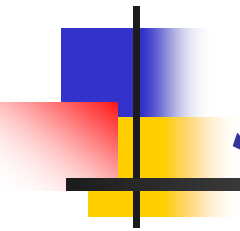
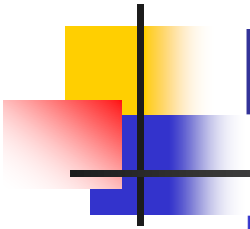


COMP 354: Introduction to Software Engineering

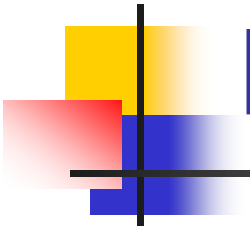
Viable Software Plan

Based on Chapter 25 of the textbook



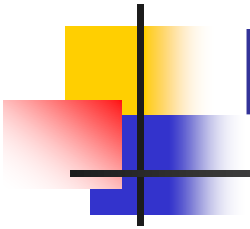
Estimation Issues

- Estimation of resources, cost, and schedule for a software engineering effort requires:
  - Experience.
  - Access to good historical information (metrics).
  - The courage to commit to quantitative predictions when qualitative information is all that exists.
- Estimation carries inherent risk and this risk leads to uncertainty:
  - Project complexity.
  - Project size (makes decomposition tougher).
  - Degree of structural uncertainty (requirements stability).



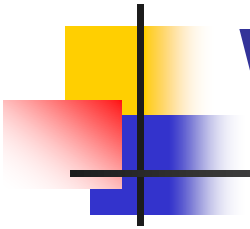
Project Planning Task Set

1. Establish project scope.
2. Determine feasibility.
3. Analyze risks (Chapter 26).
4. Define required resources.
   a. Determine required human resources.
   b. Define reusable software resources.
   c. Identify environmental resources.
5. Estimate cost and effort.
   a. Decompose the problem.
   b. Develop two or more estimates.
   c. Reconcile the estimates.



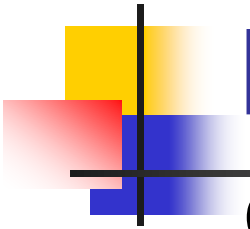
Project Planning Task Set

6. Develop an initial project schedule.
   a. Establish a meaningful task set.
   b. Define a task network.
   c. Use scheduling tools to develop a time-line chart.
   d. Define schedule tracking mechanisms.
7. Repeat steps 1 to 6 to create a detailed schedule for each prototype as the scope of each prototype is defined.



What is Scope?

- Software scope describes
  - Functions and features to be delivered to end-users.
  - Data input and output.
  - Content presented to users of using the software.
  - Performance, constraints, interfaces, and reliability that bound the system.
- Scope is defined using one of two techniques:
  - A narrative description of software scope is developed after communication with all stakeholders.
  - A set of use-cases is developed by end-users.

