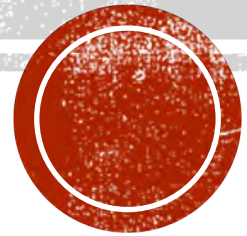
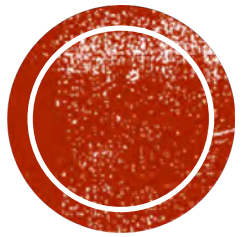
PROJECT MANAGEMENT

Software Engineering

Dr. Raj Singh





CONCEPTS

THE FOUR P'S

People — the most important element of a successful project

Product — the software to be built

Process — the set of framework activities and software engineering tasks to get the job done

Project — all work required to make the product a reality

STAKEHOLDERS

Senior managers	who define the business issues that often have significant influence on the project.
Project (technical) managers	who must plan, motivate, organize, and control the practitioners who do software work.
Practitioners	who deliver the technical skills that are necessary to engineer a product or application.
Customers	who specify the requirements and other stakeholders who have a peripheral interest in the outcome.
End-users	who interact with the software once it is released for production use.

SOFTWARE TEAMS

How to lead?

How to organize?

How to collaborate?

How to motivate?

How to create good ideas?

TEAM SELECTION CONSIDERATIONS

- the difficulty of the problem to be solved
- the size of the resultant program(s) in lines of code or function points
- the time that the team will stay together (team lifetime)
- the degree to which the problem can be modularized
- the required quality and reliability of the system to be built
- the rigidity of the delivery date
- the degree of sociability (communication) required for the project

AGILE TEAM PRINCIPLES

Team members must have trust in one another

The distribution of skills must be appropriate to the problem

Mavericks may have to be excluded from the team, if team cohesiveness is to be maintained

Team is self-organizing, adapting, and autonomous

THE PRODUCT SCOPE

Context	How does the software to be built fit into a larger system, product, or business context?
Constraints	What constraints are imposed as a result of the context?
Objectives	What value does it provide to end user?
Function and performance	What function does the software perform? Are any special performance characteristics to be addressed?
Clear understanding	Scope must be unambiguous and understandable by all stakeholders.

PROBLEM DECOMPOSITION

Sometimes called partitioning or problem elaboration

Once scope is defined decompose into a set of problem classes

Decomposition process continues until all functions or problem classes have been defined

WHY PROJECTS FAIL?

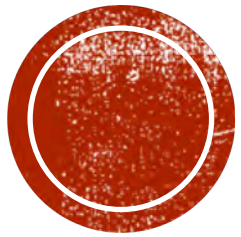
- Customer needs are not understood.
- Incomplete and unrealistic requirements and timelines.
- Scope is poorly defined.
- Changes are managed poorly.
- Resource, budget, technology, and priority changes.
- The project team lacks people with appropriate skills.
- Practitioners avoid best practices and lessons learned.

