

# TOPICS

**Overview**

**Metrics**

**Estimation**

**Planning**



# **SOFTWARE PROJECT PLANNING**

**What Software Project Planning Involves**

**Risk Analysis**

**Risk Management**

**Risk Monitoring - Project Tracking**

**Software Project Scheduling**

**Typical Task Network**

**Approaches to Project Tracking**

**Software Acquisition**

**Software Acquisition Decision Tree**

**Software Re-Engineering**

**Organizational Planning**

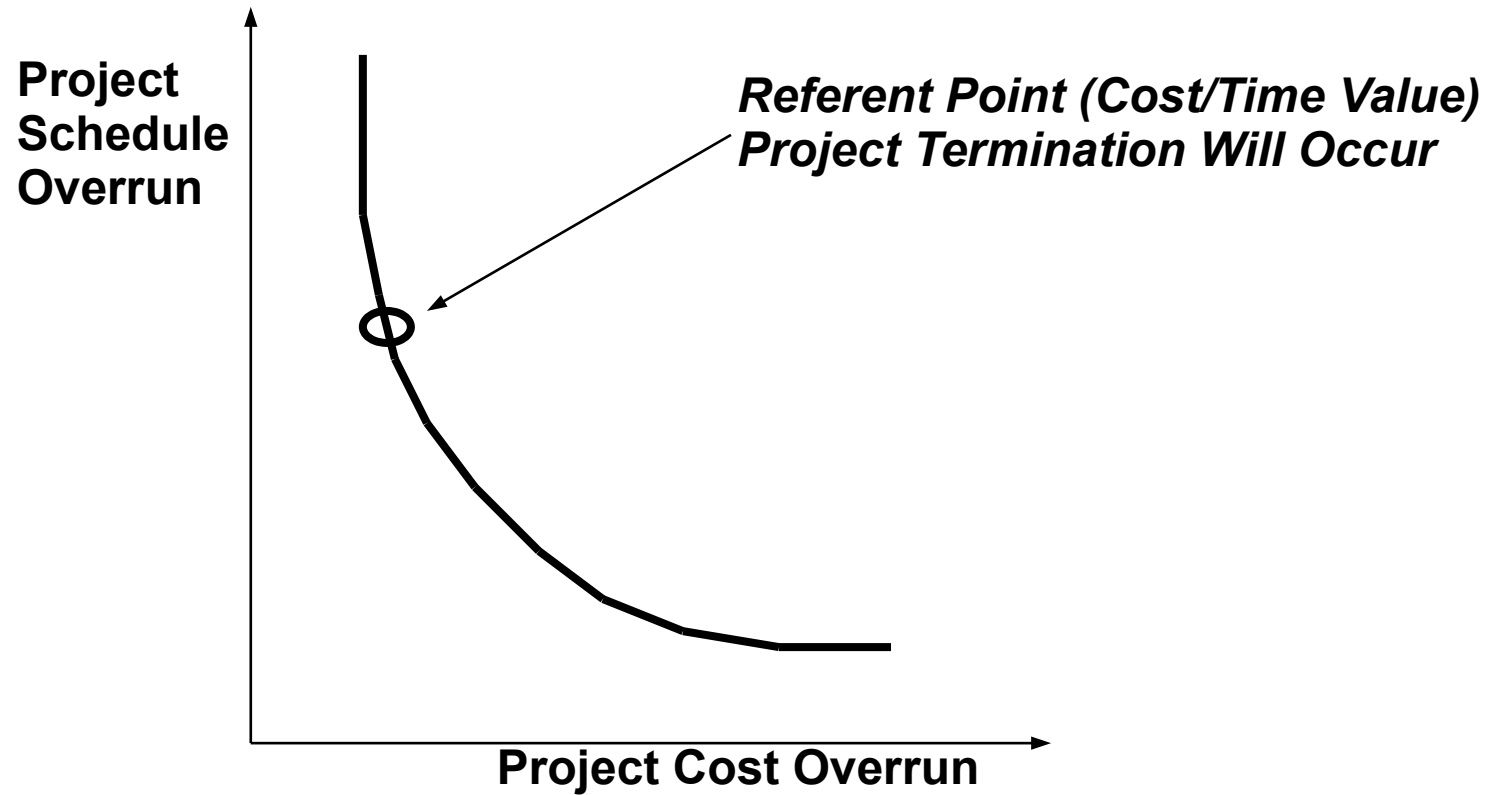
**Enhancements to a Good Organization**

**The Software Project Plan (SPP)**

# What Software Project Planning Involves

1. *Estimation*
2. **Risk Analysis**
3. **Scheduling**
4. **Acquisition Decision Making**
5. **Re-Engineering**
6. **Organizational Planning**

# Risk Analysis



# **Risk Management**

**Create risk management and monitoring plan**

**For each risk triplet, define the risk management steps**

**Risk management incurs additional project cost**

**For larger projects, there may be 30-40 risks identified**

## **Example**

**Assume:**

**Risk = High staff turnover**

**Likelihood of occurrence = 70%**

**Impact = Increase project time by 15%, project cost by 12%**

**Risk Management steps may be:**

- 1. Identify high turnover causes**
- 2. Reduce causes before project starts**
- 3. Develop techniques to assure work continuity in light of turnover**

# **Risk Monitoring - Project Tracking**

- 1. Determine if predicted risk occurs**
- 2. Properly apply risk aversion steps**
- 3. Collect info for future risk analysis**

# **Software Project Scheduling**

**People-work relationships**

**Task definition and parallelism**

**Effort distribution**

**Scheduling methods**

**An example**

# **Software Project Scheduling**

## ***People-Work Relationships***

**Adding people to a project when behind schedule is counterproductive (*adding people to a late project makes it later*)**

**Using fewer people over a longer period of time is more beneficial than lots of people for a shorter period of time**

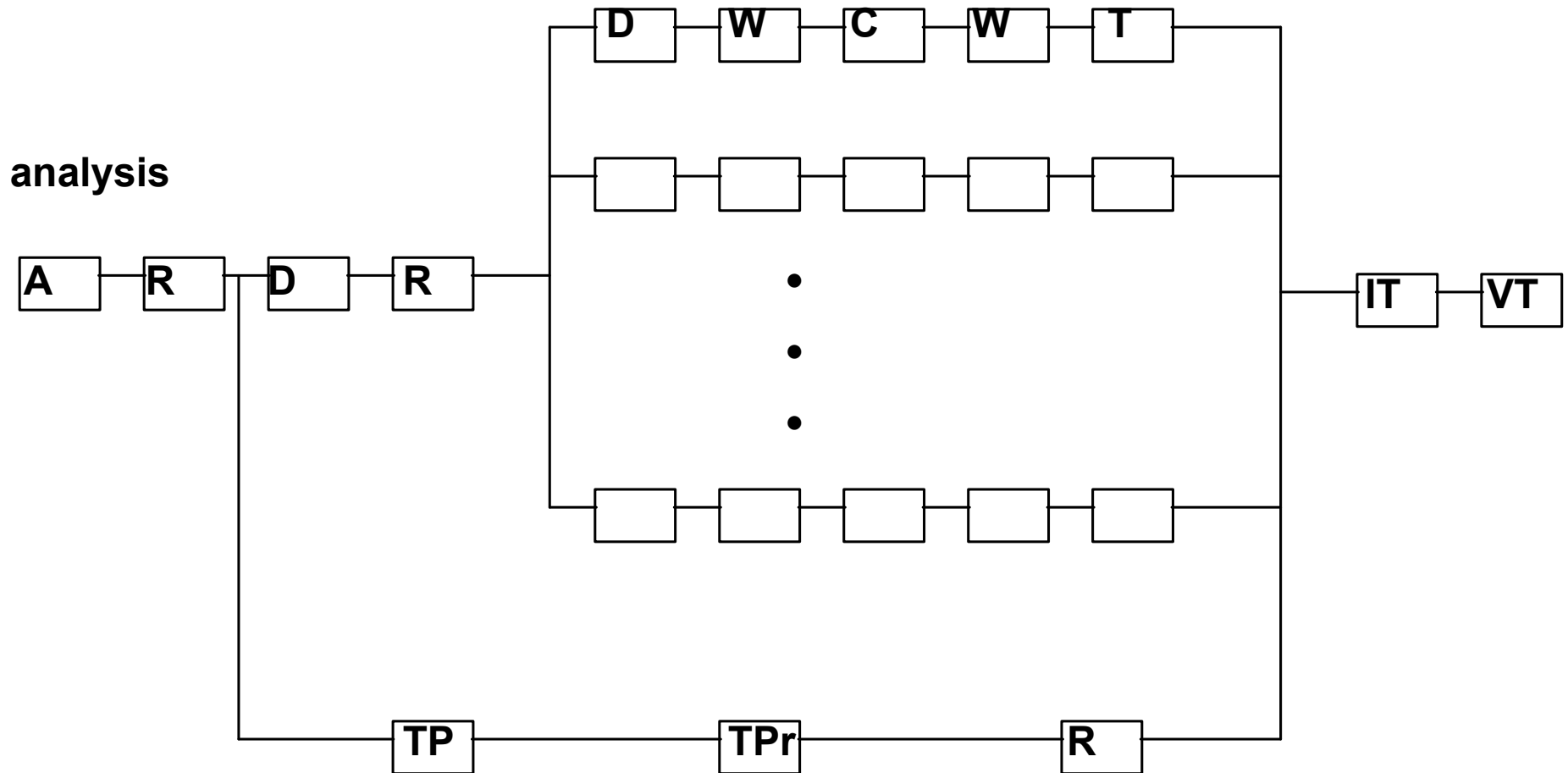
**Use of small, tightly-knit teams is productive**

**Inspire creativity and self-motivation within the structure of the project**



# Software Project Scheduling

## *Task Definition and Parallelism*



# **Software Project Scheduling**

## ***Task Definition and Parallelism***

### ***Initial Sequential Events***

**Milestone 1 Occurs After --**

**System analysis and specification**

**System requirements review**

**Milestone 2 Occurs After --**

**System architecture and data design**

**System preliminary design review**

# **Software Project Scheduling**

## ***Task Definition and Parallelism***

***Parallel Events for Each Subfunction***

**Milestone P1 Occurs After --**

**Procedural design**

**Design walkthrough**

**Milestone P2 Occurs After --**

**Coding**

**Code walkthrough**

**Milestone P3 Occurs After --**

**Unit testing**

# **Software Project Scheduling**

## ***Task Definition and Parallelism***

***System Testing Activities Can Be Performed In Parallel***

**Testing Milestone (After Unit Testing) --**

**System test planning**

**System test procedure**

**System test review**

# **Software Project Scheduling**

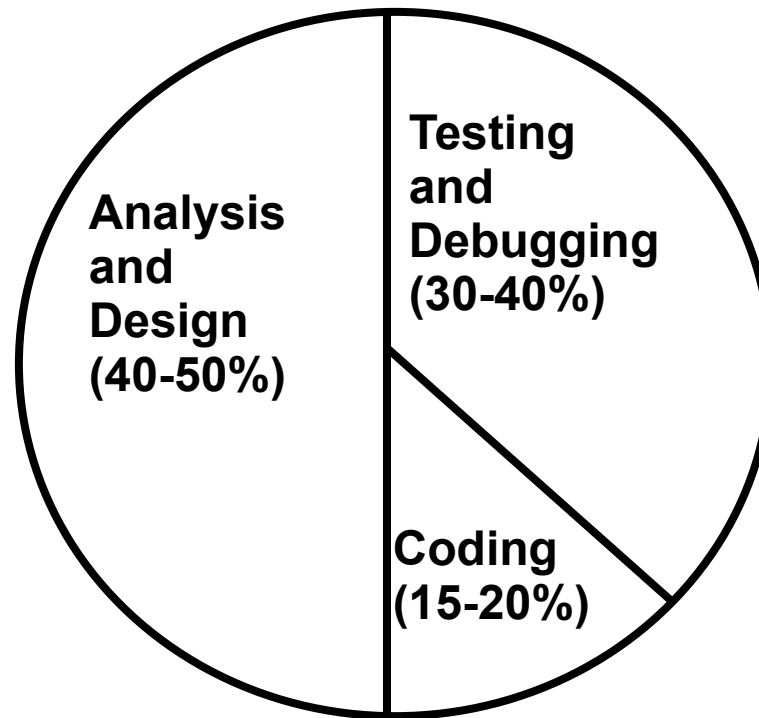
## ***Task Definition and Parallelism***

**Integration Test Milestone - completed after  
system is assembled**

**Validation Test Milestone - completed last**

# **Software Project Scheduling**

## ***Effort Distribution***



# **Software Project Scheduling**

## ***Scheduling Methods***

**PERT - *Program Evaluation and Review Technique***

**CPM - *Critical Path Method***

**PERT and CPM are:**

**Usually presented pictorially**

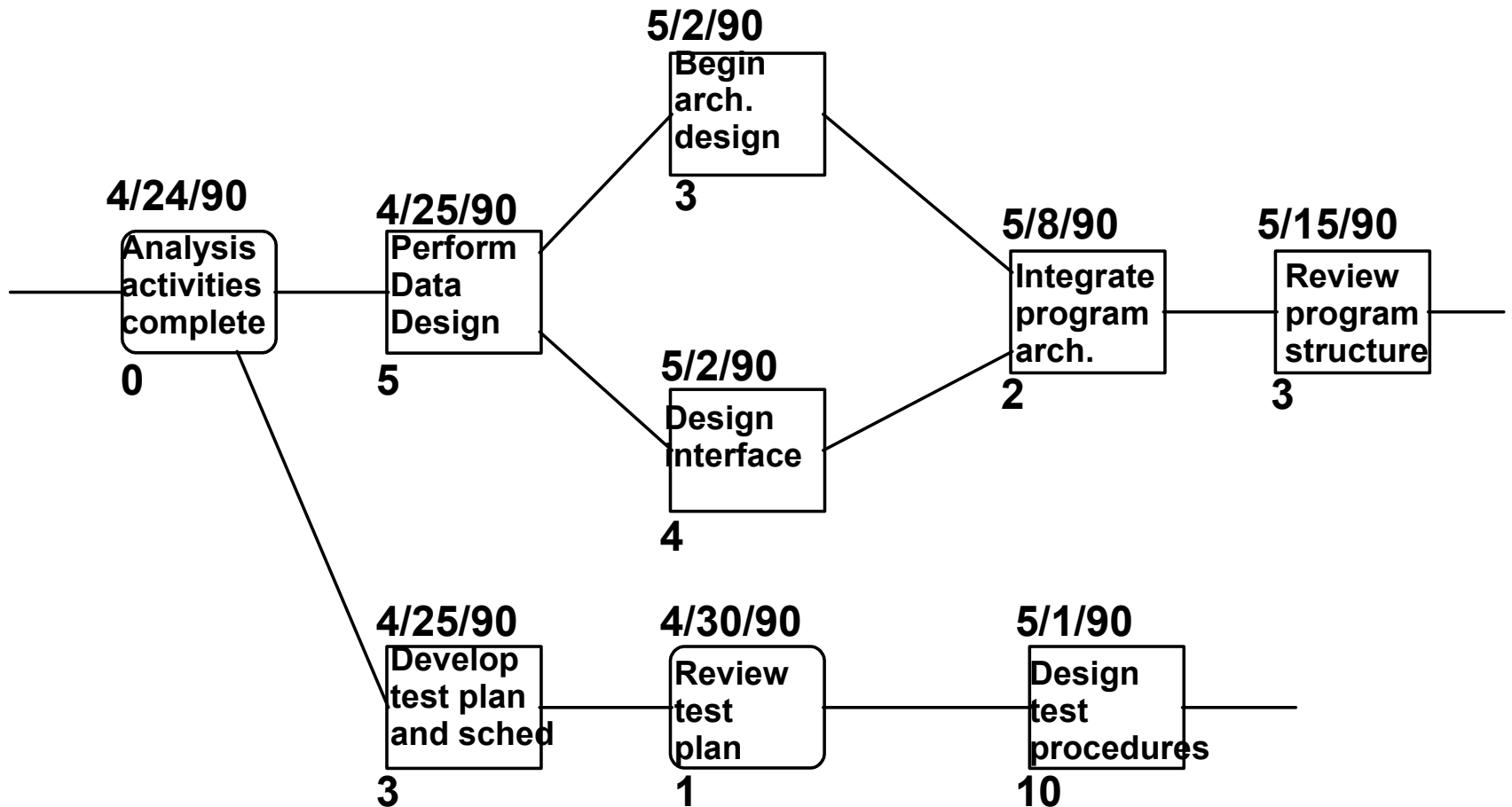
**Quantitative tools for the planner to determine:**

**Critical path**

**Most likely time estimates**

**Boundary times (earliest task start time, latest task start time, earliest task finish time, latest task finish time, total float time)**

# Typical Task Network





# **Approaches to Project Tracking**

**Conducting periodic project status meetings in which each team member reports progress and problems**

**Evaluating the results of all reviews conducted throughout the engineering process**

**Determining whether formal project milestones have been accomplished by the scheduled date**

**Comparing the actual start date to the planned start date for each task**

**Meeting informally with software engineers to obtain their subjective assessments of the progress to date and problems on the horizon**

# **Software Acquisition**

**Make or buy?**

**Who will use?**

**Buy and modify?**

**Contact outside contractor to build?**

**Decision based on:**

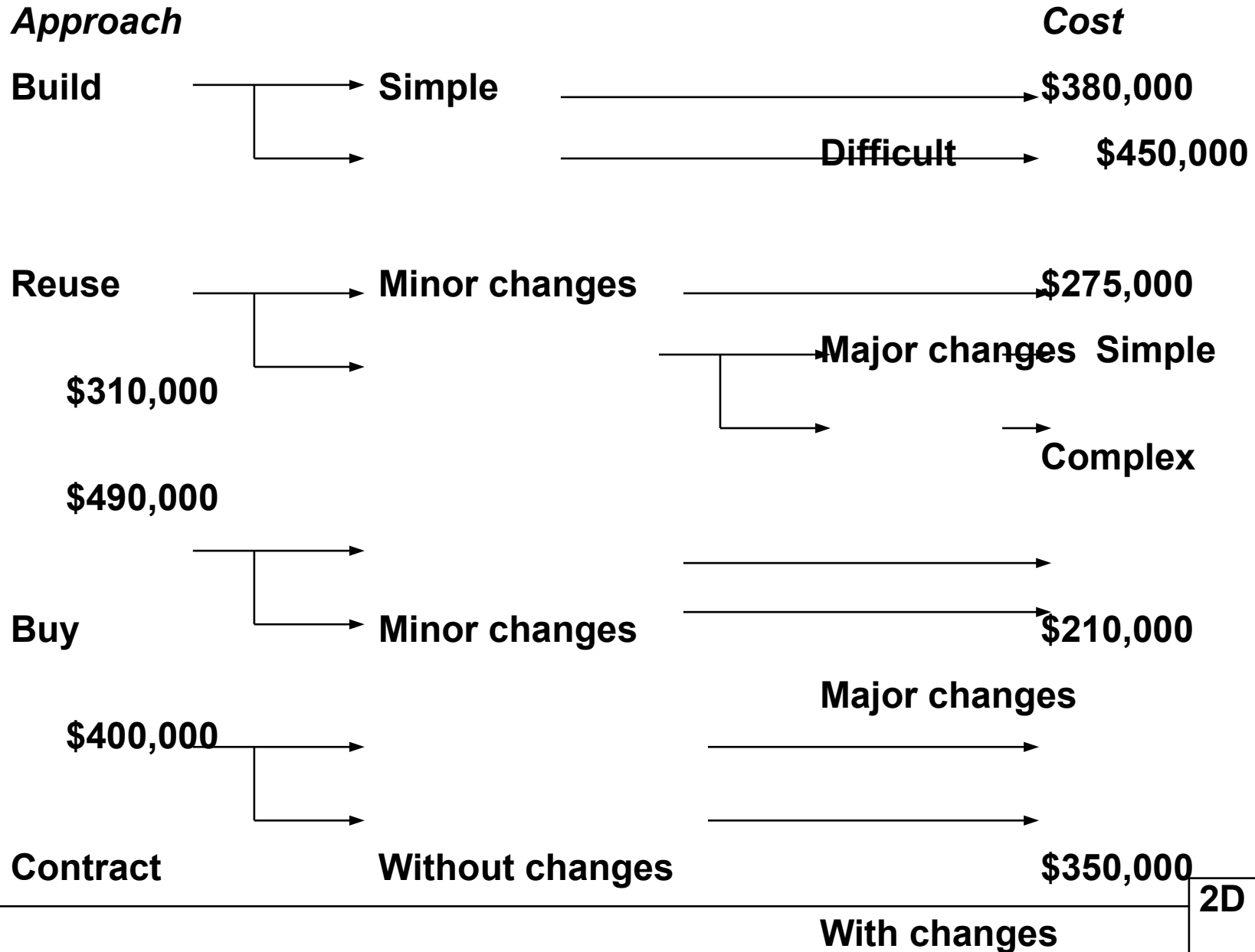
**Reduced cost**

**Earlier delivery date**

**Not enough or properly skilled people to develop**

**Better support outside**

# Software Acquisition Decision Tree



# **Software Re-Engineering**

**For often-used programs, build a controlled database of components for all to use.**

**Include documents, source code, user's guide, maintenance guide, test procedures and data, and a history of use with the components.**

**Software re-engineering may be enhanced by object-oriented design and implementation.**

# Organizational Planning

There are lots of human organizational structures for software development

Possibilities - consider N people working for K years on M different functional tasks

<i>Approach</i>		<i>Interaction</i>	<i>Level of Coordination</i>
1	Assign N people to M tasks	Individual ( $M > N$ )	Project Mgr
2	Assign N people to M tasks	Teams ( $M < N$ )	Project Mgr, Team Leader
3	Assign N people to T teams,	Formal	Project Mgr,
Teams	Team Leader	each team resp. for 1 or more tasks	

# **Enhancements to a Good Organization**

**The Chief Programmer Team**

**The Software Librarian**

**Egoless programming with a team environment**

# **The Software Project Plan (SPP)**

**A brief document which describes --**

**The scope of the project**

**The resources to be used**

**Risks and risk avoidance techniques**

**Cost and schedule**

**Overall approach to software development**

**Management, technical staff, and customer are the primary reads of the SPP.**

**The SPP provides a starting point for the rest of the project.**