

TUM

*Software Engineering II*  
*Work Breakdown Structures*

**Prof. Bernd Brügge, Ph.D**  
Technische Universität München  
Institut für Informatik  
Lehrstuhl für Angewandte Softwaretechnik  
**<http://www.bruegge.in.tum.de>**  
23 April 2002

# *Outline of the class*

- ❖ In the last lecture we introduced the SPMP Standard
- ❖ Today and in the next lecture we focus on Section 5 of the SPMP
  - ◆ **Today: Developing a Work breakdown structure (WBS)**
  - ◆ **Next lecture:**
    - ◆ **Dependencies between tasks**
    - ◆ **Scheduling**
- ❖ Notations for visualizing dependencies
- ❖ Many heuristics and examples
  - ◆ **How detailed should a WBS be?**
  - ◆ **How can you plan a long project when things are unknown or changing all the time?**

## *What is the problem?*

- ❖ Your boss: “How long will this take?”
- ❖ You: “Between 1 and 6 months.”
- ❖ People are not happy when you respond that way.
  - ◆ You figure out that finishing anytime before six months will meet your promise.
  - ◆ Your boss figures that with some hard work you can be done in a month!
- ❖ In reality, you don't have the slightest clue how long it will take, because you don't know the work to be done.
- ❖ Solution: Use divide and conquer
  - ◆ To give a good answer you have to break the work down into activities for which you can get good timing estimates
  - ◆ From these estimates you compute the estimated project duration

# *Activities to obtain good time estimates*

- ❖ Identify the work that needs to be done
  - ◆ **Work breakdown structure (WBS), SPMP Section 5.1**
- ❖ Identify the dependency between work units
  - ◆ **Dependency Graph, SPMP Section 5.2**
- ❖ Estimate the duration of the work to be done
  - ◆ **Schedule, SPMP Section 5.5**

# *Software Project Management Plan (IEEE Std 1058)*

- ✓ 0. Front Matter
- ✓ 1. Introduction
- ✓ 2. Project Organization
- ✓ 3. Managerial Process
- ✓ 4. Technical Process
- ➔ **5. Work Elements, Schedule, Budget**
  - ➔ 5.1 Work Breakdown Structure (WBS) (today)
  - ➔ 5.2 Dependencies between tasks (today)
  - ❖ 5.3 Resource Requirements (Lecture on May 7)
  - ❖ 5.4 Budget (Lecture on June 18)
  - ➔ 5.5 Schedule (Lecture on April 30)
- ❖ Optional Inclusions

## *(From last lecture) Let's Build a House*

- ❖ What are the activities that are needed to build a house?

# *1) Identify the work to be done: Work Breakdown Structure*

- ❖ 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

**Finding these activities is a brainstorming activity.  
It requires similar activities used during requirements engineering  
And analysis (use case modeling)**

## 2) *Hierarchically organize the activities*

- ❖ 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

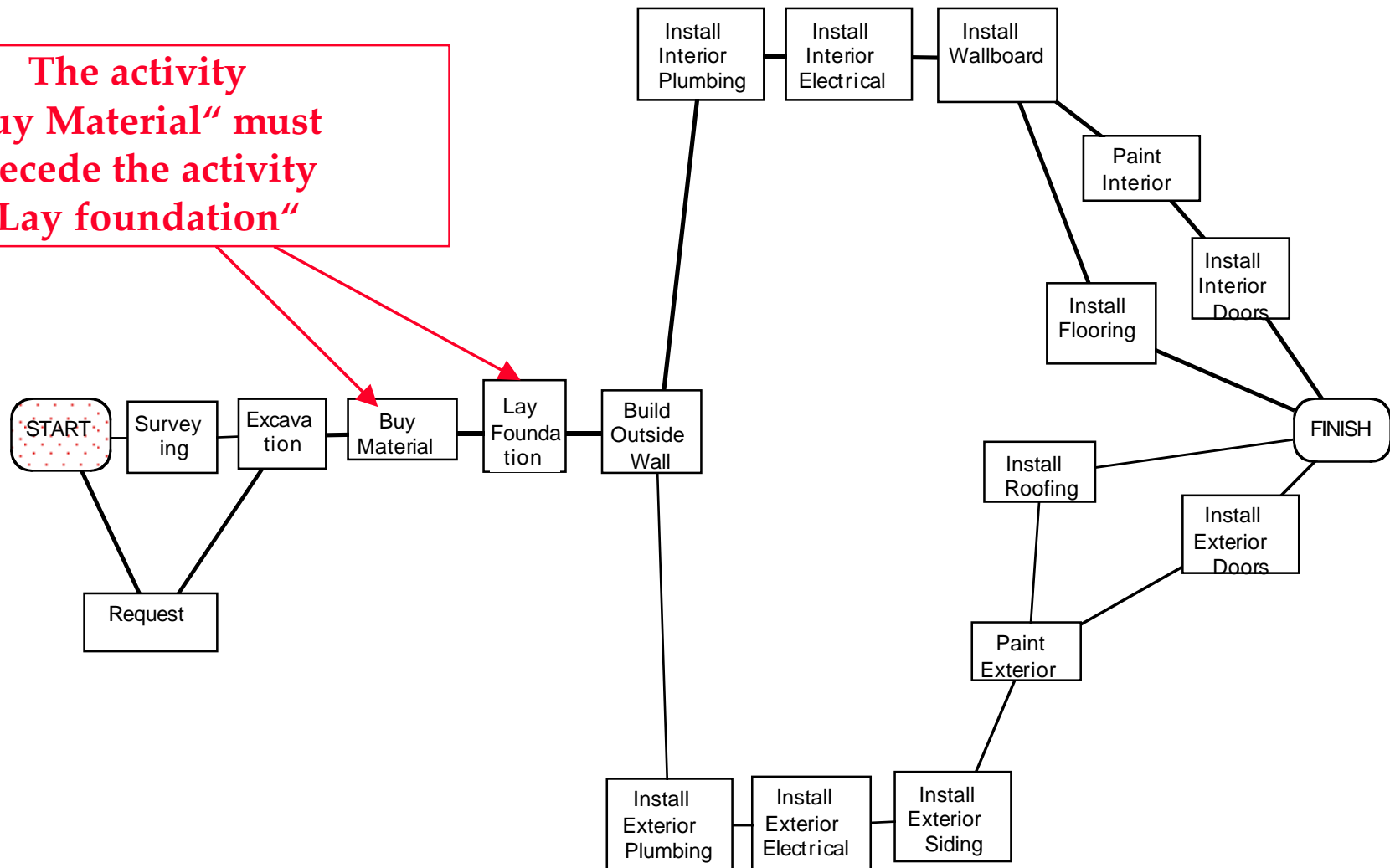
**Finding this organization involves categorization and refinement. Good after brainstorming, not during brainstorming**

### *3) Identify dependencies between tasks*

- ❖ **The work breakdown structure does not show any dependence among the activities/tasks**
  - ◆ Can we excavate before getting the permit?
  - ◆ How much time does the whole project need if I know the individual times?
    - ◆ What can be done in parallel?
  - ◆ Are there any critical activities, that can slow down the project significantly?
- ❖ **Dependencies like these are shown in the dependency graph**
  - ◆ Nodes are activities
  - ◆ Lines represent temporal dependencies

# Building a House (Dependency Graph)

The activity „Buy Material“ must Precede the activity „Lay foundation“

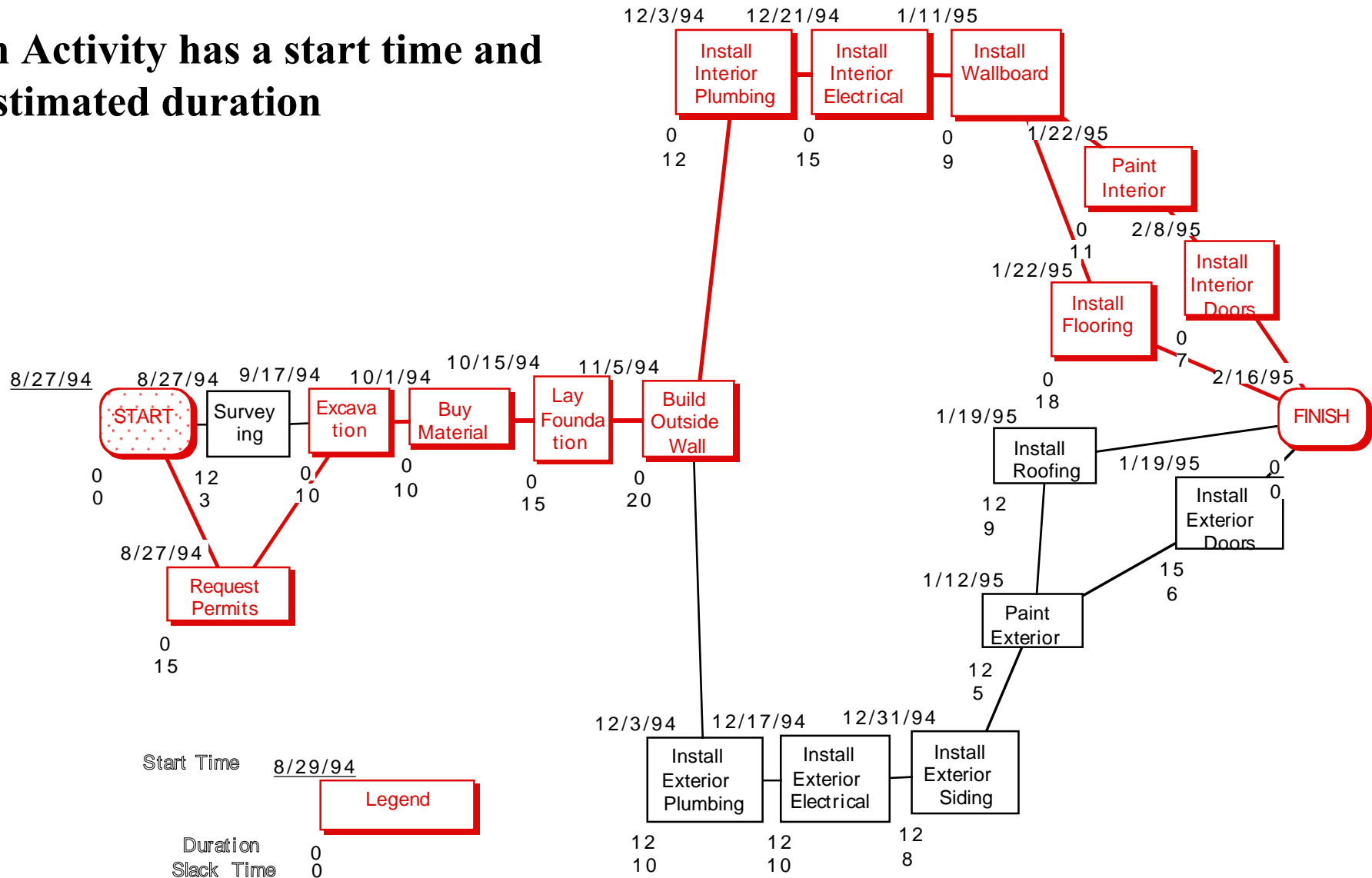


## *4) Map tasks onto time*

- ❖ Estimate starting times and durations for each of the activities in the dependency graph
- ❖ Compute the longest path through the graph: This is the estimated duration of your project

# Building a House (Schedule, PERT Chart)

Each Activity has a start time and an estimated duration



# *How do we get good estimate times?*

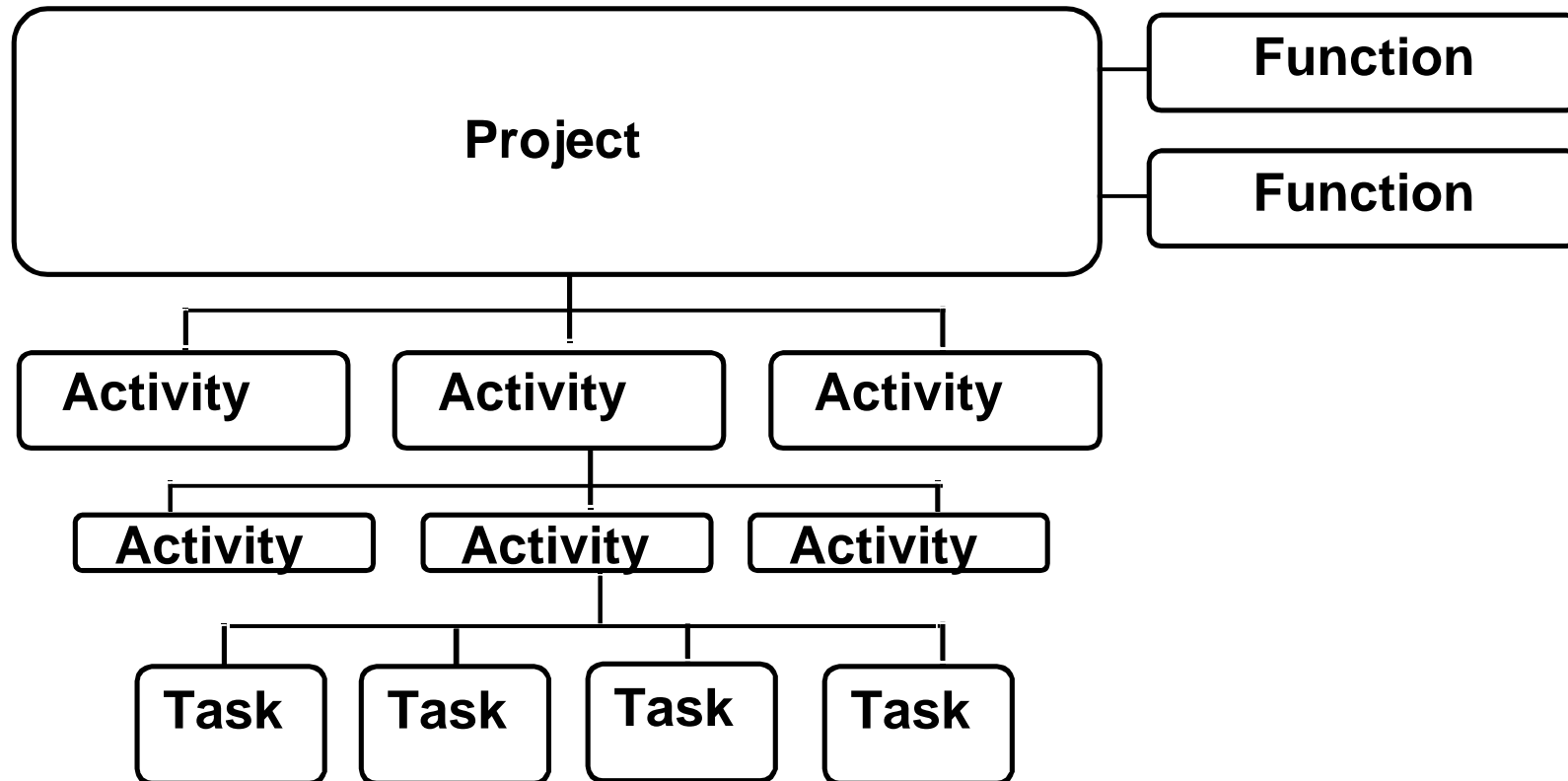
- ❖ Estimation of starting times and durations is crucial for setting up a plan.
- ❖ In this lecture we will discuss methods and heuristics on how to do it and how to establish a project schedule.
- ❖ First let us learn a few more technical terms defined in the SPMP IEEE Std 1058

# *Recall SPMP Definitions from Lecture 1*

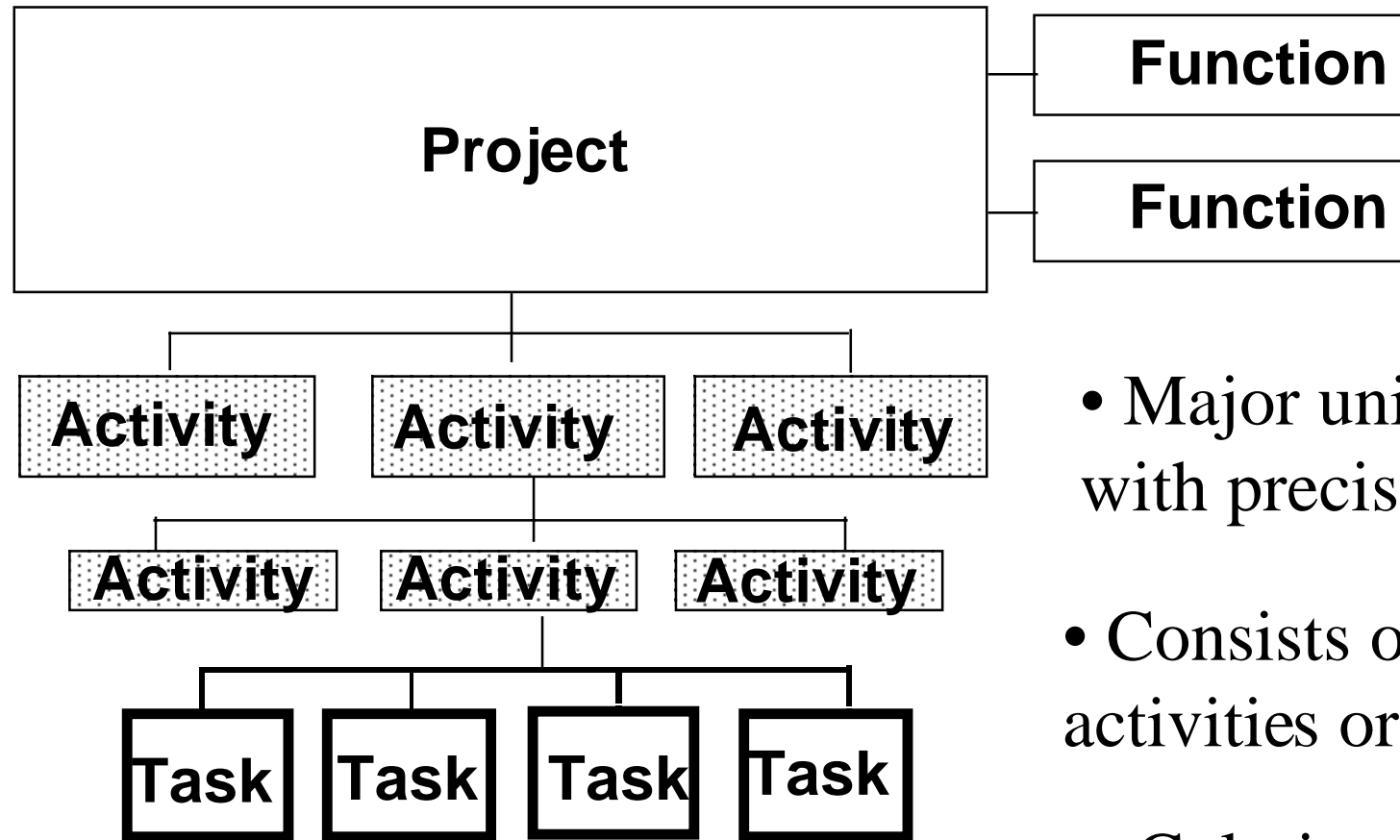
- ❖ **Project:**
  - ◆ **A Project has a duration and consists of functions, activities and tasks**
- ❖ **Work Package:**
  - ◆ **A description of the work to be accomplished in an activity or task**
- ❖ **Work Product:**
  - ◆ **Any tangible item that results from a project function, activity or task.**
- ❖ **Project Baseline:**
  - ◆ **A work product that has been formally reviewed and agreed upon.**
  - ◆ **A project baselines can only be changed through a formal change procedure**
- ❖ **Project Deliverable:**
  - ◆ **A work product to be delivered to the customer**

# *Definitions: Functions, Activities and Tasks*

A Project has a duration and consists of functions, activities and tasks



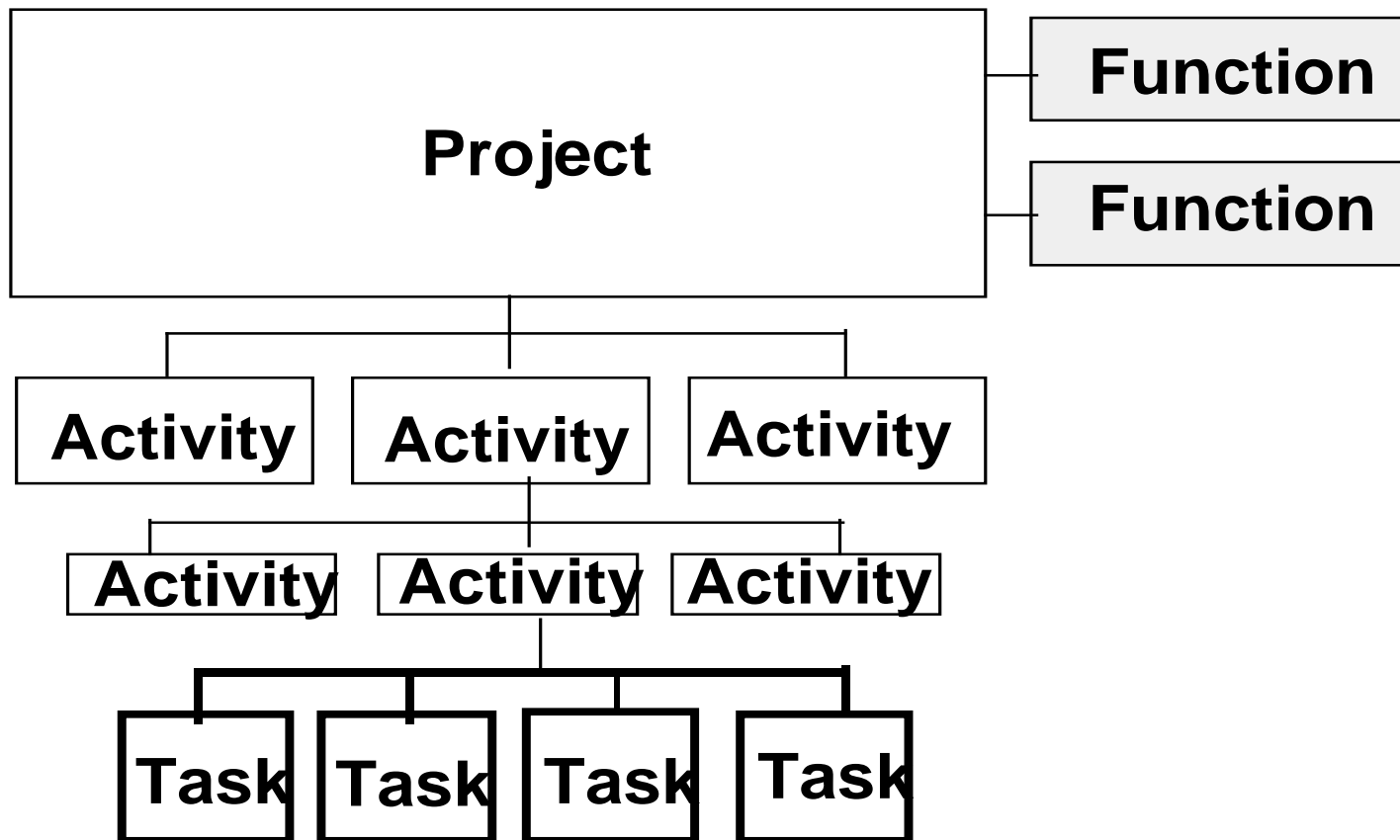
# Activities



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

# *Project Functions*

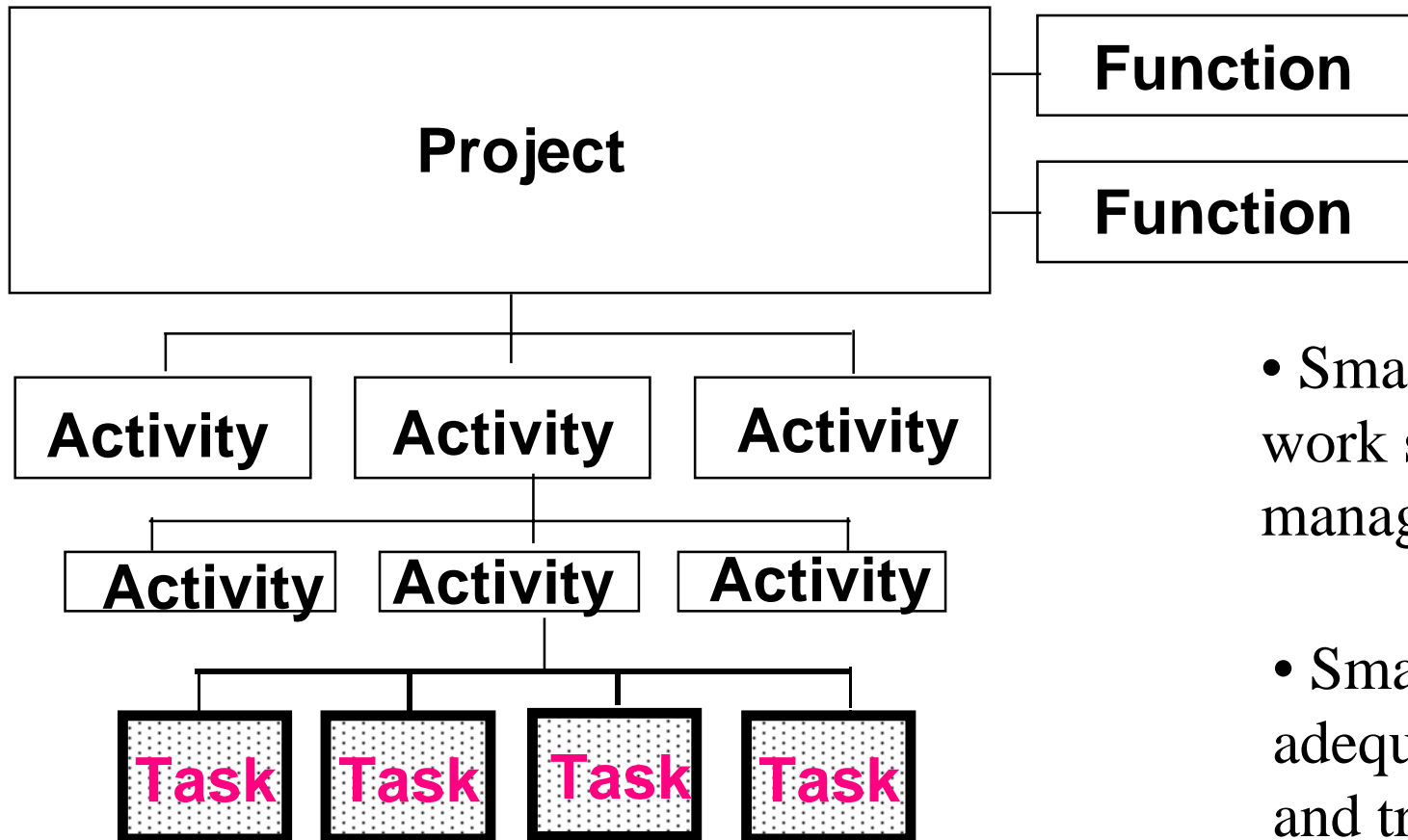
- ❖ **Definition (Project) Function:** An activity or set of activities that span the duration of the project



# *Project Functions*

- ❖ Examples:
  - ◆ Project management
  - ◆ Configuration Management
  - ◆ Documentation
  - ◆ Quality Control (Verification and validation)
  - ◆ Training
  
- ❖ Question: Is system integration a project function?
  - ◆ It Depends...
  
- ❖ Mapping of terms: Project Functions in the IEEE 1058 standard are called **Integral processes** in the IEEE 1074 standard. Sometimes also called cross-development processes

# 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)
- ❖ The description of a task is done in a Work package
  - ◆ Name, description of work to be done
  - ◆ Preconditions for starting, duration, required resources
    - ◆ Other Work packages that need to be completed before this task can be started.
  - ◆ 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 problematic
  - ◆ **Todo lists and templates 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**
  - ◆ **Depends on nature of work and how well task is understood.**
  - ◆ **Work package usually corresponds to well defined work assignment for one worker for a week or two.**
  - ◆ **Work assignments are also called action items**

# Action Item

- ❖ **Definition Action Item:** A *task* assigned to a *person* , a a to-do, to be done by a certain *time*
  - ◆ **What?, Who?, When?**
  - ◆ **Heuristics for Duration:** be done within one week or two weeks
- ❖ Action items should be tracked by the project manager
- ❖ They should appear on the meeting agenda in the Status Section
- ❖ Examples of Todos:
  - ◆ **Unit test class Foo**
  - ◆ **Develop project plan.**
- ❖ Example of an action item:
  - ◆ **Bob posts the next agenda for the context team meeting before Sep 10, 12 noon.**
  - ◆ **The test team develops the test plan by Sep 18**

# *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, ...)
  - ◆ Activities allow separation of concerns
  - ◆ Precedence relations often exist among activities

# *Approaches for Developing Work Breakdown Structures*

- ❖ There are several different approaches to develop and display a work breakdown structure. Each is effective under different circumstances
- ❖ Approaches to break activities into detail by
  - ◆ **Product component approach**
    - ◆ Examples: Design documents, manuals, the running system
  - ◆ **Functional approach**
    - ◆ Analysis, design, implementation, integration, testing, delivery, reviews
  - ◆ **Geographical area**
    - ◆ Examples: TUM team, CMU team, off-shore team, ...
  - ◆ **Organizational approach**
    - ◆ Research, product development, marketing, sales

# *When to use what approach*

- ❖ Distributed teams:
  - ◆ **Geographical area approach**
- ❖ Experienced teams:
  - ◆ **Product component approach**
- ❖ Project has mostly beginners or project manager is inexperienced:
  - ◆ **Functional approach**
- ❖ Project is a continuation of previously successful projects, no change in requirements, no new technology
  - ◆ **Organizational approach**
  
- ❖ When you choose an approach, stick with it to prevent possible overlap in categories

# *Mixing different WBS Approaches is bad*

❖ Consider the WBS for an activity „Prepare report“

❖ Functional approach:

- ◆ Write draft report
- ◆ Have draft report reviewed
- ◆ Write final report

❖ Product component approach:

- ◆ Chapter 1
- ◆ Chapter 2
- ◆ Chapter 3

❖ Don't try to mix. Why is this bad?

- ◆ Chapter 1
- ◆ Chapter 2
- ◆ Chapter 3
- ◆ Have draft report reviewed
- ◆ Write final report

**“Prepare the final version of Chapter 3”  
can be included in either of the  
categories:  
“Chapter 3” or “Write final report”**

# *How do you develop a good WBS?*

## ❖ **Top down approach:**

- ◆ **Start at the highest, top level activities and systematically develop increasing levels of detail for all activities.**

## ❖ **Brainstorming:**

- ◆ **Generate all activities you can think of that will have to be done and then group them into categories.**

## ❖ **Which one you use depends on**

- ◆ **how familiar you and your team are with the project,**
- ◆ **whether similar projects have successfully been performed in the past, and**
- ◆ **how many new methods and technologies will be used.**

# *The Top Down WBS approach*

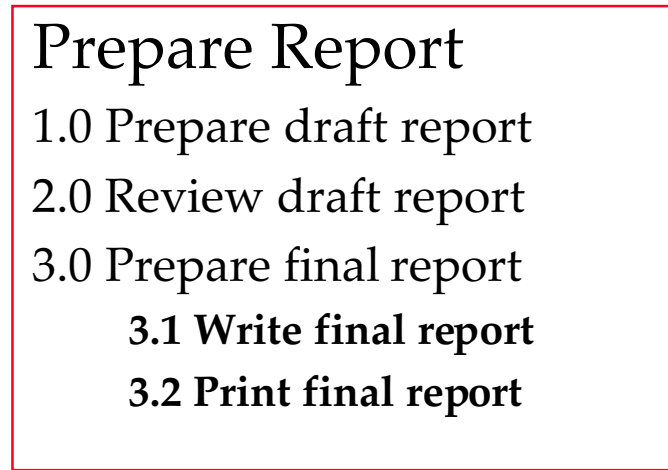
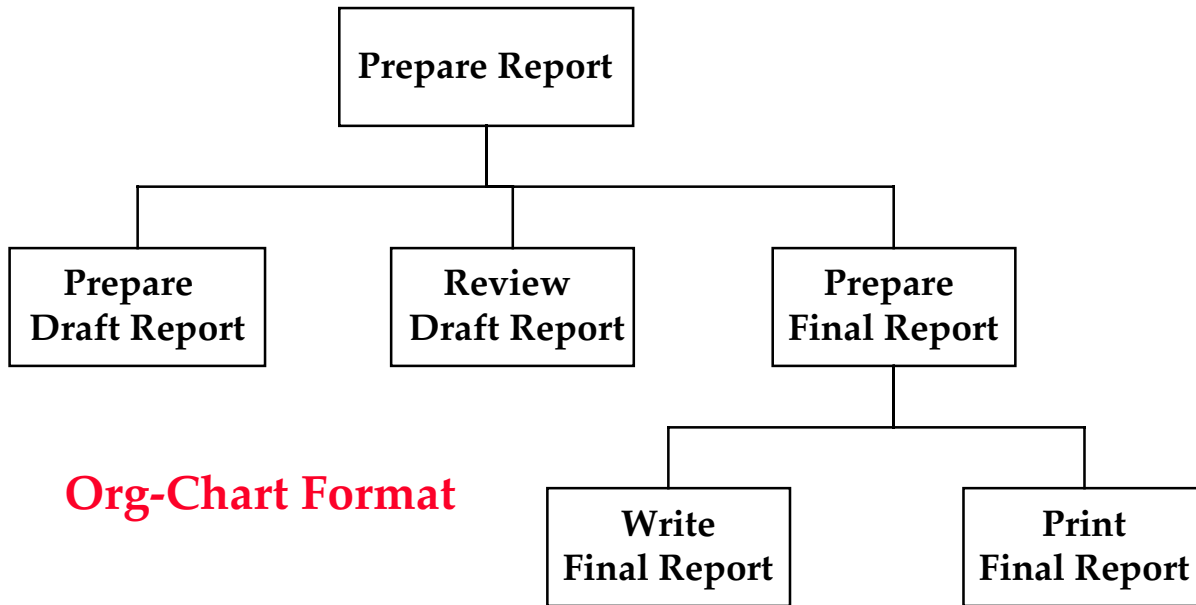
- ❖ Specify all activities required for the entire project to be finished
- ❖ Determine all task required to complete each activity
- ❖ If necessary specify subactivities required to complete each task
- ❖ Continue in this way until you have adequately detailed your project.
- ❖ **Approach is good if**
  - ◆ You are or your team is familiar with the problem.
  - ◆ You have successfully managed a similar project in the past
  - ◆ You are not introducing new methodologies, methods or tools

# *The Brainstorming WBS approach*

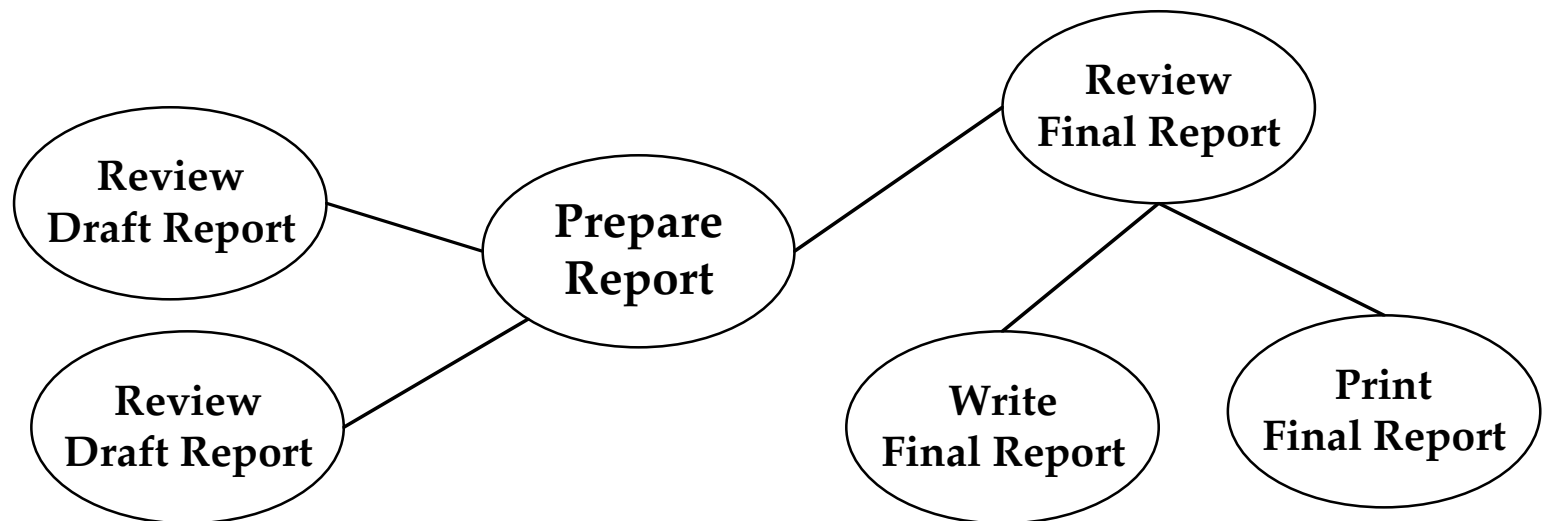
- ❖ On a single list, write any activities you think will have to be performed for your project.
- ❖ Brainstorming means you
  - ◆ **Don't worry about overlap or level of detail**
  - ◆ **Don't discuss activity wordings or other details**
  - ◆ **Don't make any judgements**
  - ◆ **Write everything down**
- ❖ Then study the list and group activities into a few major categories with common characteristics.
- ❖ If appropriate group activities under a smaller number of tasks
- ❖ Consider each category you have created and use the *top-down WBS approach* to determine any additional activities you may have overlooked.

# *Displaying Work Breakdown Structures*

- ❖ Three different formats are usually used
- ❖ Organization-chart format:
  - ◆ Effectively portrays an overview of your project and the hierarchical relationships of different activities and tasks.
- ❖ Outline format
  - ◆ Subactivities and tasks are indented
- ❖ Bubble format
  - ◆ The bubble in the center represents your project
  - ◆ Lines from the center bubble lead to activities
  - ◆ Lines from activities lead to tasks



**Bubble Format**



# *Best format for displaying WBS?*

## ❖ **Org-chart format:**

- ◆ Often good for a “bird view” of the project (executive summaries,...)
- ◆ Less effective for displaying large numbers of activities

## ❖ **Outline format:**

- ◆ Easier to read and understand if WBS contains many activities

## ❖ **Bubble format:**

- ◆ Effective for supporting the brainstorming process
- ◆ Not so good for displaying work breakdown structures to audiences who are not familiar with the project.
- ◆ Use bubble format to develop the WBS, then turn it into Org-Chart or outline format.

## ❖ **In large projects:**

- ◆ Use a combination of org-chart and outline formats:
  - ◆ **Display activities in org-chart format,**
  - ◆ **Display subactivities and tasks in outline format.**

# *Heuristics for developing high quality WBS*

- ❖ Involve the people who will be doing the work in the development of the WBS
  - ◆ **In particular involve the developers**
- ❖ Review and include information from work breakdown structures that were developed for similar projects
  - ◆ **Use a project template if possible**
- ❖ Use more than one WBS approach
  - ◆ **Do project component and functional approach simultaneously**
  - ◆ **This allows you often to identify overlooked activities**
- ❖ Make assumptions regarding uncertain activities
  - ◆ **Identify risky activities**
  - ◆ **These are often the activities that whose times are hard to estimate**
- ❖ Keep your current work breakdown structure current  
Update your WBS regularly

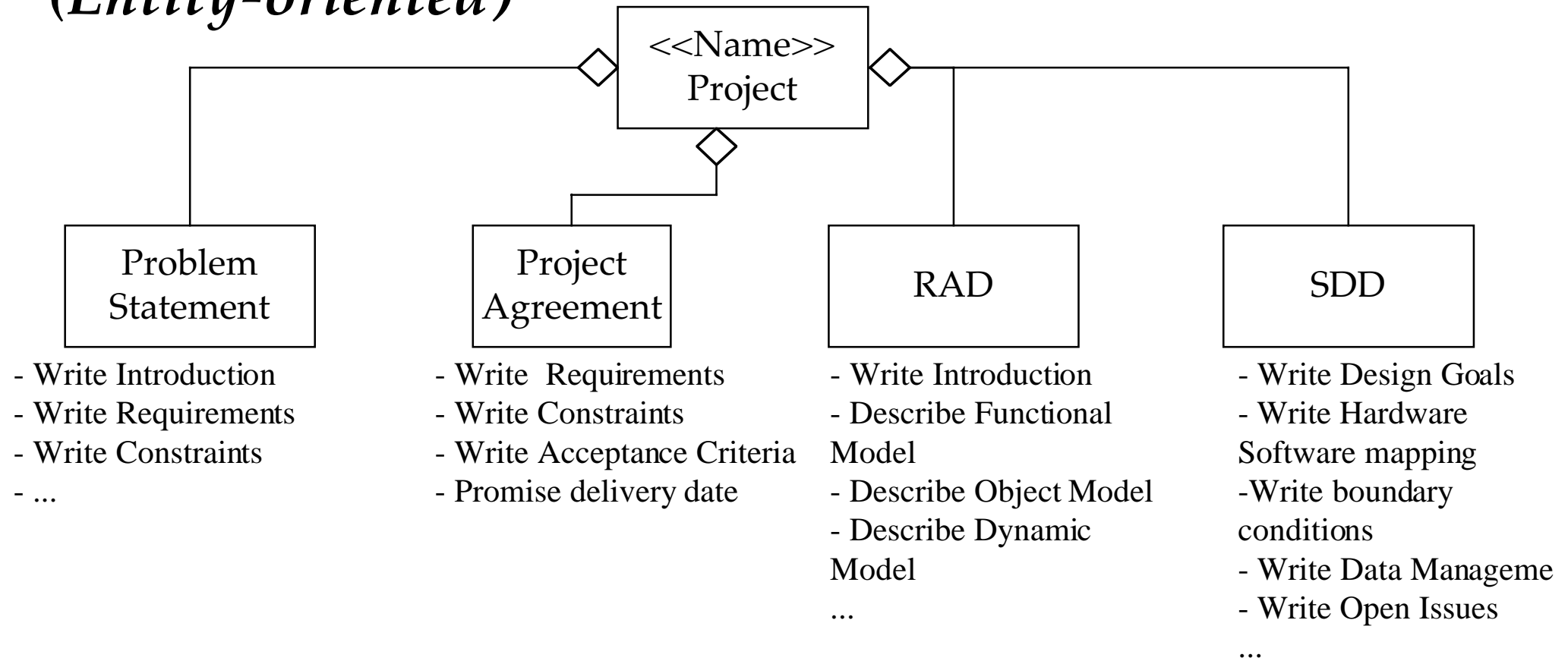
## *Heuristic: Use Templates*

- ❖ Try to derive the SPMP from a template, either an existing one or one that you start developing with this project.
  - ◆ **A template reflects the cumulative experience gained from doing numerous projects of a particular type.**
  - ◆ **Using templates can save you time and improve your accuracy**
- ❖ When developing templates, develop them for frequently performed tasks (reviews, meetings, ...). “Checklists”
  - ◆ **Develop and modify your WBS templates from previous projects that worked, not from plans that looked good.**
  - ◆ **Use templates as starting points, not as ending points**
  - ◆ **Continually update your templates to reflect the experience gained from performing different projects.**

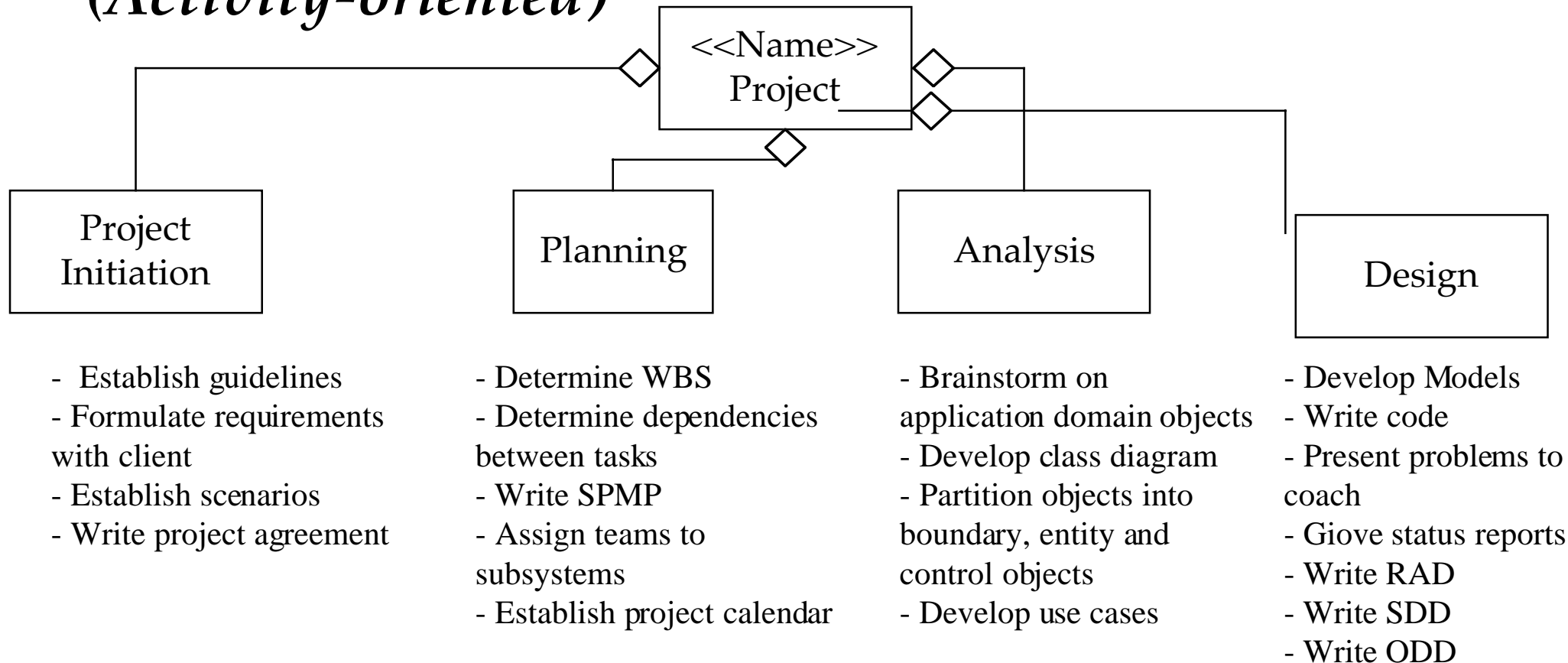
## *Heuristic: Develop always more than one WBS*

- ❖ Consider to create more several different hierarchies with different categories for your work breakdown structure.
  - ◆ **Having two or more different perspectives helps you identify activities you may overlook.**
- ❖ Good starting point are the following hierarchies:
  - ◆ **Entity-oriented decomposition**
  - ◆ **Activity-oriented decomposition**
- ❖ Example: You are running your first object-oriented project.
  - ◆ **Develop a WBS based on the project documents**
  - ◆ **Develop a WBS based on the software process activities**

# WBS Based on Project Documents (Entity-oriented)



# WBS Based on Software Process (Activity-oriented)



Question: Which activities mentioned in the WBS based on Project documents is left out in the WBS based on Software Process?

## *Heuristic: Identifying Risky activities*

- ❖ When you identify activities for a work breakdown structure, you can also identify the risks in your project.
- ❖ Risks are usually associated with “unknown information”.
- ❖ Unknown information comes in two flavors
  - ◆ **A known unknown:** Information that you don't have but someone else does.
    - ◆ Find out who has the information and determine what the information is. (Interviews, Phone calls, tasks analysis)
  - ◆ **An unknown unknown:** Information that you don't have because it does not yet exist.
    - ◆ Develop **contingency plans** for each of these risks.
    - ◆ These contingency plans need be followed when you find out the information does not exist.
- ❖ Write these risks down in SPMP section 3.3 Risk Management

# *Risk Management Examples*

- ❖ Risk: Members in key roles leave the project.
  - ◆ *Contingency Plan?*
  - ◆ Roles are assigned to somebody else. **Functionality of the system is renegotiated with the client.**
- ❖ Risk: The project is falling behind schedule.
  - ◆ *Contingency Plan?*
  - ◆ **Extra project meetings are scheduled.**
- ❖ Risk: Team 1 cannot provide functions needed by team 2.
  - ◆ *Contingency Plan?*
  - ◆ **The liaisons of both teams get together to solve this problem**
- ❖ Risk: The SPOT computer will not be available.
  - ◆ *Contingency Plan?*
  - ◆ **We will use an IPAQ instead.**

## *Risk Management Examples ctd*

- ❖ Risk: The selection of the DBMS takes too much time
  - ◆ *Contingency Plan?*
  - ◆ **The Database team uses a bridge pattern and provides a test stub to be used by the other teams for data access while the selection process goes on.**
- ❖ Risk: The customer is not available for discussing and reviewing the user interface during development.
  - ◆ *Contingency Plan?*
  - ◆ **Make the design decisions that we feel are appropriate**
- ❖ Risk: No suitable wireless library can be found.
  - ◆ *Contingency Plan?*
  - ◆ **The wireless team develops its own library**

## *Choose a single WBS format*

- ❖ Writing the WBS in different formats is good, because it allows you to identify activities that you may have overlooked
- ❖ However, after you identify these activities add them to either WBS
- ❖ Choose a *single* WBS format to be used in the SPMP and for your project:
  - ◆ **Nothing confuses people fast than trying to use two different work breakdown structures to describe the same project.**

## *How Detailed should the WBS be?*

- ❖ Sometimes the activities are not clear at all, especially in software projects:
  - ◆ Unclear requirements and/or changing requirements
  - ◆ Dependency on technology enablers that appear or are promised to appear after project kickoff
  - ◆ Simultaneous development of hardware and software (“concurrent engineering”)
- ❖ A project plan, especially for an innovative software project, should not address details beyond 3 months.
  - ◆ Even for the first 3 months project activities might not all be detailable, for example when the requirements are unclear or change or introduction of technology enablers is expected.
- ❖ How should we describe a WBS for a longer project?

## *Doing a WBS for Long-Term Projects*

- ❖ When developing a work breakdown structure for a long-term project (longer than 3 months), introduce at least two phases
- ❖ *Phase 1* (3 months): Plan your WBS in detail
  - ◆ Here list all activities that take two weeks or less to complete
- ❖ *Phase 2, Phase 3, ... (n-months)* Plan the WBS for these phases in less and less detail
  - ◆ Here list activities that you estimate will take between one and two months
- ❖ At the end of phase 1, revise the phase 2 activities to the two week level for the next 3 months.
  - ◆ Modify any future activities as necessary based on the results of your first three months work.
- ❖ Continue to revise the SPMP this way throughout the project. (SPMP as an “evolving” document)

# *Phases and large Projects*

- ❖ Project-Initiation Phase
- ❖ Steady State Phase
  - ◆ **Initial Planning phase**
- ❖ Project-Termination Phase

# *Project-Initiation Phase*

- ❖ Fred Brooks, The mythical months
- ❖ Activities
  - ◆ Meet with client, develop the scenarios (as-is, visionary) for problem statement
  - ◆ Develop an initial top level design: System as a set of subsystems.
  - ◆ Establish staffing plan (flat staffing, ramping up)
  - ◆ Identify human resources: existing employees, new employees.
  - ◆ Hire team members
  - ◆ Assign a subsystem to each team. Establish two additional cross-functional teams: Architecture&Documentation.
  - ◆ Write problem statement (with client and other stake holders, involve project members early)
  - ◆ Write initial SPMP with WBS, without schedule, without budget.
  - ◆ Get project plan approved
  - ◆ Kick project off with 2 documents: Problem statement and SPMP
- ❖ Duration: About 4 weeks
- ❖ When?
  - ◆ **Before project kickoff**

# *Initial Planning Phase*

- ❖ Usually after project kickoff, often called “planning phase”
- ❖ Activities
  - ◆ Do innovation management on technology enablers that might influence the design or nonfunctional requirements
  - ◆ Revise requirements and initial design if necessary
  - ◆ Revise team structure, reassign team members if necessary
  - ◆ Revise WBS and dependencies
  - ◆ Establish cost and scheduling information
  - ◆ Agree with client on requirements, duration and cost of the project (write this in a “project agreement”, a companion document to the SPMP)
- ❖ Duration: About 2 weeks time.
- ❖ When?
  - ◆ Parallel to “requirements elicitation phase”

# *Project-Termination Phase*

- ❖ Do a project-review: “What went right, what went wrong”
  - ◆ also often called “project post-mortem review”
- ❖ Based on input from the post-mortem session
  - ◆ Revise your software process, identify in particular any new activities that happened in the project
  - ◆ Revise your project kickoff activities
  - ◆ Revise the SPMP template (to be reused for your next project)

# *Where are we?*

SPMP IEEE Std 1058

- ✓ 0. Front Matter
- ✓ 1. Introduction
- ✓ 2. Project Organization
- ✓ 3. Managerial Process
- ✓ 4. Technical Process
- ➔ 5. Work Elements, Schedule, Budget
  - ✓ 5.1 Work Breakdown Structure (WBS)
  - ◆ 5.2 Dependencies between tasks
  - ◆ 5.3 Resource Requirements
  - ◆ 5.4 Budget ( => Lecture on cost estimation)
  - ◆ 5.5 Schedule
- ❖ Optional Inclusions

# *Exercises*

- ❖ Homework 1 (Due May 7):
  - ◆ **Model activities, functions and tasks as a UML class diagram (Start with drawings and definitions from slides 15-22 of this lecture and extend Figure 11-2 in [Bruegge-Dutoit])**

# Summary

- ❖ Software Project Management Plan, Section 5:
  - 5.1 Work Breakdown Structure (Today)
  - 5.2 Dependencies between tasks (=> Lecture on April 30)
  - ❖ 5.3 Resource Requirements (=> Lecture on May 7)
  - ❖ 5.4 Budget (=> Lecture on June 18)
  - 5.5 Schedule (=> Lecture on April 30)
- ❖ **Work Breakdown Structure (WBS):** Set of activities to do (“use cases”)
- ❖ **Dependency Graph:** Identification of dependency relationships between activities identified in the WBS
- ❖ **Schedule:** Dependency graph decorated with time estimates for each activity
- ❖ **PERT:** One of the first techniques proposed to analyse complex dependency graphs and schedules
- ❖ **Gantt Chart:** Notation used to visualize schedule