COMMON-SENSE APPROACH TO PROJECTS

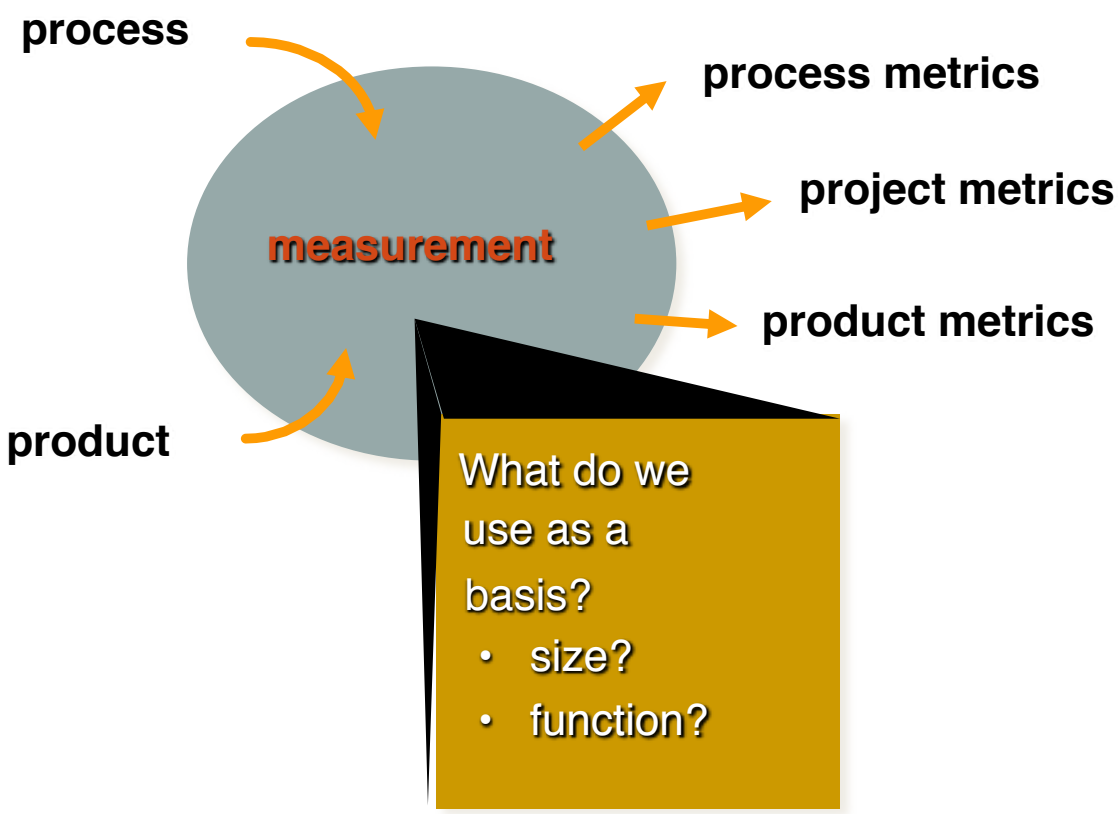
- Start on the right foot.
- Maintain momentum.
- Track progress.
- Make smart decisions.
- Conduct a postmortem analysis.

CRITICAL PRACTICES

- Formal risk management
- Empirical cost and schedule estimation
- Metrics-based project management
- Earned value tracking
- Defect tracking against quality targets
- People aware project management



PROCESS AND PROJECT METRICS

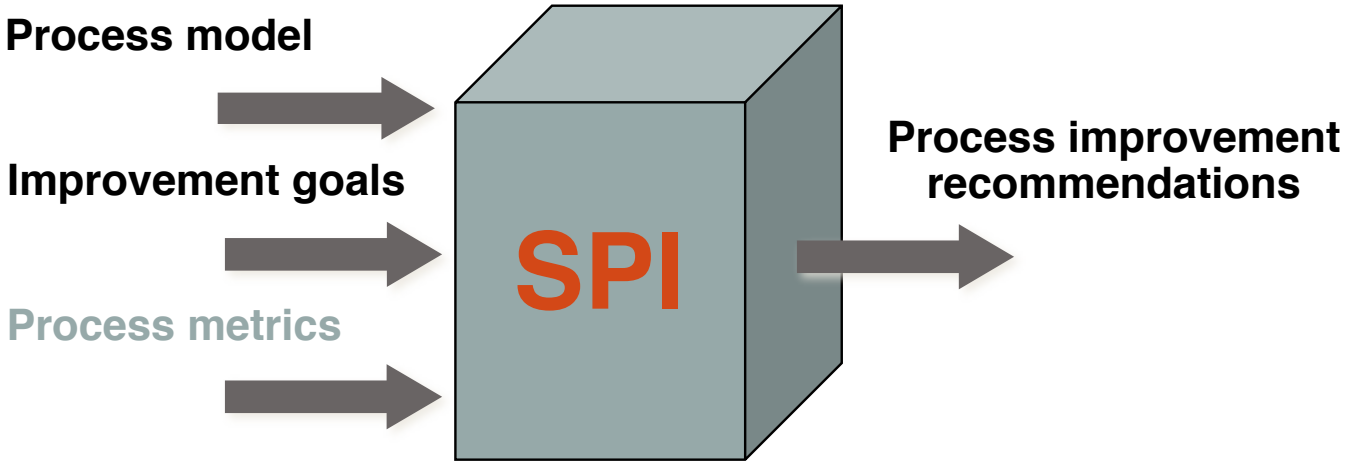


A GOOD MANAGER MEASURES

WHY DO WE MEASURE?

