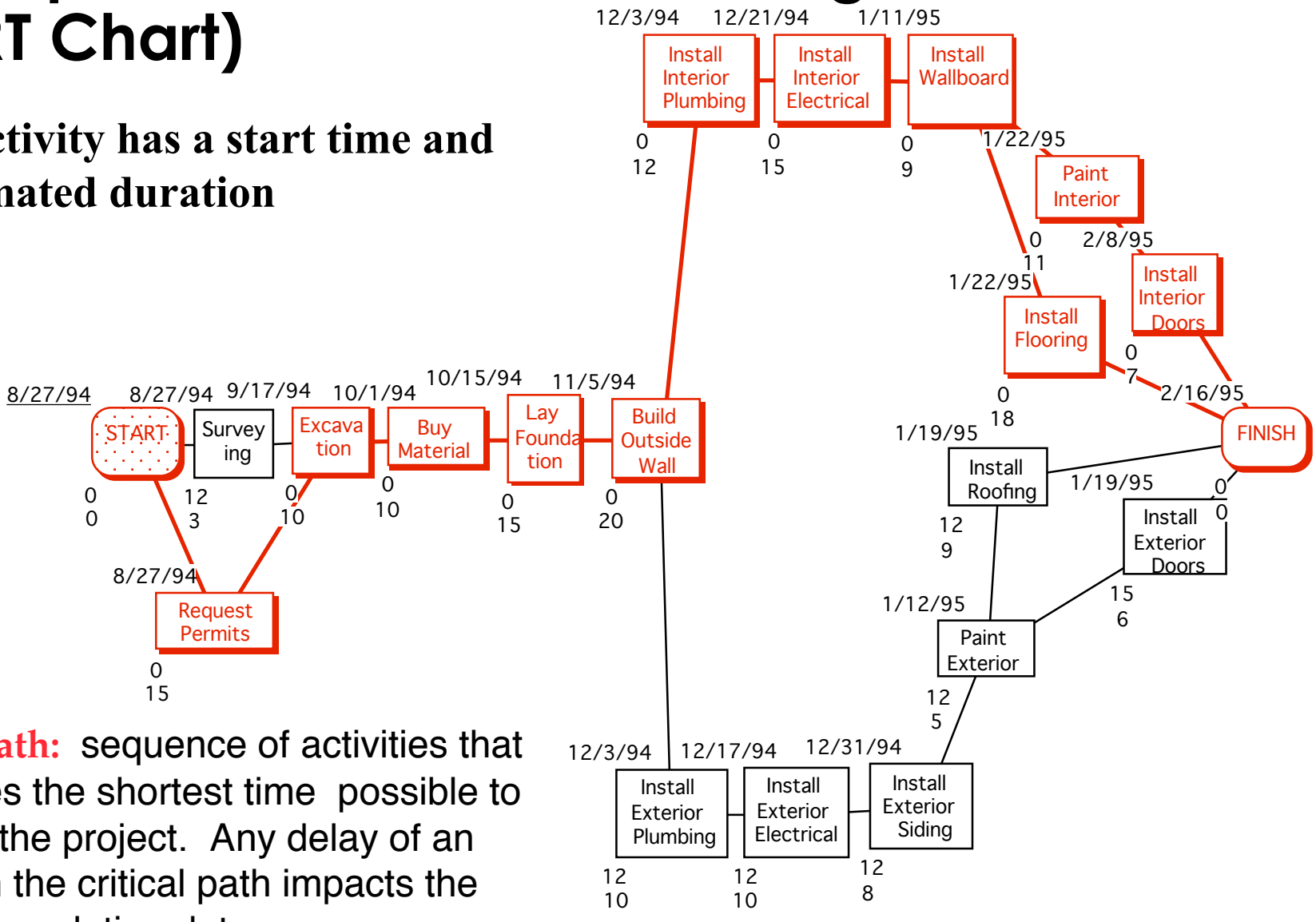
- Preparing the building site consists of
  - Surveying
  - Excavation
  - Buying of material
  - Laying of the foundation
  - Requesting permits

Example of a Dependency Graph: Building a House



Example of a Schedule: Building a House (PERT Chart)

Each activity has a start time and an estimated duration



Critical path: sequence of activities that determines the shortest time possible to complete the project. Any delay of an activity on the critical path impacts the planned completion date.

Goals of PERT Charts

- Determination of total project time („project duration“)
- Determination of the critical path
- Determination of slack times

Software Lifecycle Activities

Requirements Elicitation	Analysis	System Design	Object Design	Implementation	Testing

Structural Relationships: