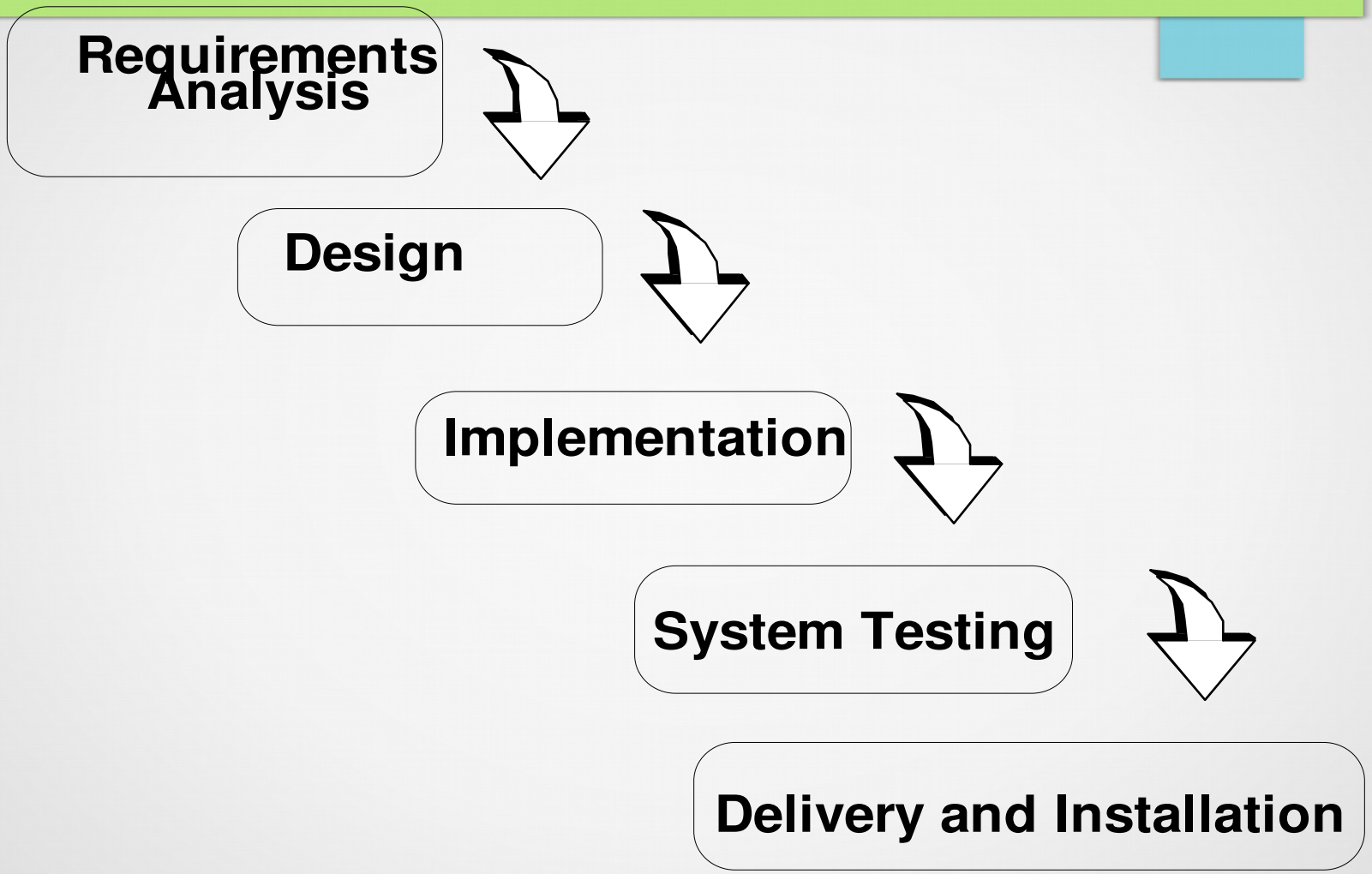
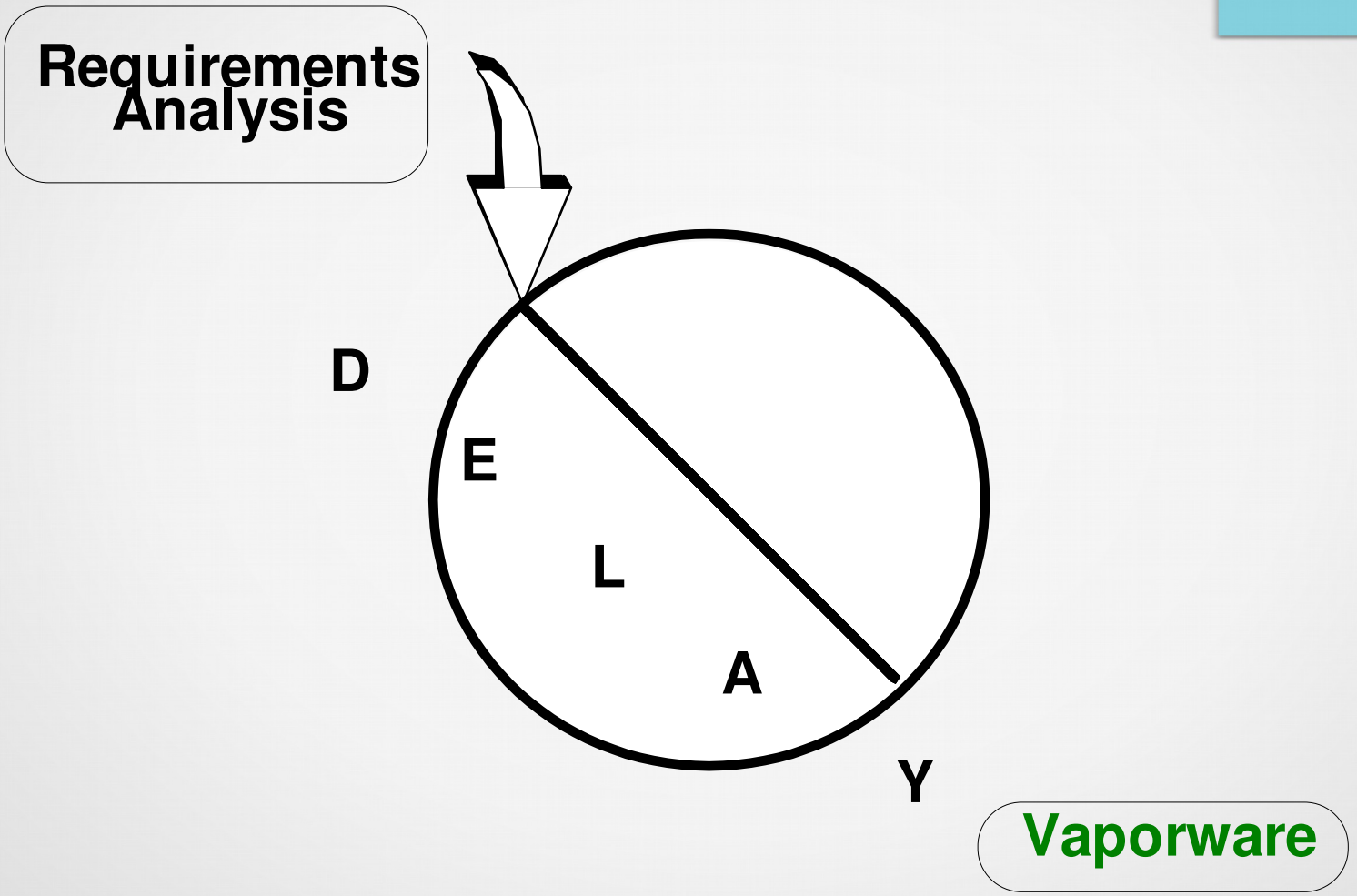
Laws of Project Management

- 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.

How it should go



How it often goes



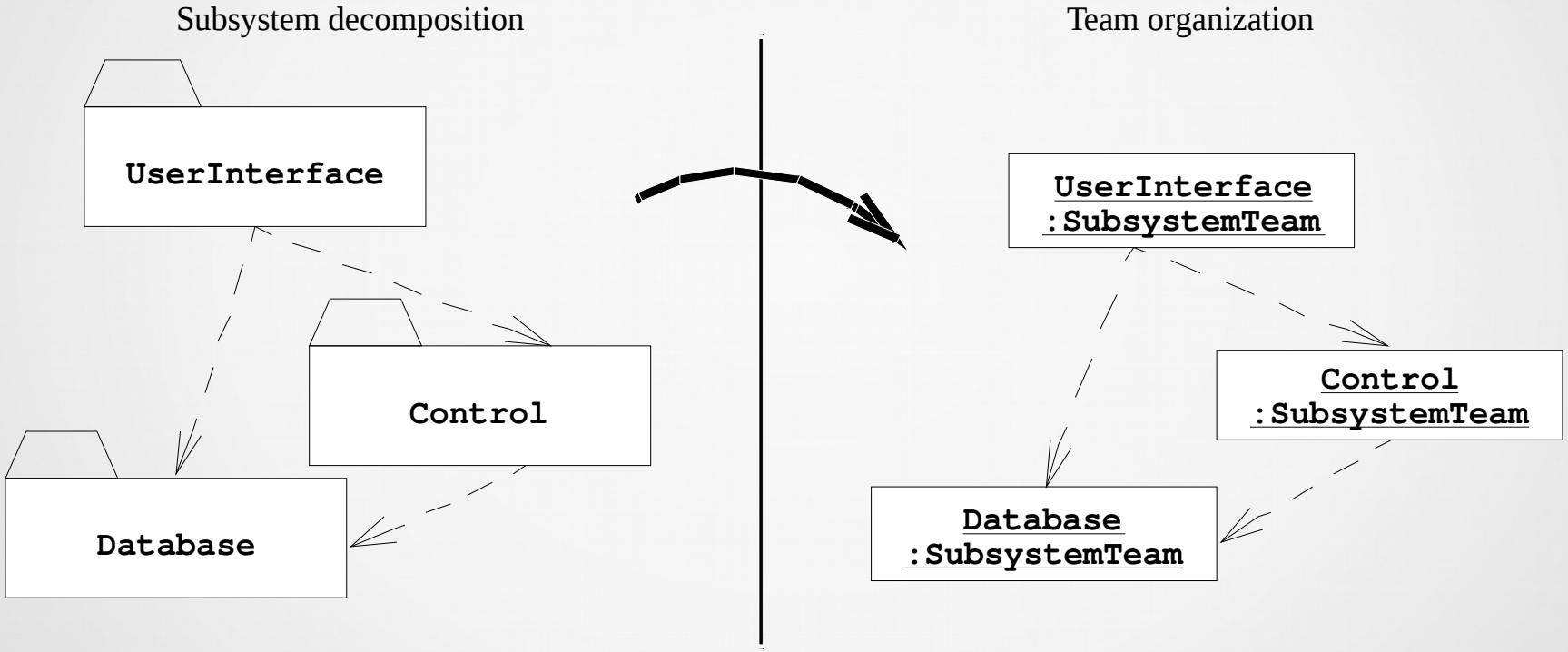
Outline

- Organizational Structures
  - Functional, Project and Matrix Organizations
- Key project roles in organizational structures
  - Project Manager, Team members, upper management, ...
- Relationships between roles
- Information flows between roles
  - Decision making, status reporting, communication
- Identifying people
  - Audience List, Drivers, Supporters, Observers
  - Involvement of audience members during the lifetime of a project
- Properties of roles:
  - Responsibilities, Authority and Delegation
- And if time permits:
  - Micromanagement (and how to avoid it)

Organizational Structures

- Types of Organization
  - Functional organization
  - Project-based organization
  - Matrix organization
- Parameters for each organization type
  - Organizational Unit
  - Key players
  - Roles and Responsibilities
  - Structure: Information flow between roles
  - Benefits and Challenges (“pros and cons”)
- Heuristics
- Let’s start with an example and a few definitions….

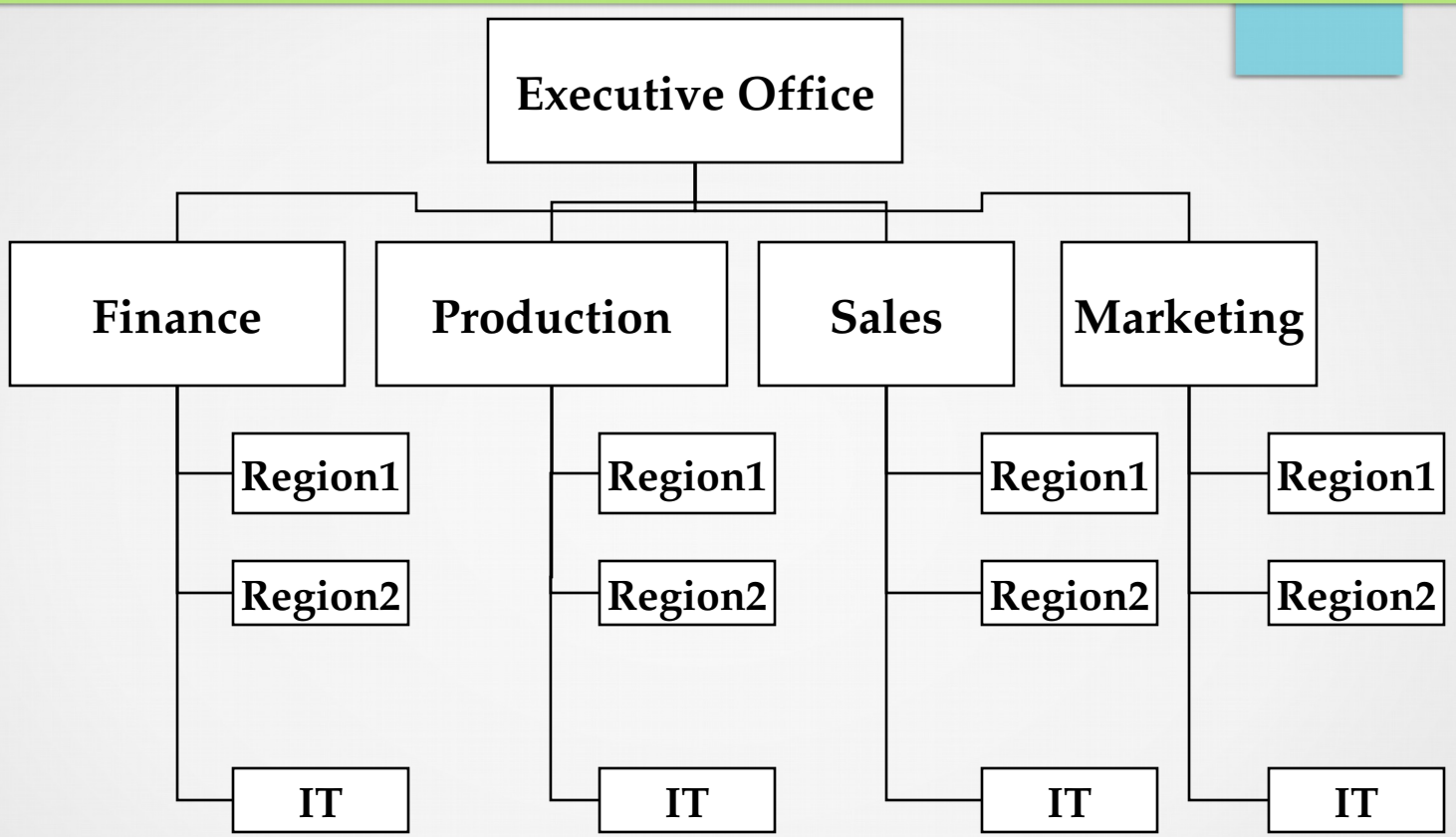
Toy Project with 3 Teams



Functional Organization

- **Definition**: In a **functional organization** participants are grouped into so-called **departments**, each of which addresses a function.
- Examples of departments:
  - Traditional businesses: Research, development, production, sales, finance.
  - In software companies the departments correspond to the activities in the software process: Analysis, design, integration, testing departments.
- Key properties:
  - Projects are usually pipelined through the departments of a functional organization. The project starts in research, then it moves to development, then it moves to production, ….
  - Only a few participants are involved in the complete project.
  - Separate departments often address the same cross-functional needs (Examples: configuration management, IT infrastructure)

Example of a Functional Organization



Line organization of a „traditional business“

Properties of Functional Organizations

- Advantages:
  - Members of a department have a good understanding of the functional area they support.
  - Departments don‘t compete with another to get the support of their support teams
- Disadvantages:
  - Because each department has its own support team, different work procedures and reporting systems are the rule.
  - It is difficult to make major investments in equipment and facilities.
    - Example: Two departments with a budget of 50,000 Euro each need a printer that costs 100,000 Euro.
    - Both need only 50% of the maximum capacity.
    - Neither department can buy it, because they don‘t have sufficient funds.
  - High chance for overlap or duplication of work among departments
  - Conflicts between departments with different objectives

Project Organization

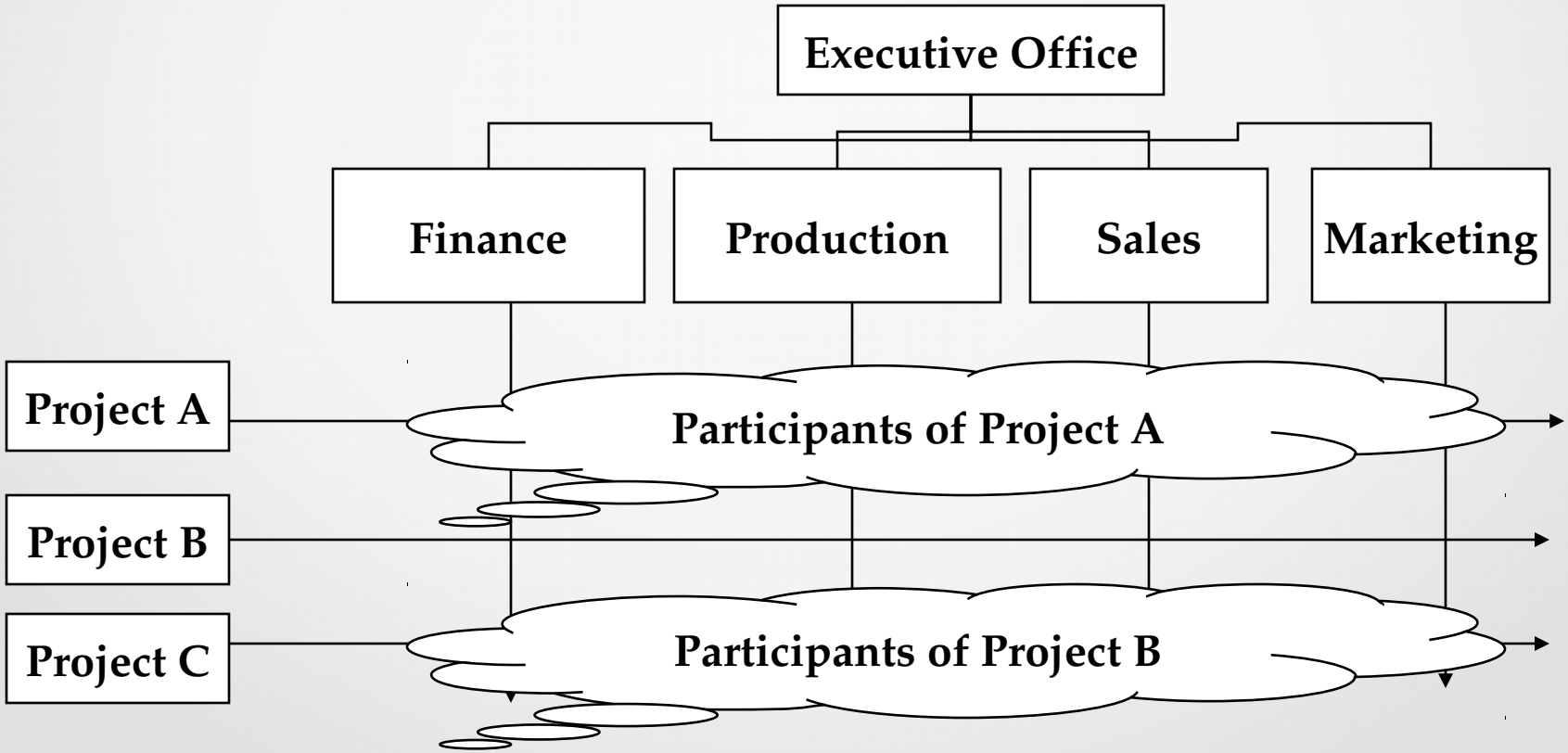
- In a **project organization** participants are grouped into **projects**, each of which has a problem to be solved within time and budget.
- **Key properties:**
  - Teams are assembled for a project as it is created. Each project has a project leader.
  - All participants are involved in the complete project.
  - Teams are disassembled when the project terminates

Properties of Project Organizations

- Advantages
  - Very responsive to new project requests (because the project is newly established and can be tailored around the problem)
  - New people can be hired/selected who are very familiar with the problem or who have special capabilities.
  - There is no waste of staff workload
- Disadvantages:
  - Teams cannot be assembled rapidly. Often it is difficult to manage the staffing/hiring process.
  - Because there are „no predefined lines“, roles and responsibilities need to be defined at the beginning of the project
  - What people will do after project completion?

Matrix Organization

- In a matrix organization, participants from different departments of the functional organization are assigned to work on projects as they are created.
- The project manager and team members may be assigned to the project for less than 100 % of their time



Properties of Matrix Organizations

- Advantages:
  - Teams for projects can be assembled rapidly
  - Scarce expertise can be applied to different projects as needed
  - Consistent work and reporting procedures can be used for projects of the same type.
- Disadvantages:
  - Team members usually are not familiar with each
  - Team member have different working styles
  - Team members must get used to each other
  - Conflicts between functional and project manager

New Challenges in Matrix Organizations

- Team members must respond to two different bosses with different focus:
  - Focus of the functional manager: Assignments to different projects, performance appraisal
  - Focus of the project manager: Work assignments, project team support
- Team members working on multiple projects have competing demands for their time
  - Team members working on more than one project have even more project members to report to
  - Some people who have claim on the team member‘s time may be at similar levels in the organization‘s hierarchy
- Multiple work procedures and reporting systems are used by different team members
  - Development of common procedures needs to be addressed at project kickoff time

When to use a Functional Organization

- Projects with high degree of certainty, stability, uniformity and repetition.
  - Requires little communication
  - Role definitions are clear
- When?
  - The more people on the project, the more need for a formal structure
  - Customer might insist that the test team be independent from the design team
  - Project manager insists on a previously successful structure

When to Use a Project or Matrix Organization

- Project with degree of uncertainty
  - Open communication needed among members
  - Roles are defined on project basis
- When?
  - Requirements change during development
  - New technology develops during project

Definition: Role

- A **role** is a set of responsibilities
- A role is instantiated during a project and assigned to one or more persons.
- Instances of roles are often also called **players** or **stakeholders**

Key Roles in Organizations

- **Project Manager:** The person ultimately responsible for the successful completion of the project
- **Project Team Member:** Participants who are responsible for performing individual activities and tasks (in a project or matrix organization)
- **Functional Manager:** The team member‘s supervisor in the department (in a functional organization)
- **Upper management:** People in charge of the departments or projects

In the following we focus only on roles in project and matrix organizations

Responsibilities of the Project Manager

- Determine objectives, schedule and resource budgets
- Design a software project management plan (SPMP)
- Create and sustain focused and motivated teams
- Determine the team‘s work procedures, reporting systems and communication infrastructure.
- Accomplish project objective within time and budget
- Monitor performance against the plan
- Resolve technical conflicts and interpersonal conflicts
- Control changes in the project
- Report on project activities to upper management
- Keep the client informed and committed
- Contribute to the team members performance approval

General Responsibilities of Team Members

- Technical responsibilities:
  - Perform assigned tasks within time and budget
  - Acquire technical skills and knowledge needed to perform the work
- People responsibilities
  - Identify situations and problems that might affect your team members‘s tasks
  - Keep your team members informed of your progress and problems you encounter

Other Team Member Roles

- Project Management
  - Coach
  - Team leader
  - API Liaison
  - Planner
- Meeting Management
  - Minute Taker
  - Scribe
  - Primary facilitator
- Development
  - Analyst
  - Designer (Software Architect)
  - Programmer
  - Tester
  - Maintainer
  - Trainer
  - Document Editor
  - Web Master
  - Configuration Manager

Responsibilities of the Coach

- Listen to gripes from individual team members
- Attend weekly project meetings
- Review weekly team status reports
- Schedule and prepare meetings with project manager
- Insist that project guidelines are followed
- Assign presentations to team members (in-class project meetings, client review, client acceptance test)
- Resolve team member conflicts if they cannot be resolved otherwise

Responsibilities of the Team Leader

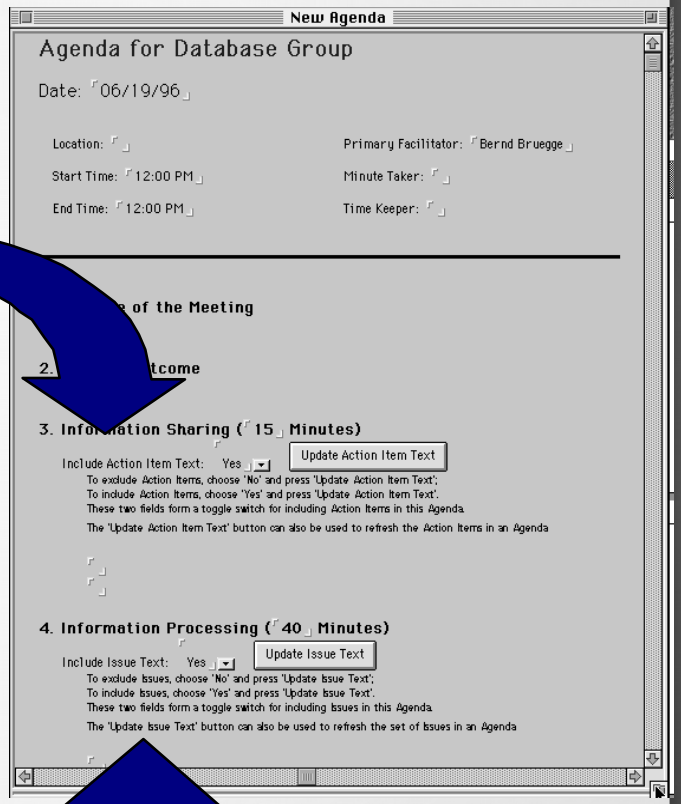
- Responsible for intra-team communication
  - Run the weekly project meeting
  - Post the agenda before the meeting
  - Define and keep track of action items assigned to team members (who, what, when)
  - Measure progress (Enforce milestones)
  - Deliver work packages for the tasks to the project manager
  - Present team status to project manager
- *Heuristics: The team leader should to be rotated among members of the team.*

Team Leader: Create an Agenda

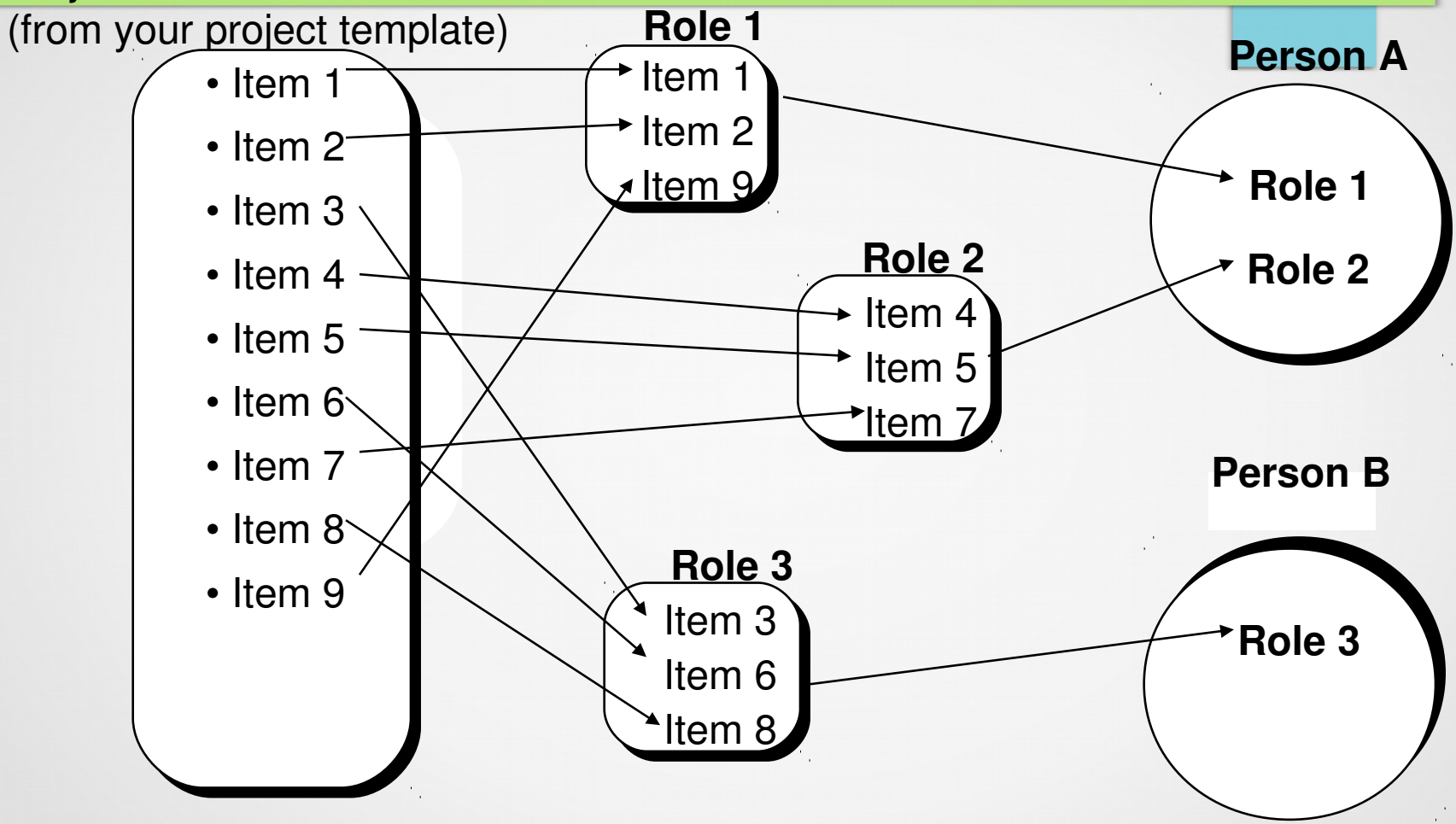
- Purpose of Meeting
- Desired Outcome
- Information Sharing
- Information Processing
- Meeting Critique

Action Items (Check Previous Meeting)

Issues (Check Previous Meeting & BBoards)



Assigning Responsibilities To People



Promoter Roles

- Promoter are self appointed individuals who identify themselves with the outcome of the project.
  - They are member of the corporate organization and may not necessarily be directly involved with the project.
  - Instead, they are the interface to the rest of the corporate organization.
- Because of their power, knowledge of technology, or familiarity with the project's processes, they are able to promote and push specific changes through an existing organization which are needed to make the project a success.

Power Promoter

- Also called executive champion or project champion
- Pushes the change through the existing organizational hierarchy.
  - not necessarily at the top of the organization, but must have protection from top level management, otherwise project opponents might be able to prevent the success of the project.
- **Tasks:**
  - Constantly identify difficulties, resolve issues, and communicate with the project members, especially with the developers.
- **Example at project level:** Project Leader.
- **Example at corporate level:** Chief Executive Officer (CEO).

Knowledge Promoter

- Also called the technologist
- Promotes change arising in the application domain or the solution domain. Usually closely associated with the power promoter.
- **Tasks:** Acquire information iteratively, understand the benefits and limitations of new technologies, and argue its adoption with the other developers.
- **Example at project level:** System architect.
  - Reports to project manager
  - Does not have any direct subordinate in the reporting hierarchy
  - Has final say over all technical decisions in the system.
- **Example at corporate level:** Chief Technical Officer (CTO).

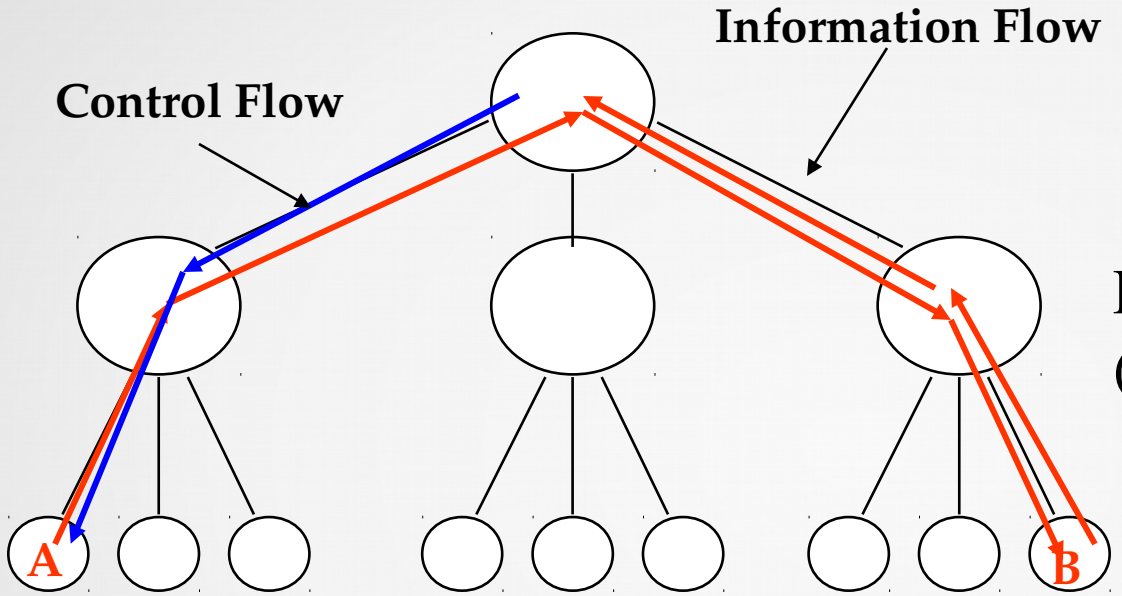
Process Promoter

- The process promoter has intimate knowledge of the projects processes and procedures.
- The process promoter is in constant interaction with the power promoter to get consensus on the overall goals.
- **Tasks:** Bridge between the power and knowledge promoters, who often do not speak or understand the same language.
- **Example at project level:** Development lead. Responsible for the administrative aspects of a project, including planning, milestones definition, budgeting and communication infrastructure.
- **Example at corporate level:** Chief Information Officer (CIO

Hierarchical Organization

- Often also called ***centralized organization***. Examples: Military, church, traditional businesses.
- **Key property:** The organization has a tree structure. Decisions are made at the root and communicated to the leaf nodes. The decision association is also used for reporting and communication.
- Advantages:
  - Centralized control over project selection
  - One set of management and reporting procedures for all project participants across all projects
  - Established working relationships among people
  - Clearly established lines of authority to set priorities and resolved conflicts
  - Authority to pressure people to honor their action items
  - Clearly defined career path

Hierarchical Project Organization



Chief Executive

First Level Manager
("Front-Line Manager")

Project Members

A wants to talk to B: Complicated Information Flow interdipendenza

B wants to make sure A does a certain change: Complicated Controlflow

Basis of organization:
Complicated information and control flow
across hierarchical boundaries

Heuristics for Project Managers

1. Create team identity
   - Clarify team vision and working relationships
   - Define team procedures (meeting management, configuration management, system integration strategy)
   - Clarify each participant‘s authority
   - Make sure your team is functioning
   - Be sure only one person is assigned as project manager
2. Create team membery buy-in
   - Get commitment to the project goals (tough in matrix environment)
   - Get to know other people‘s style
3. Get support from the environment
   - Get a project champion (for example a power promoter)
4. Develop general procedures for
   - Conflict resolution
   - Communication between teams and project leaders, communication with upper management and for communication with the client

Guidelines for Establishing the Audience List

- Develop your audience list from a template that worked well in a previous project
- Eventually instantiate instances from each category with position and name
- When in doubt, **ADD** a person‘s name
- Separately include a person‘s name for every different role played by him or her
- Speak with a wide range of people
- Allow sufficient time to developing your audience list (mainly during project initiation time)
- Continue to maintain the audience list during the project (remove names, add names)
- Encourage project participants to identify new candidates

Another Categorization of the Audience List

- Drivers:
  - People who have some say in defining the results of the project
- Supporters:
  - People who help to perform the activities and tasks of the project. Supporters include those who authorize resources for the project as well as those who work on it.
- Observers:
  - People who are interested in the activities and results of the project. Observers have no say in the project and they are not actively involved. However, the project may affect them at some point in the future.
- Project Champion (Power Promoter):
  - A Person who strongly supports the project, advocates it in disputes, takes whatever is necessary to help ensure the succesful completion of the project.

Key Concepts for Mapping Roles to People

- Authority:
  - The ability to make binding decisions between people and roles
- Responsibility:
  - The commitment to achieve specific results.
- Accountability: (rispondere di, rendere conto)
  - Tracking a task performance to a participant.
- Delegation:
  - Binding a responsibility assigned to one person (including yourself) to another person.

Authority vs Responsibility vs Accountability

- Authority vs Responsibility
  - They are similar: Both are upfront agreements. Before you start a project, you agree on who can make decisions and who will ensure that particular results are achieved.
  - They are different: Authority focuses on process such as activities and tasks, responsibility focuses on outcome such as work products and deliverables
    - Good leaders delegate authority; they never delegate responsibility
- Responsibility vs Accountability:
  - Similarity: Both focus on results
  - Difference: Responsibility is a before-the-fact agreement, accountability is an after-the-fact process.
  - If you are responsible you should be held accountable.
  - If you are not responsible you should not be held accountable.
  - Scapegoating: Making somebody accountable who was not responsible

Delegation

- **Delegation:** Rebinding a responsibility assigned to one participant (including yourself) to another project participant.
- Three reasons for delegation:
  - Time Management: To free yourself up to do other tasks
  - Expertise: To have the most qualified person make decisions
  - Training: To develop another person’s ability to handle additional assignments.
- You can delegate authority, but not responsibility
- You can share responsibility
  - Shared relationship between activities and roles can be described in a linear responsibility chart

- For successful Lean leadership we need to separate responsibility and authority. This seems strange because we normally think that authority and responsibility are linked together. Could this be another Lean thinking paradox!
- The focus in a Lean organisation has shifted from “who has the authority” to “what is the right thing to do”. This is achieved by getting each person to take initiative to actually solve problems that improve his or her job, by placing individual responsibility at the lowest possible level where the work is actually done. and ensuring that every person’s job is aligned with providing value for the customer that ultimately leads to prosperity for the company.
- Our job as a Lean leader is to help expose problems and then make sure people have the skills and the tools to solve these problems. It is more a philosophy of “let’s figure this out together” and creating an environment where learning from mistakes is an accepted part of our continuous improvement process.
- To help expose problems we must spend more time in the process asking why, and then focus on giving people the responsibility and ownership for developing and implementing the solution. Lean leaders avoid relying on authority, instead leading by influence and example, as if they have no authority.

Linear Responsibility Chart

- A **linear responsibility chart** is a matrix that depicts the role that each project participant will play in different activities identified in the work breakdown structure.
- **Rows: Project activities**
- **Columns: Roles/Project participants**
- **Entries: Type of responsibility**
  - *P (Primary responsibility):* You have committed to ensure that the desired result is achieved
  - *S (Secondary responsibility):* You have committed to some portion of the result
  - *A (Approval):* You are not doing the work, but you will approve what has been done
  - *R (Review):* You will review and comment on the work product of an activity
  - *O (Output):* You will receive the work product of an activity
  - *I (Input):* You will provide input for a task or activity

Example of a Responsibility Chart

	Project Manager	Team Leader	Team Member A	Team Member B
Develop SPMP	**P**			
Run weekly meeting		**A**	**P**	**S**
Write SDD	**P**	**S**	**S**	**S**

Legend:
P = Primary responsibility
S = Secondary responsibility)
A = Approval

Analysing Responsibility Charts identifies Risks

- Problem: Somebody is heavily committed.
  - Possible Project Management Issues: Not enough time to handle all duties, making too many key decisions, What if this person leaves during the project
- Problem: The project manager has no direct responsibilities
  - Issues: Will the project manager fully understand status reports?
- Problem: An activity requires many approvals
  - Issue: Does anyone else have to approve the activity. Are there too many people involved approvals? Is your estimated duration of the activity too optimistic, because the approval is out of your hands?
- After you identify an issue, you should address it in your risk management plan.

Micro Management

- Micromanagement is the excessive involvement of a manager in the details of a task assigned to a team member.
- Micromanagement is inefficient use of the time and energy of all project participants.
- It leads to tension and low morale among all project members.
- Why do people micromanage?

Reasons for Micro Management

- The manager is interested in and enjoys the work
- The manager is a technical expert and feels he/she can do the job best.
- The manager may feel they did not explain the assignment clearly.
- The manager is looking for a way to stay involved with the person and or the team.
- The manager feels threatended because you have more technical knowledge.
- The manager does not have a clear understanding on how to spend project time.
- The manager wants to stay up-to-date in case somebody else asks about the work.

Overcoming Micro Management

- Don‘t be defensive when the manager asks questions.
  - Doing so make it appear as if you are hinding something and the manager will worry even more.
- Thank the micromanager for the interest and time.
  - Complaining about micromanagement will cause the micromanager to do it even more.
- Offer to explain to the micromanager how you will approach your tasks
- Work with the micromanager to develop a scheme for sharing progress and accomplishments.

Software Project Management Plan

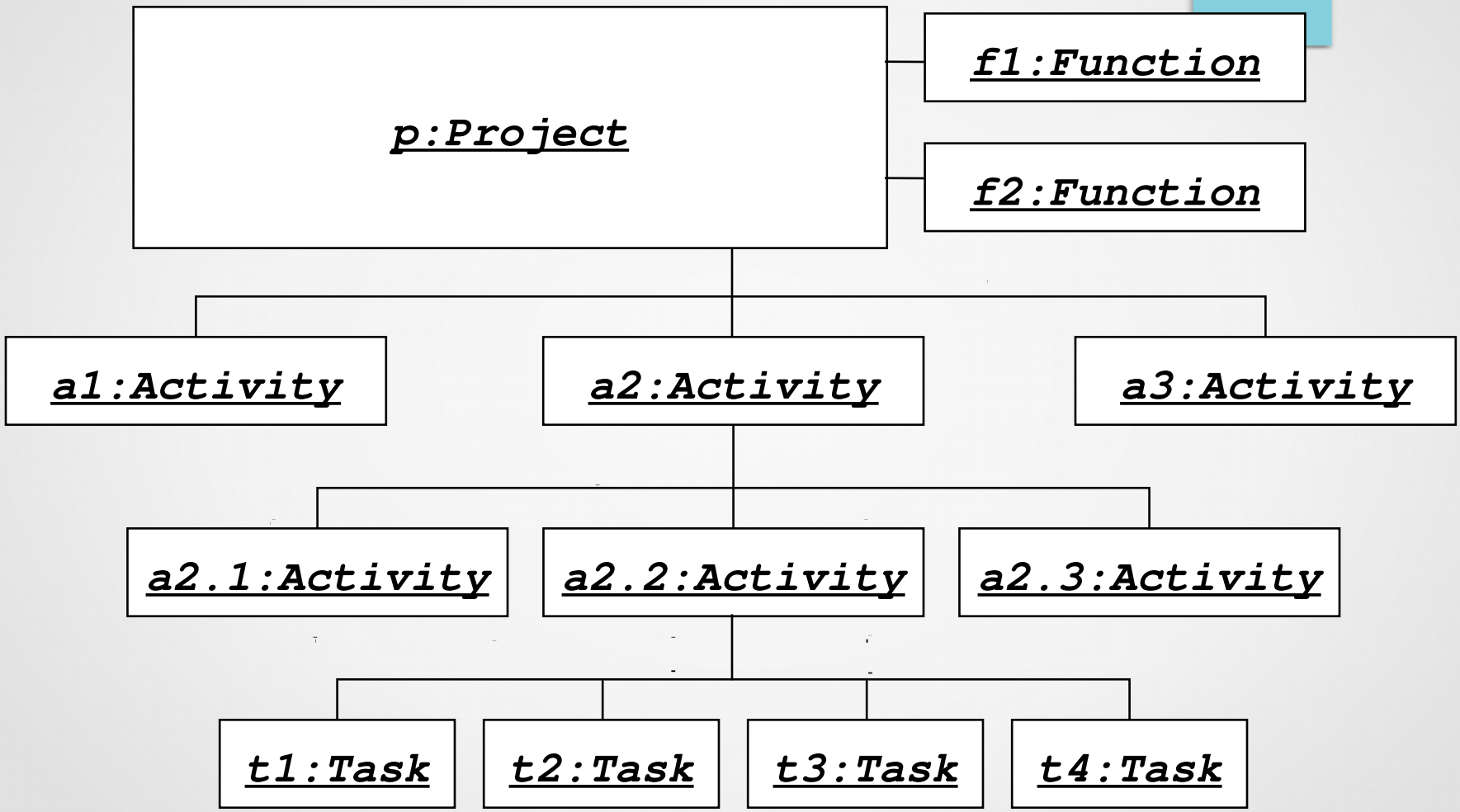
- Software Project:
  - All *technical* and *managerial* activities required to deliver the deliverables to the client.
  - A software project has a specific duration, consumes resources and produces *work products*.
  - Management categories to complete a software project:
    - Tasks, Activities, Functions

- Software Project Management Plan:
  - 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.
  - SPMP may be part of project agreement.

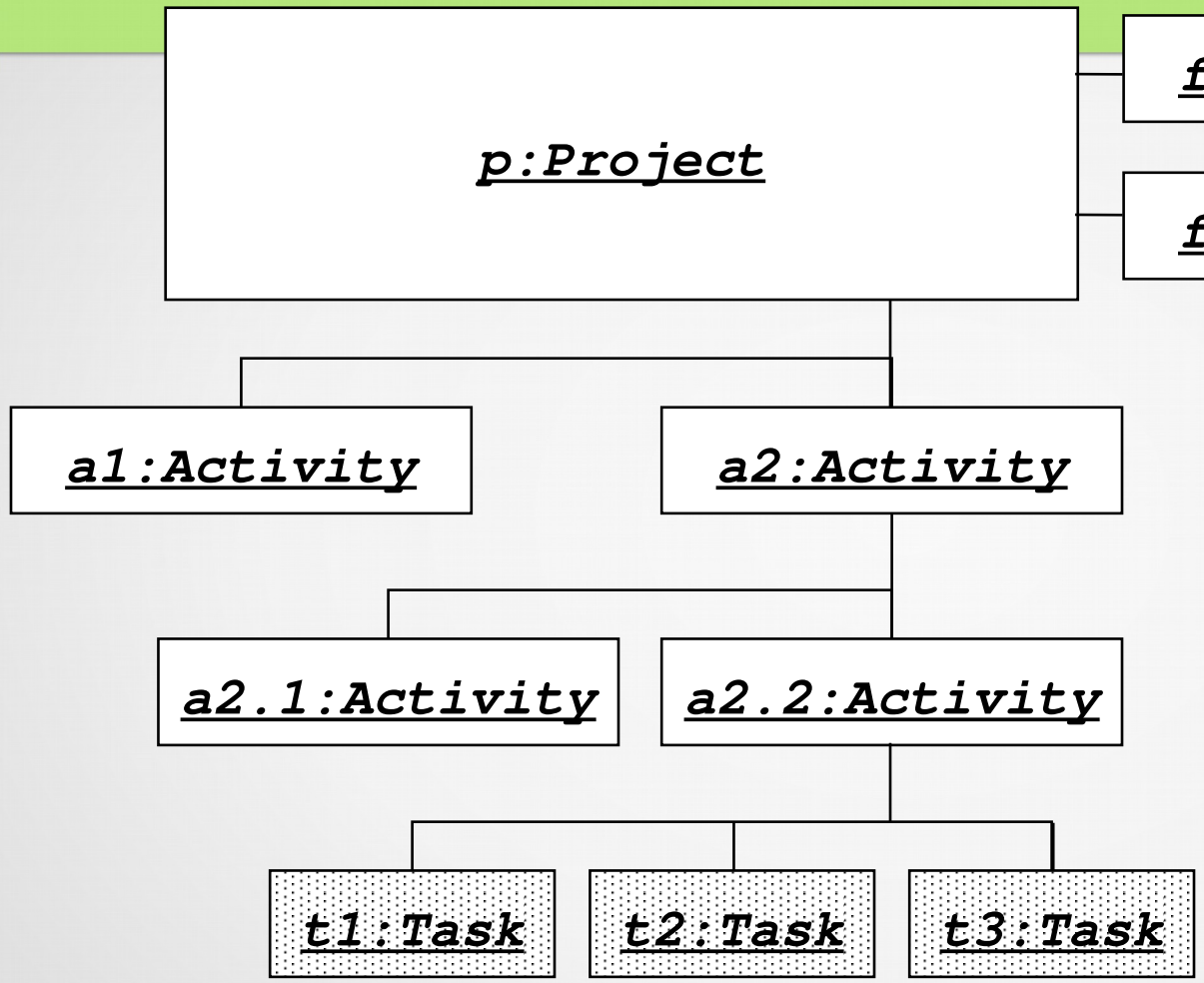
Project Agreement

- Document written for a client that defines:
  - the scope, duration, cost and deliverables for the project.
  - the exact items, quantities, delivery dates, delivery location.
- Can be a contract, a statement of work, a business plan, or a project charter.
- Client: Individual or organization that specifies the requirements and accepts the project deliverables.
- Deliverables (= Work Products that will be delivered to the client):
  - Documents
  - Demonstrations of function
  - Demonstration of nonfunctional requirements
  - Demonstrations of subsystems

Project: Functions, Activities and Tasks



Tasks



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

Task Sizes

- Finding the appropriate task size is problematic
  - Todo lists from previous projects
  - 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
- Tasks must be decomposed into sizes that allow monitoring
  - Work package usually corresponds to well defined work assignment for one worker for a week or a month.
  - Depends on nature of work and how well task is understood.

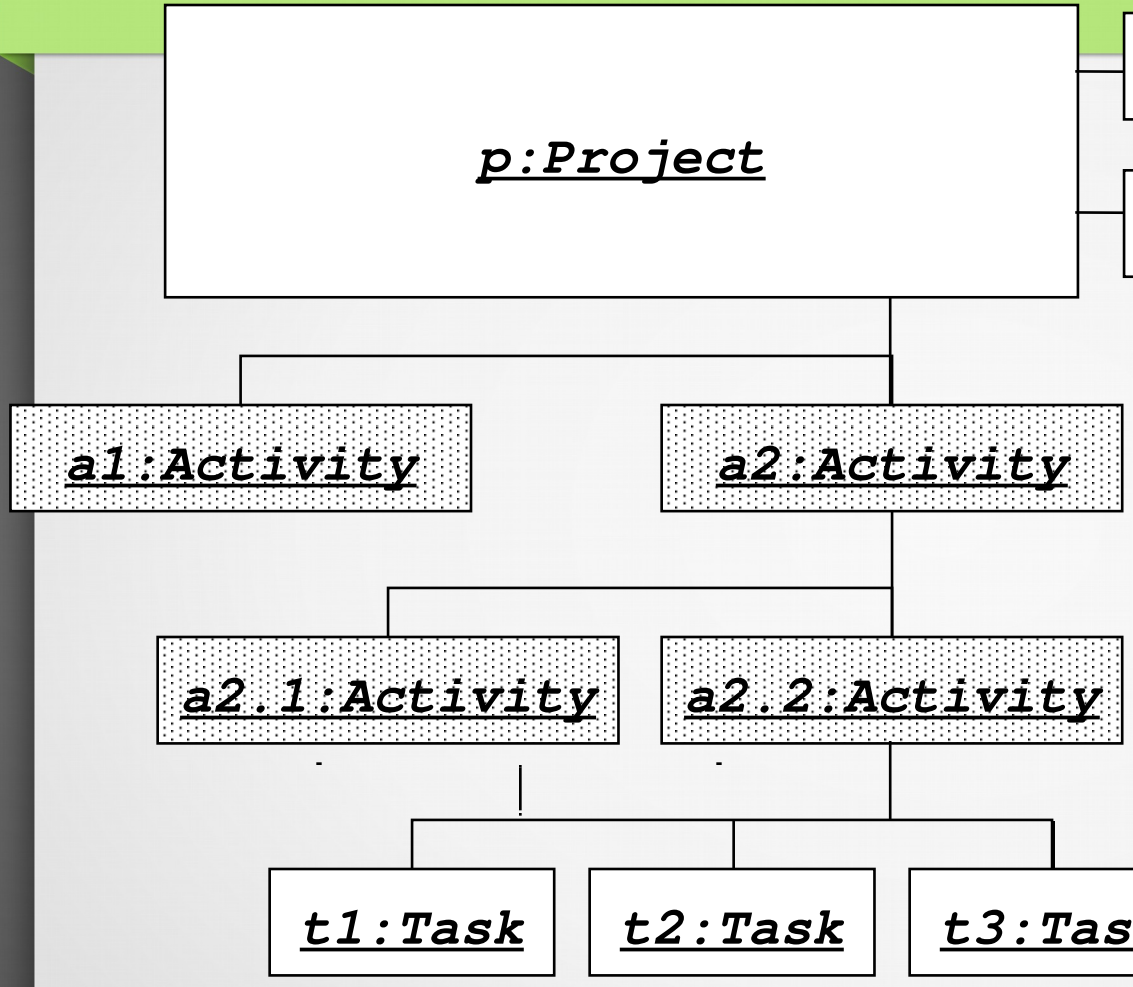
Examples of Tasks

- Unit test class “Foo”
- Test subsystem “Bla”
- Write user manual
- Write meeting minutes and post them
- Write a memo on NT vs Unix
- Schedule the code review
- Develop the project plan
- Related tasks are grouped into hierarchical sets of functions and activities.
- Action item

Action Item

- Definition: A task assigned to a person that has to be done within a week or less
- Action items
  - Appear on the agenda in the Status Section (See lecture on communication)
  - Cover: What?, Who?, When?
- Example of action items:
  - Denise unit tests class “Foo” by next week
  - Marcus develops a project plan before the next meeting
  - Bob posts the next agenda for the Simulation team meeting before Sep 10, 12noon.
  - The team develops the project plan by Sep 18

Activities



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

Structure of a Software Project Management Plan

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

2.2 Organizational Structure

- Internal management, organization chart

2.3 Organizational Interfaces

- Relations with other entities

2.4 Project Responsibilities

- Major functions and activities; nature of each; who’s in charge

Process Model

- Shows relationships among
  - Functions, activities, tasks
  - Milestones
  - Baselines
  - Reviews
  - Work breakdown structure
  - Project deliverables
  - Sign-offs
- Visualization of process model
- Project Management Aids
  - MS Project (Microsoft)
  - MAC Project (Claris)
  - EasyTrak (Planning Control International)

SPMP Part 3: Managerial Processes

3.1 Management Objectives and Priorities

- Philosophy, goals and priorities

3.2 Assumptions, Dependencies, Constraints

- External factors

3.3 Risk Management

- Identifying, assessing, tracking, contingencies for risks

3.4 Monitoring and Controlling Mechanisms

- Reporting mechanisms and formats, information flows, reviews

3.5 Staffing Plan

- Needed skills (what?, how much?, when?)