- assess the status of an ongoing project
- track potential risks
- uncover problem areas before they go “critical”
- adjust work flow or tasks
- evaluate the project team’s ability to control quality of software work products.

SOFTWARE PROCESS IMPROVEMENT



PROCESS METRICS

Quality-related	focus on quality of work products and deliverables
Productivity-related	Production of work-products related to effort expended
Statistical data	error categorization & analysis
Defect removal efficiency	propagation of errors from process activity to activity
Reuse data	The number of components produced and their degree of reusability

PROJECT METRICS

Effort/time per software engineering task

Errors uncovered per review hour

Scheduled vs. actual milestone dates

Changes (number) and their characteristics

Distribution of effort on software engineering tasks

METRICS GUIDELINES

- Use common sense and organizational sensitivity when interpreting metrics data.
- Provide regular feedback to the individuals and teams who have worked to collect measures and metrics.
- Don’t use metrics to appraise individuals.
- Work with practitioners and teams to set clear goals and metrics that will be used to achieve them.
- Never use metrics to threaten individuals or teams.
- Metrics data that indicate a problem area should not be considered “negative.” These data are merely an indicator for process improvement.
- Don’t obsess on a single metric to the exclusion of other important metrics.

MEASURING QUALITY

Correctness	the degree to which a program operates according to specification
Maintainability	the degree to which a program is amenable to change
Integrity	the degree to which a program is impervious to outside attack
Usability	the degree to which a program is easy to use

ESTABLISHING A METRICS PROGRAM

Identify

business goals, entities, attributes, data, actions

Formalize

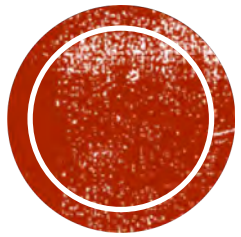
your measurement goals

Define

The measures to be used, and make these definitions operational

Prepare

a plan for implementing the measures



ESTIMATION

PROJECT PLANNING TASK SET

ESTABLISH PROJECT SCOPE

DETERMINE FEASIBILITY

ANALYZE RISKS

DEFINE REQUIRED RESOURCES

ESTIMATE COST AND EFFORT

DEVELOP A PROJECT SCHEDULE

ESTIMATION

Estimation of resources, cost, and schedule for a software engineering effort requires

experience	historical information	commitment

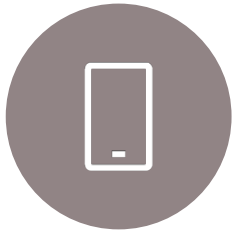
Estimation carries inherent risk and this risk leads to uncertainty

TO UNDERSTAND SCOPE …

UNDERSTAND THE CUSTOMERS NEEDS

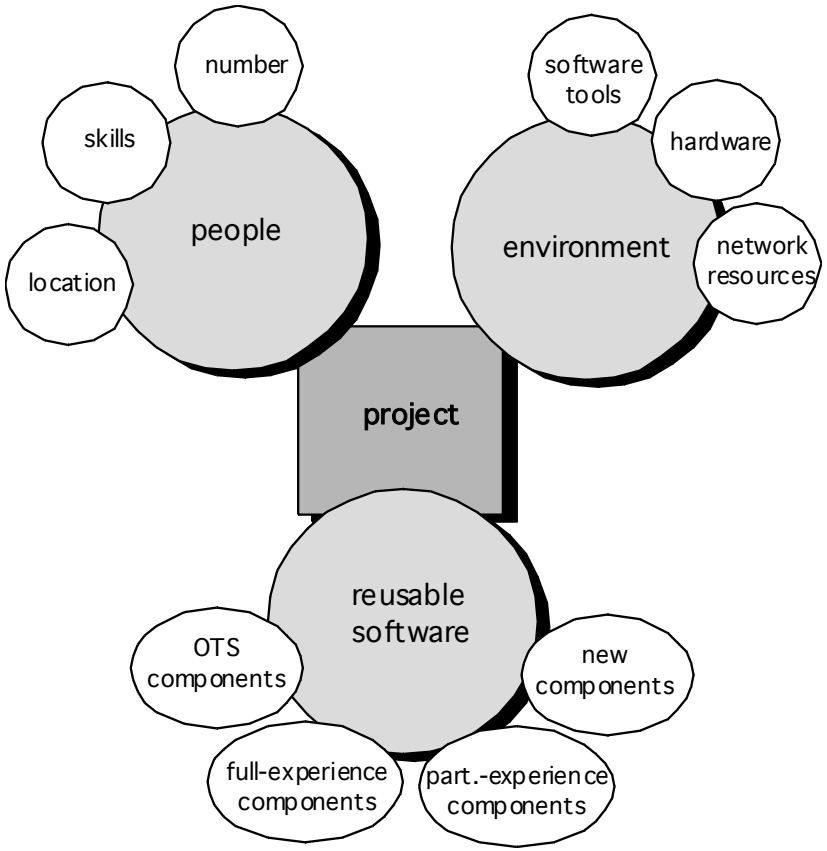
UNDERSTAND THE BUSINESS CONTEXT

UNDERSTAND THE PROJECT BOUNDARIES



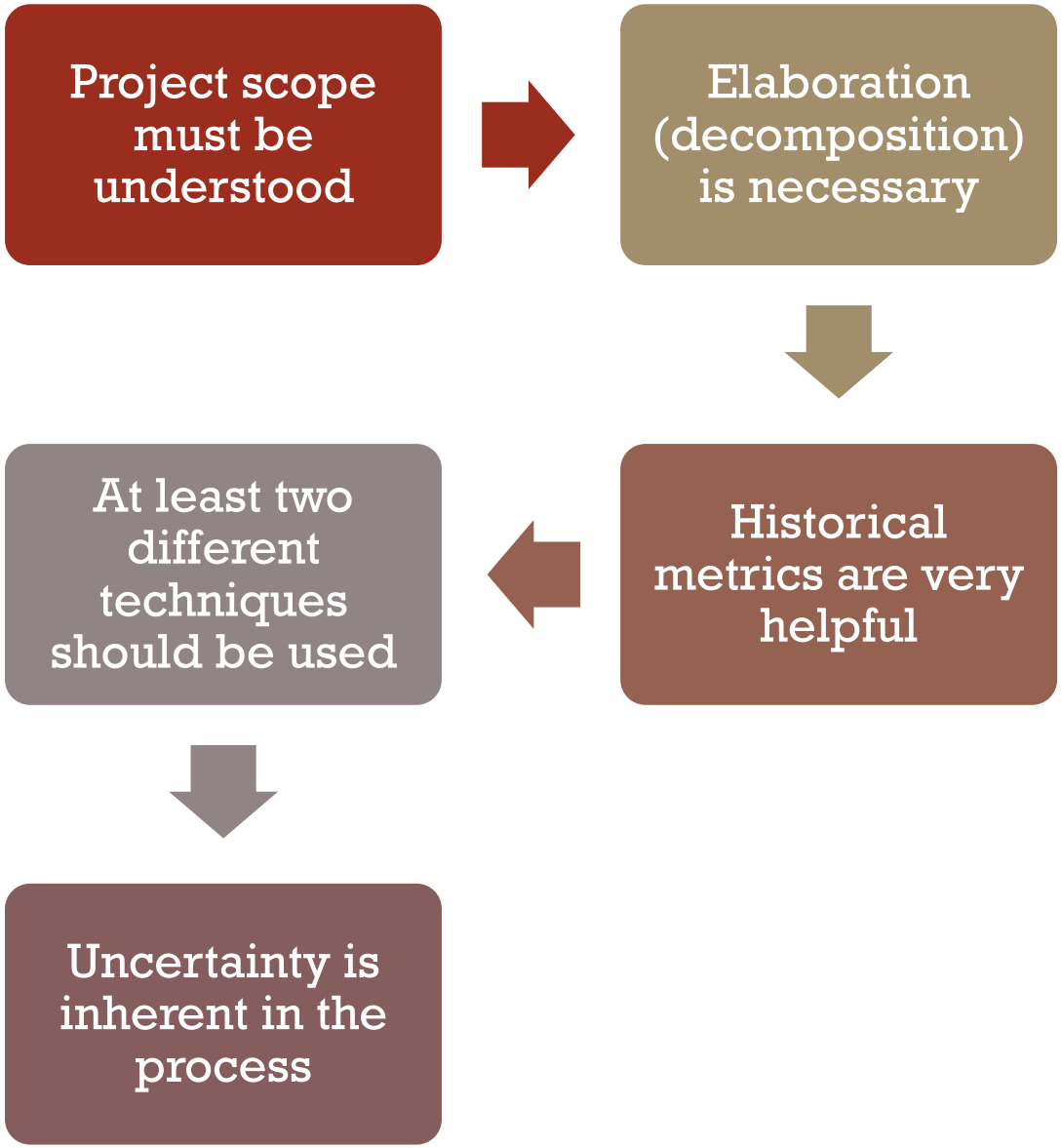
UNDERSTAND THE CUSTOMER'S MOTIVATION

UNDERSTAND THE LIKELY PATHS FOR CHANGE


number
skills
people
location
software tools
hardware
environment
network resources
project
reusable software
OTS components
new components
full-experience components
part.-experience components

RESOURCES

PROJECT ESTIMATION



ESTIMATION TECHNIQUES

Past (similar) project experience

Conventional estimation techniques

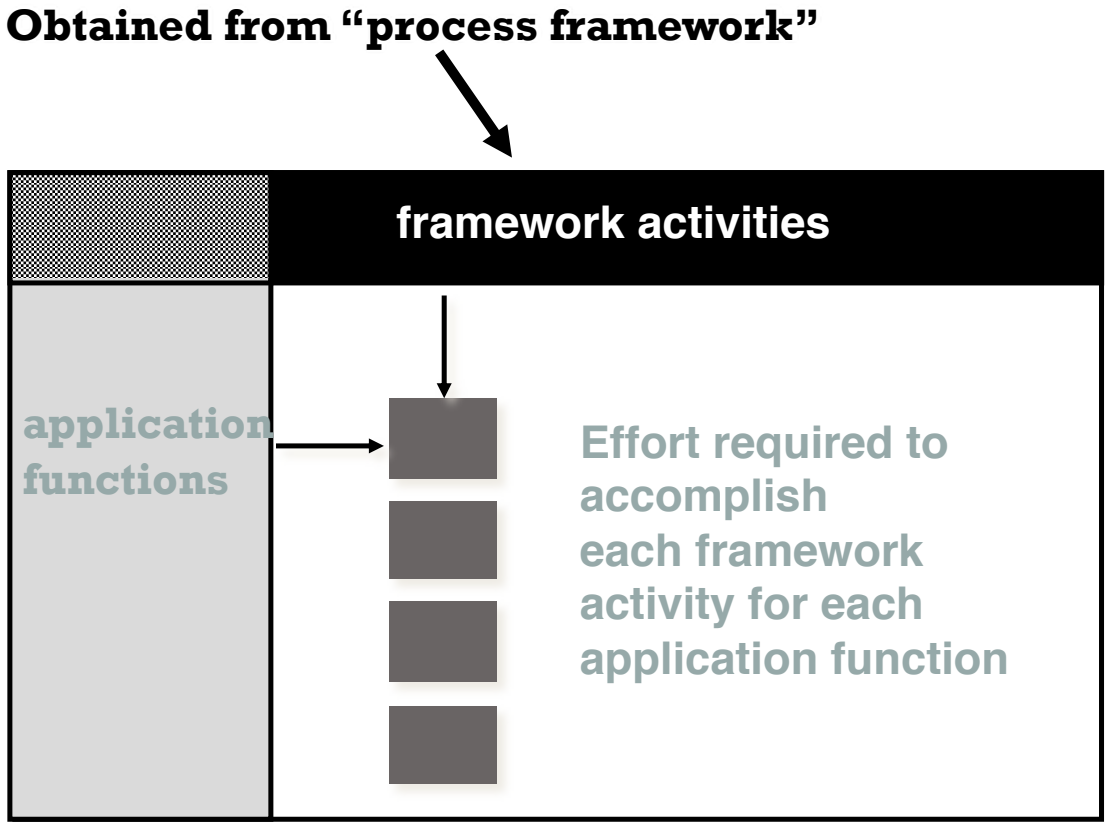
task breakdown and effort estimates

size (e.g., FP) estimates

Empirical models

Automated tools

PROCESS-BASED ESTIMATION



PROCESS-BASED ESTIMATION EXAMPLE

Activity →	CC	Planning	Risk Analysis	Engineering		Construction Release		CE	Totals
Task →				analysis	design	code	test		
Function									
▼									
UICF				0.50	2.50	0.40	5.00	n/a	8.40
2DGA				0.75	4.00	0.60	2.00	n/a	7.35
3DGA				0.50	4.00	1.00	3.00	n/a	8.50
CGDF				0.50	3.00	1.00	1.50	n/a	6.00
DSM				0.50	3.00	0.75	1.50	n/a	5.75
PCF				0.25	2.00	0.50	1.50	n/a	4.25
DAM				0.50	2.00	0.50	2.00	n/a	5.00
Totals	0.25	0.25	0.25	3.50	20.50	4.50	16.50		46.00
% effort	1%	1%	1%	8%	45%	10%	36%		

CC = customer communication CE = customer evaluation

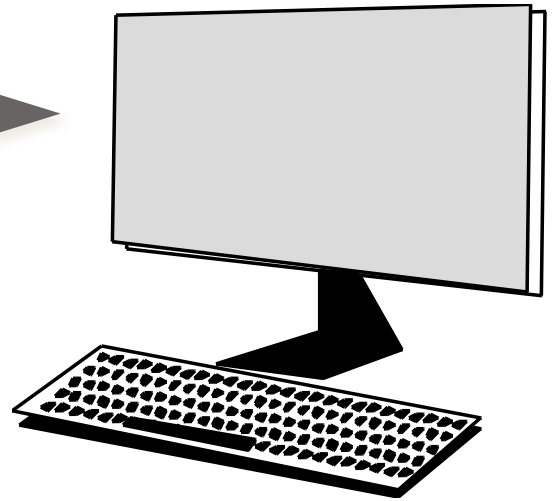
- Based on an average burdened labor rate of $8,000 per month, **the total estimated project cost is $368,000 and the estimated effort is 46 person-months.**

TOOL-BASED ESTIMATION

project characteristics

calibration factors

LOC/FP data



ESTIMATION FOR AGILE PROJECTS

Each user scenario is considered separately for estimation purposes.