Examples of Assumptions

- There are enough cycles on the development machines
- Security will not be addressed
- There are no bugs in Together-J, the CASE Tool recommended for the project
- A demonstration of the Starnetwork system will be given by the client

Examples of Dependencies

- The database team depends on the EPC database provided by DaimlerChrysler
- The automatic code generation facility in the CASE tool depends on JDK. The current release of Together-J supports only JDK 1.1.6

Examples of Constraints

- The length of the project is 3 months. limited amount of time to build the system
- The project consists of beginners. It will take time to learn how to use the tools
- Not every project member is always up-to-date with respect to the project status
- The use of UML and a CASE tool is required
- Any new code must be written in Java
- The system must use Java JDK 1.1.6

Risk Management

- **Risk: Members in key roles drop the course.**
  - Contingency: Roles are assigned to somebody else. Functionality of the system is renegotiated with the client.
- **Risk: The project is falling behind schedule.**
  - Contingency: Extra project meetings are scheduled.
- **Risk: One subsystem does not provide the functionality needed by another subsystem.**
  - *Contingency: ?*
- **Risk: Ibutton runs only under JDK 1.2**
  - *Contingency: ?*

SPMP Part 4: Technical Process

4.1 Methods, Tools and Techniques

- Computing system, development method, team structure, etc.
- Standards, guidelines, policies.

4.2 Software Documentation

- Documentation plan, including milestones, reviews and baselines.

4.3 Project Support Functions

- Plans for functions (quality assurance, configuration management).

SPMP Part 5: Work Elements

5.1 Work Packages (Work breakdown structure)

- Project decomposed into tasks; definitions of tasks

5.2 Dependencies

- Precedence relations among functions, activities and tasks

5.3 Resource Requirements

- Estimates for resources such as personnel, computer time, special hardware, support software.

5.4 Budget and Resource Allocation

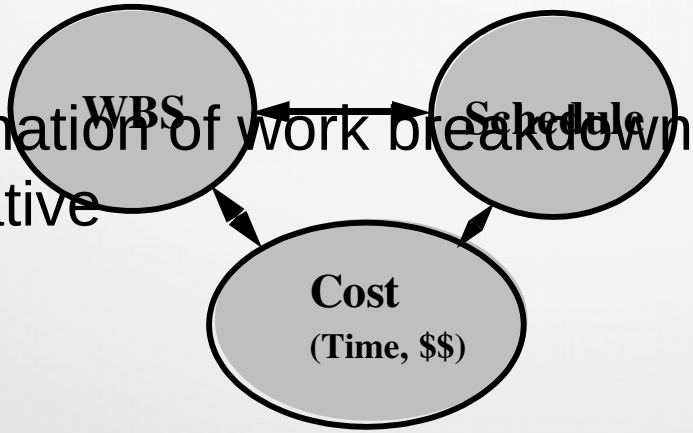
- Connect costs to functions, activities and tasks.

5.5 Schedule

- Deadlines, accounting for dependencies, required milestones

WBS Trade-offs

- Work breakdown structure influences cost and schedule
- Thresholds for establishing WBS in terms of percentage of total effort:
  - Small project (7 person-month): at least 7% or 0.5 PM
  - Medium project (300 person-month): at least 1% or 3 PMs
  - Large project (7000 person-month): at least 0.2 % or 15 PMs
- Determination of work breakdown structure is incremental and iterative



Source: Software Engineering Economics, Barry W. Boehm p. 47, Prentice Hall, N.J., 1981

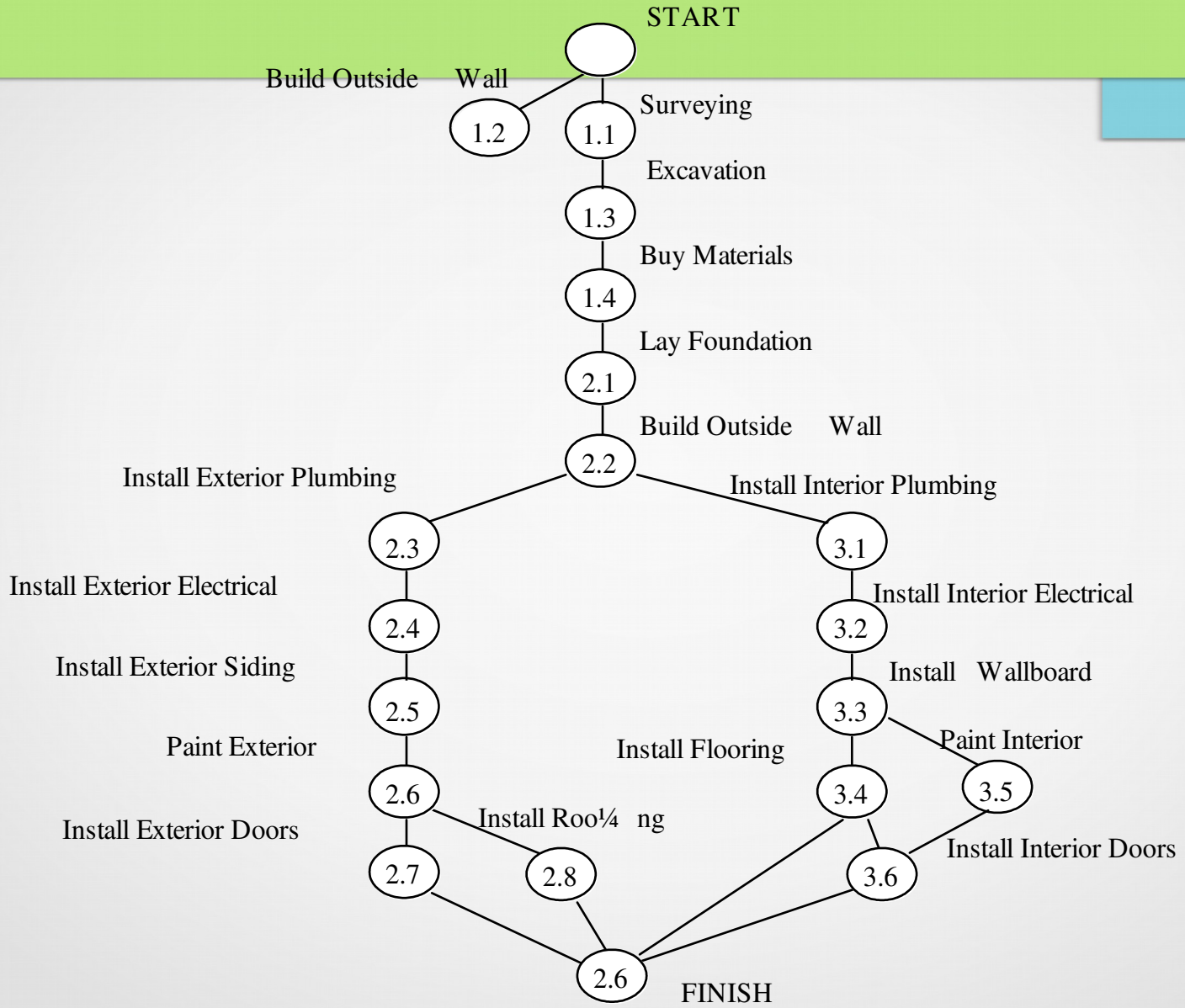
Project: Building a House

- Activity 1: Landscaping the lot
  - Task 1.1: Clearing and grubbing
  - Task 1.2: Seeding the Turf
  - Task 1.3: Planting shrubs and trees
- Activity 2: Building the House
  - Activity 2.1 : Site preparation
  - Activity 2.2: Building the exterior
  - Activity 2.3: Finishing the interior
- Activity 2.1 : Site preparation
  - Task 2.1.1: Surveying
  - Task 2.1.2: Obtaining permits
  - Task 2.1.3: Excavating
- Task 2.1.4: Obtaining materials

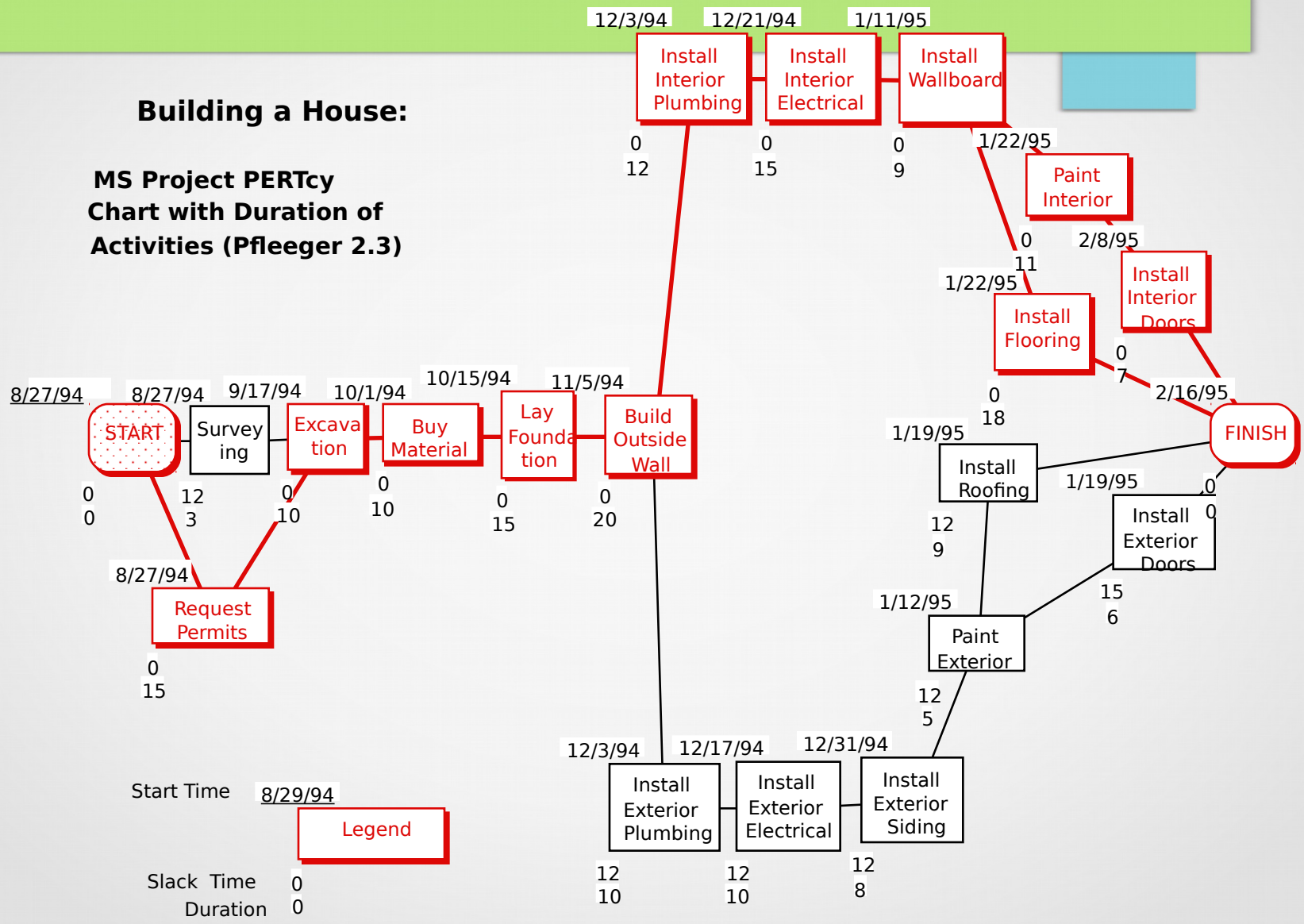
Activity 2: Building a House, ctd

- Activity 2.2: Building the exterior
  - Task 2.2.1: Foundation
  - Task 2.2.2: Outside Walls
  - Task 2.2.3: Exterior plumbing
  - Task 2.2.4: Exterior electrical work
  - Task 2.2.5: Exterior siding
  - Task 2.2.6: Exterior painting
  - Task 2.2.7: Doors and Fixtures
  - Task 2.2.8: Roof
- Activity 2.3 : Finishing the Interior
  - Task 2.3.1: Interior plumbing
  - Task 2.3.2: Interior electrical work
  - Task 2.3.3: Wallboard
  - Task 2.3.4: Interior painting
  - Task 2.3.5: Floor covering
  - Task 2.3.6: Doors and fixtures

Activity Graph for Activity “Building a House”



PERT Chart Example for "Building a House"



Project Management Heuristics

- Make sure to be able to revise or dump a project plan
  - Complex system development is a nonlinear activity
- If project goals are unclear and complex use team-based project management. In this case
  - Avoid GANTT charts and PERT charts for projects with changing requirements
  - Don’t look too far into the future
- Avoid micro management of details
- Don’t be surprise if current project management tools don’t work:
  - They were designed for projects with clear goals and fixed organizational structures

Varie:

- Esempio di documento: BOTS_SPMP.pdf

BackCasting (vs forward):

- Parto dal risultato finale desiderato alla data prevista e calcolo tutto cio' che e' necessario per la sua realizzazione, tornando via via indietro nel tempo
- Se la data di inizio e' molto prima di oggi, il progetto richiede piu' risorse di quello che credevamo
- Se e' molto dopo oggi, il progetto e' piu' facile da realizzare del previsto
- Vicino ad oggi: previsioni sensate

Vantaggi del BackCasting :

- Ogni task deriva dal fare parte di un risultato atteso, se ne conosce l'utilita', non si sprecano energie in task inutili
- Mancanze di risorse , conoscenza, tecnologie sono identificate all'inizio e non dopo, riducendo fortemnente I rischi associati
- Aree critiche o di aspettative irrealistiche sono individuate precocemente

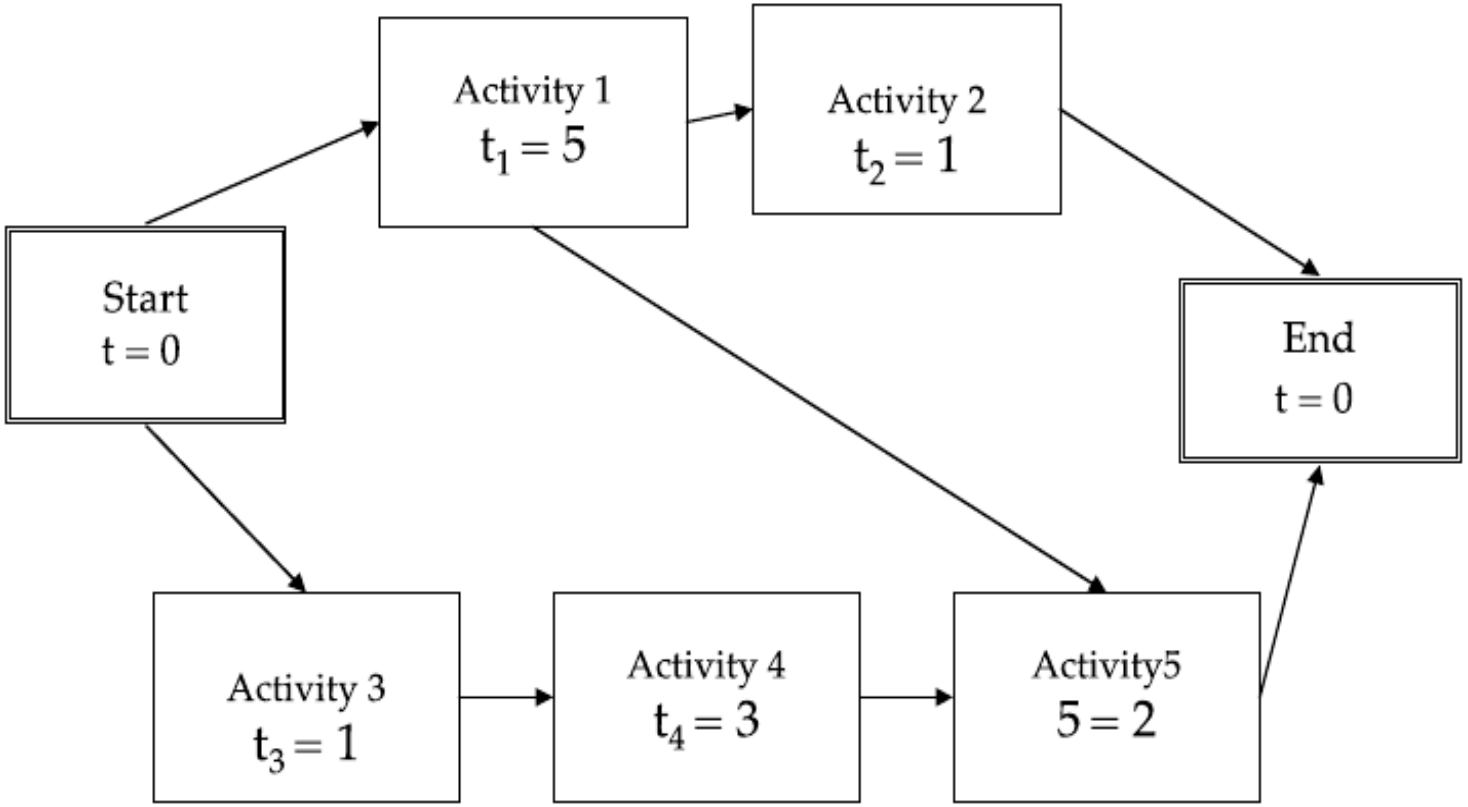
Dependency Diagrams (Review)

- Dependency diagrams consist of 3 elements
- **Event** (also called **milestone**): A significant occurrence in the life of a project
- **Activity**: Work required to move from one event to the next.
- **Span time** (also called **duration** or **elapsed time):** The actual calendar time required to complete an activity.
  - Span time parameters: people's availability, parallelizability of the activity, availability of nonpersonnel resources, ….
  - **The span time of an event is zero or small when compared with Activity)**
- A **Dependency Diagram** is drawn as a connected graph of nodes and edges.
- 2 notations to display dependency diagrams:
  - **1) Activity-on-the-Edge**(Mealy automaton)
  - **2) Activity-in-the-Node** (Moore automaton)

PERT

- PERT is an activity-on-the-edge notation
- PERT = Program Evaluation and Review Technique
- Developed in the 50s to plan the Polaris weapon system in the USA.
- PERT allows to assign optimistic, pessimistic and most likely estimates for the span times of each activity.
- You can then compute the probability to determine the likelihood that overall project duration will fall within specified limits.

Example: Activity-in -the -Node Diagram



What can we do with these Diagrams?

- Compute the project duration
- Determine activities that are critical to ensure a timely delivery

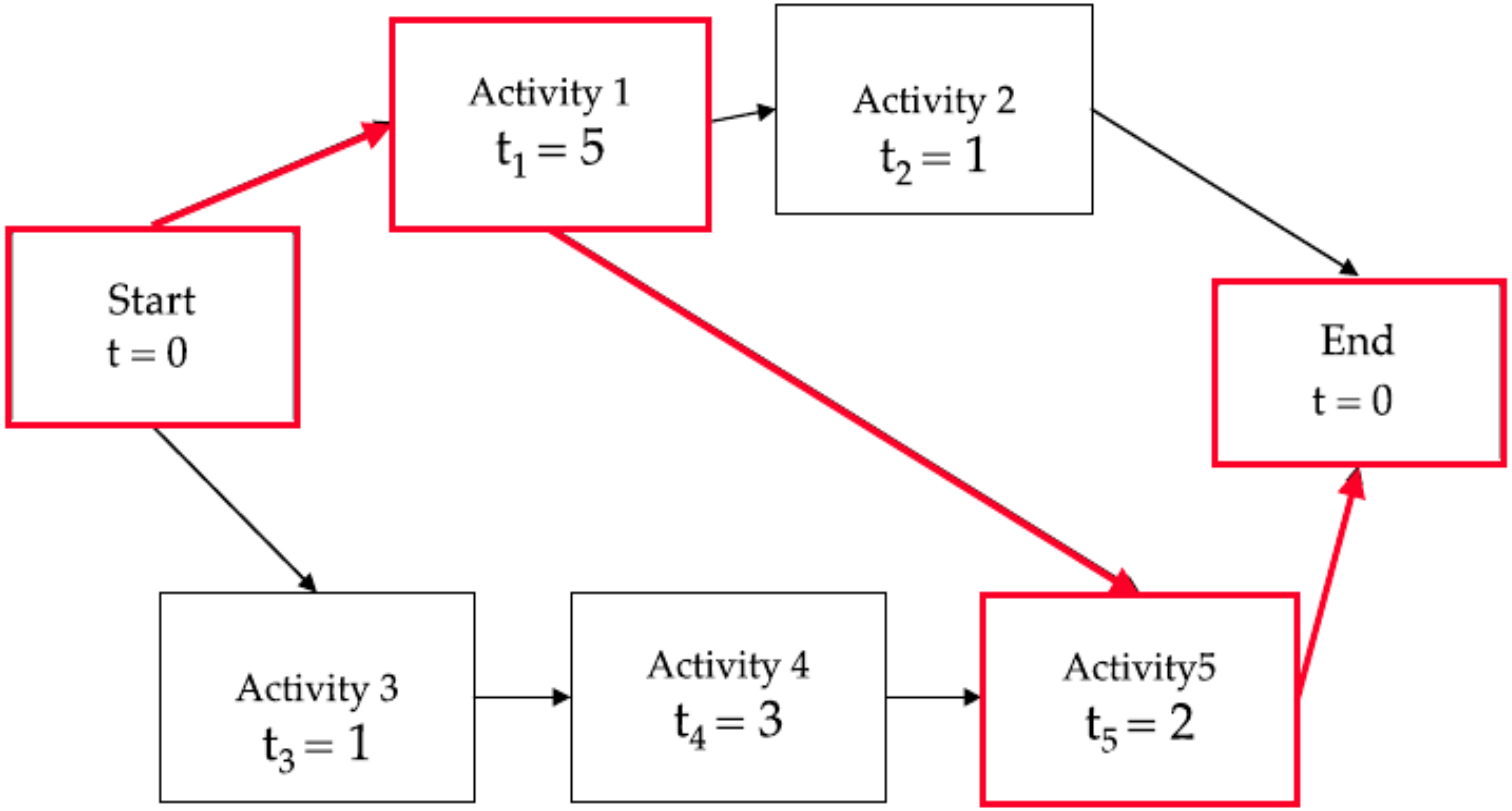
- Analyse the diagrams
  - **to find ways to shorten the project duration**
  - **To find ways to do activities in parallel**

- 2 techniques are used
  - **Forward pass (determine critical paths)**
  - **Backward pass (determine slack time)**

Definitions: Critical Path and Slack Time

- Critical path:
  - A sequence of activities that take the longest time to complete
  - The length of the critical path(s) defines how long your project will take to complete.
- Noncritical path:
  - A sequence of activities that you can delay and still finish the project in the shortest time possible.
- Slack time:
  - The maximum amount of time that you can delay an activity and still finish your project in the shortest time possible.