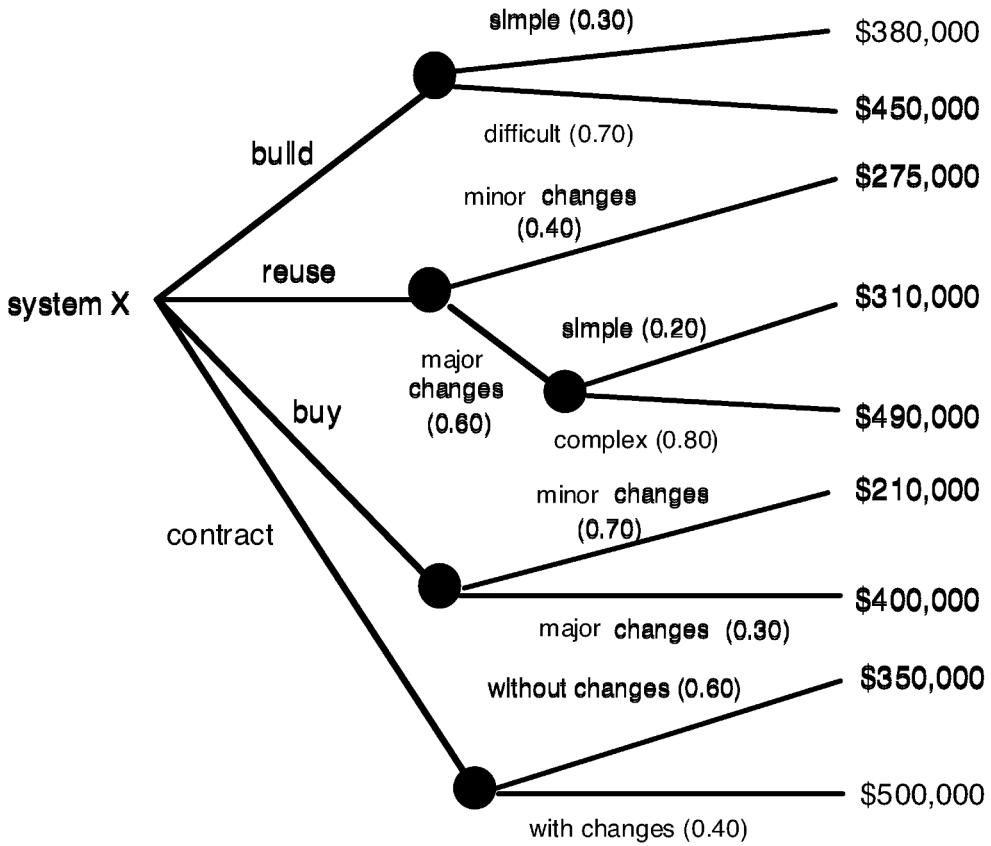
The scenario is decomposed into the set of software engineering tasks.

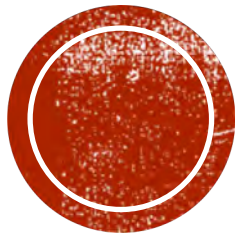
Each task is estimated separately.

Estimates for each task are summed to create an estimate for the scenario.

The effort estimates for all scenarios are summed to develop the effort estimate for the increment.

THE MAKE-BUY DECISION





PROJECT SCHEDULING

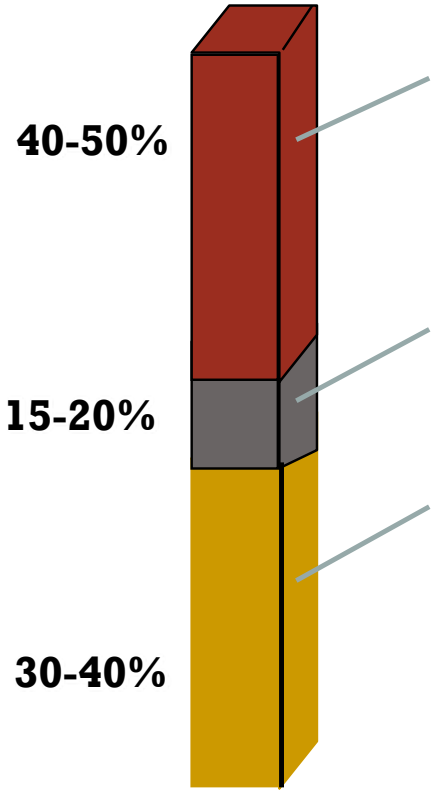
WHY ARE PROJECTS LATE?

- an unrealistic deadline
- changing customer requirements
- underestimate of the effort and resources
- unpredictable risks
- technical difficulties
- miscommunication
- a lack of action to correct the problem

SCHEDULING PRINCIPLES

Compartmentalization	define distinct tasks
Interdependency	indicate task interrelationship
Effort validation	be sure resources are available
Defined responsibilities	people must be assigned
Defined outcomes	each task must have an output
Defined milestones	review for quality

EFFORT ALLOCATION



- "front end" activities
  - customer communication
  - analysis
  - design
  - review and modification
- construction activities
  - coding or code generation
- testing and installation
  - unit, integration
  - white-box, black box
  - regression