Example of a critical path



Critical path in bold face

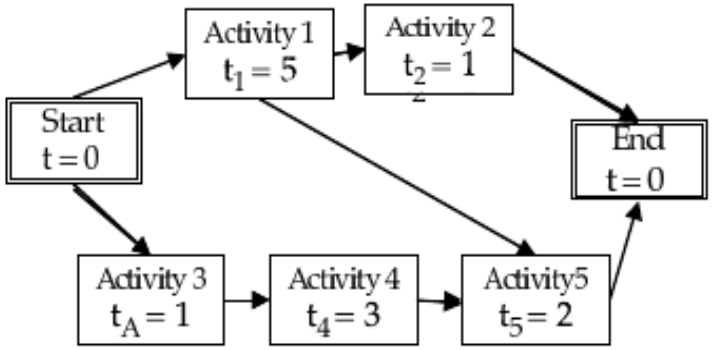
Definitions: Start and Finish Dates

- Earliest start date:
  - **The earliest date you can start an activity**
- Earliest finish date:
  - **The earliest date you can finish an activity**
- Latest start date:
  - **The latest date you can start an activity and still finish the project in the shortest time.**
- Latest finish date:
  - **The latest date you can finish an activity and still finish the project in the shortest time.**

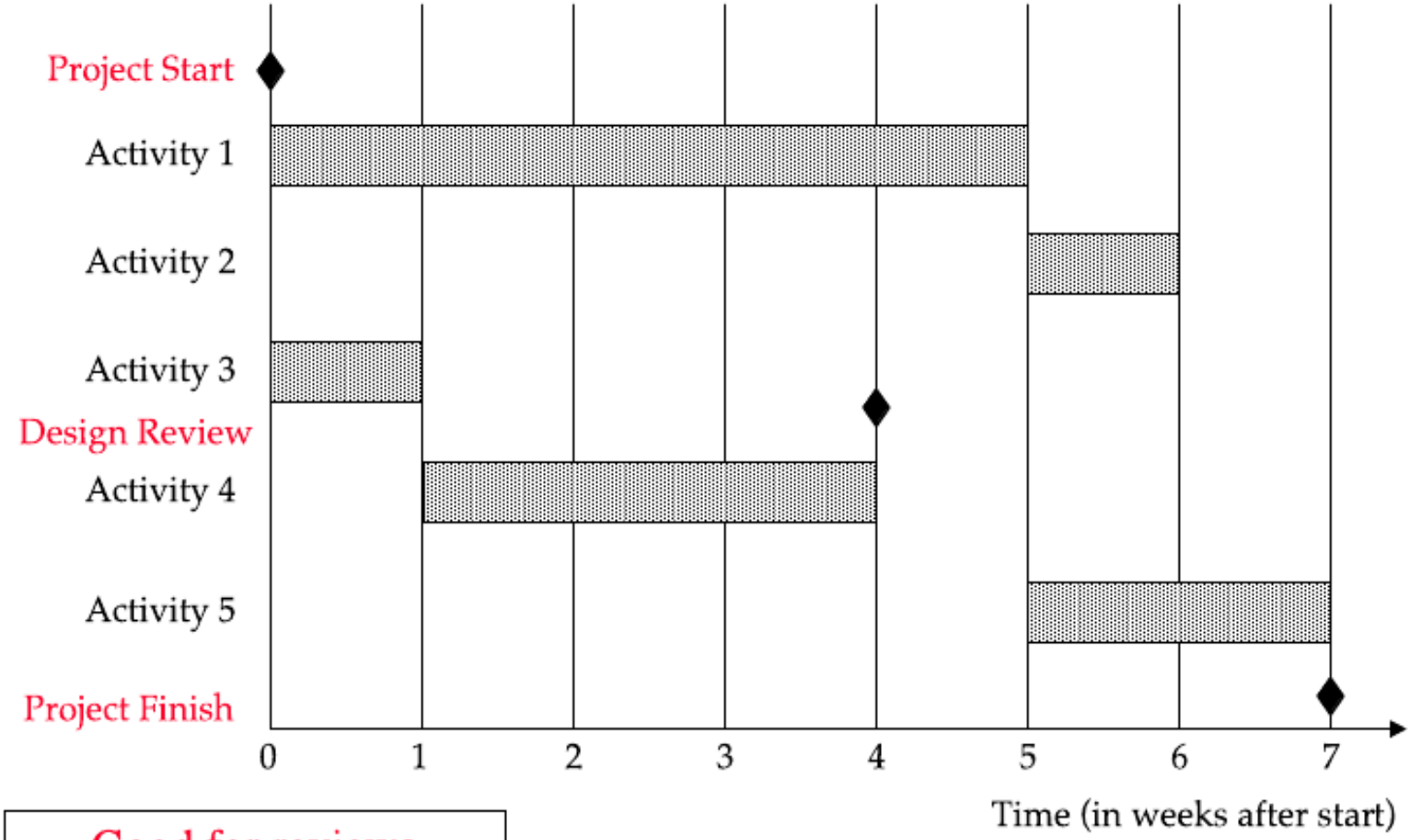
Computation of slack times

- Slack time ST of an activity A:
  - **$ST_A = LS_A - ES_A$**
  - **Subtract the earliest start date from the latest start date for each activity**
- Example: $ST_{A4} = 3 - 2 = 1$
- Slack times on the same path influence each other.
- Example: When Activity 3 is delayed by one week, activity 4 slack time becomes zero weeks.

Activity	Slack time
A1	0
A2	1
A3	1
A4	1
A5	0



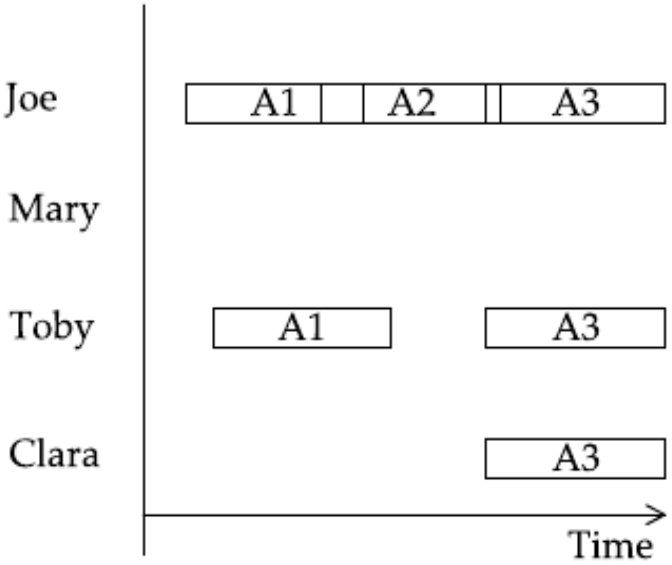
Gantt Chart *with milestones*

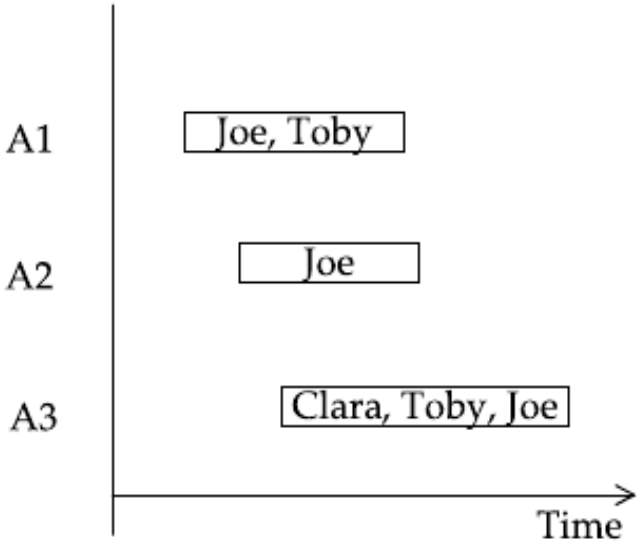


Good for reviews.

Gantt Charts come in several Flavors

- Person-Centered View
  - **To determine people's load**
- Activity-Centered View
  - **To identify teams working together on the same tasks**





Choose one view, stay with it. Usually base the view on the WBS structure

Managing Experienced Teams: Person-centered view

Managing Beginners: Activity oriented view

Tools support for Establishing Schedules

- Tool support for
  - **Graphical user interface for entering activity data**
  - **Automatic computation of critical paths**
  - **Multiple views (PERT, Gantt, table views) and switching between these views**
- Many products available. Examples
  - **Fast Track (Demo) (http://www.aecsoft.com/downloads/demo/downloads_listindex.asp?bhcp=1)**
    - **Main view: Gantt Charts**
  - **Microsoft Project (http://www.microsoft.com/office/project/default.asp)**
    - **PERT Charts, Gantt Charts, combined Milestone/Gantt Charts**
- Tool use and training beyond the scope of this class

How to reduce the planned project time

- Recheck the original span time estimates
  - **Ask other experts to check the estimates**
  - **Has the development environment changed? (batch vs interactive systems, desktop vs laptop development)**
- Hire more experienced personnel to perform the activities
  - **Trade-off: Experts work fast, but cost more**
- Consider different strategies to perform the activities
  - **Consider to Buy a work product instead of building it (Trade-off: Buy-vs-build)**
  - **Consider extern subcontractor instead of performing the work work internally**
- Try to find parallelizable activites on the critical path
  - **Continue coding while waiting for the results of a review**
  - **Risky activity, portions of the work may have to be redone.**
- Develop an entirely new strategy to solve the problem

The "Backing in" Mistake

- Definition "Backing In":
  - **You start at the last milestone of the project and work your way back toward the starting milestone, while estimating durations that will add up to the amount of the available time**
- Problems with Backing in:
  - **You probably miss activities because your focus is on meeting the time constraints rather than identifying the required work**
  - **Your span time estimates are based on what you allow activites to take, not what they actually require**
  - **The order in which you propose activities may not be the most effective one.**
- Instead, start with computing all the required times and then try to shorten the project duration

Identify when a project goes off-track

- Determine what went wrong: Why is your project got off track?
  - Behind schedule
  - Overspending of resource budgets
  - Not producing the desired deliverables

- **Identify the Reason(s):**
  - You are new on the job, this is your first project, and you made mistakes
  - Key people left the teams or new ones are joining it
  - Key people lost interest or new ones entered the picture
  - The requirements have changed
  - New technology has emerged
  - The business objectives have changed
  - Organizational priorities have shifted (for example after a merger)

What makes a Software Project successful?

❖ User involvement	20
❖ Support from upper management	15
❖ Clear Business Objectives	15
❖ Experienced Project Manager	15
❖ Shorter project phases („Small milestones“)	10
❖ Firm core requirements („basic requirements“)	5
❖ Competent Staff	5
❖ Proper Planning	5
❖ Ownership	5
❖ Other	5
	100 %

From Standish Group
http://www.standishgroup.com/sample_research/chaos1998.pdf

Become a better software project manager

- End User and Management involvement 35%
  - **Learn how to involve the customer and end users**
  - **Learn how to get support from your upper management**
- Practice project management 30 %
  - **Do as many projects as possible**
  - **Learn from your project failures**
- Focus on business objectives and requirements 20%
  - **Distinguish between core, optional and fancy requirements**