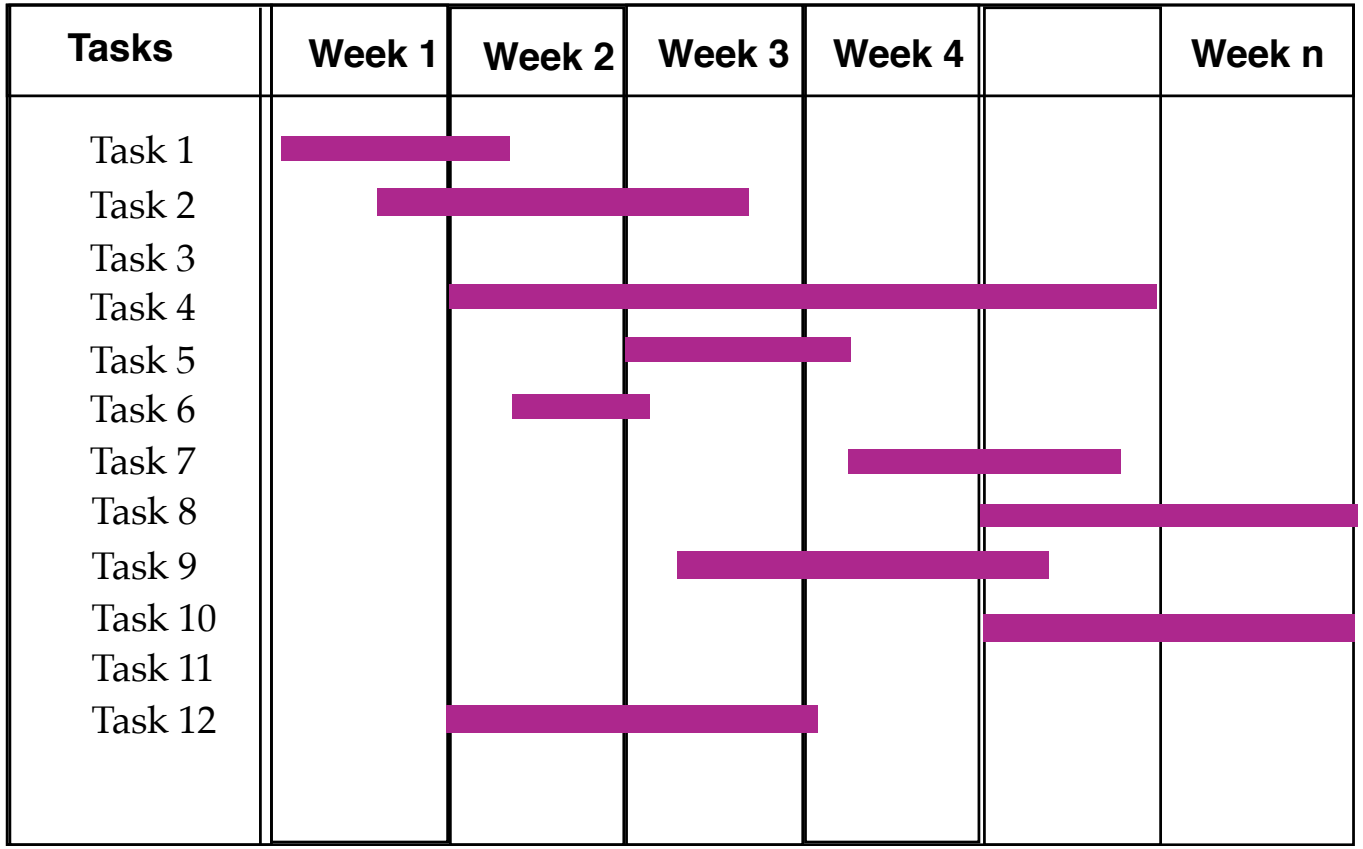
DEFINING TASK SETS

determine type of project

assess the degree of rigor required

identify adaptation criteria

select appropriate software engineering tasks



TIMELINE CHARTS

REFERENCE

- Roger Pressman, Software Engineering: A Practitioner's Approach, 8th edition, McGraw Hill, ISBN 0078022126