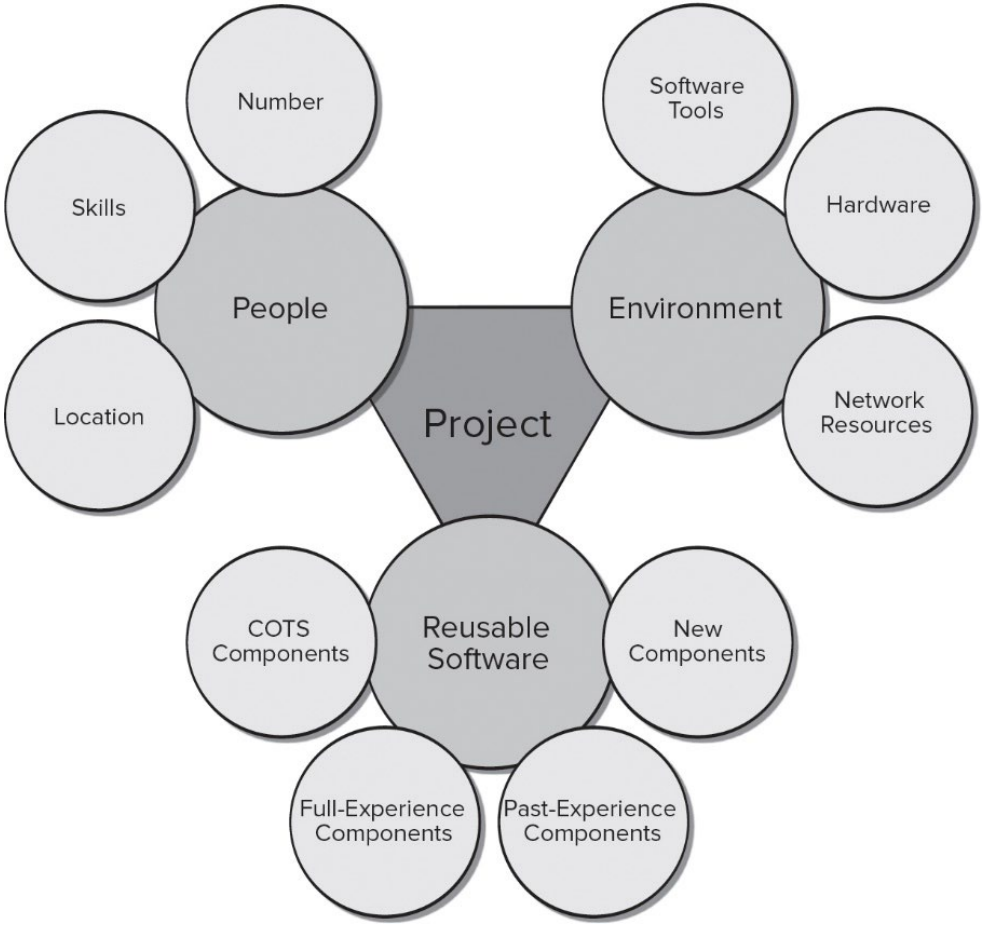


Project Feasibility

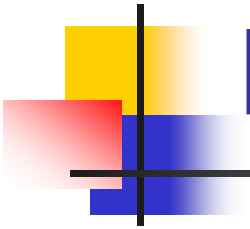
Once scope has been identified (with the concurrence of the customer), it is reasonable to ask:

- Can we build software to meet this scope?
- Is the project feasible?
- You must try to determine if the system can be created using available technology, dollars, time, and other resources.
- Consideration of business need is important too - it does no good to build a high-tech system or product that no one wants.

Project Resources

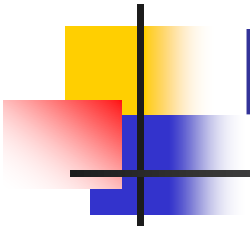


Data Analytics and Estimation Accuracy



To achieve reliable cost and effort estimates several options arise:

- Delay estimation until late in the project (we can achieve 100 percent accurate estimates after the project is complete!).
- Base estimates on similar projects that have already been completed (works great if you have completed similar projects).
- Use relatively simple decomposition techniques to generate project cost and effort estimates (similar to divide and conquer).
- Use one or more empirical models for software cost and effort estimation (often derived using statistical regression models).

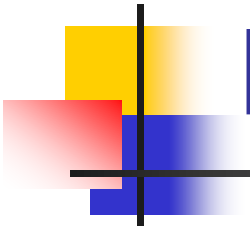


Reconciling Estimates

- Any estimation technique must be checked by computing at least one other estimate using a different approach.
- If you have created multiple estimates they need to be compared and reconciled.
- If both estimates show agreement, there is good reason to believe that the estimates are reliable.
- Widely divergent estimates can often be traced to one of two causes:
  1. The scope of the project is not adequately understood or has been misinterpreted by the planner.
  2. Productivity data used for problem-based estimation techniques is inappropriate for the application or has been misapplied.

Problem-Based Estimation

- LOC and FP data are used in two ways during software project estimation:
  1. as estimation variables to "size" each element of the software
  2. as baseline metrics collected from past projects and used with other variable to develop cost and effort projections.
- When collecting productivity metrics for projects, be sure to establish a taxonomy of project types.
- Be sure that your estimates include the effort required to develop "infrastructure" software.

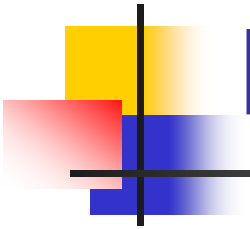


Problem-Based Estimation

- Begin with a bounded statement of software scope.
- Decompose the statement of scope into problem functions that can each be estimated individually.
- LOC or FP is then estimated for each function.
- Baseline productivity metrics (For example, LOC/pm or FP/pm) are then applied to the appropriate estimation variable.
- Cost/effort for the function is derived using historic data.
- Function estimates are combined to produce an overall estimate for the entire project.

LOC-Based Estimation Table

Function	Estimated LOC
User-interface and control facilities (UICF)	2,300
Two-Dimensional geometric analysis (2DGA)	5,300
Three-Dimensional geometric analysis (3DGA)	6,800
Database management (DBM)	3,350
Computer graphics display facilities (GCDF)	4,950
Peripheral control function (PCF)	2,100
Design analysis modules (DAM)	8,400
Estimated lines of code	33,200



LOC-Based Estimation

- Average productivity for these systems is 620 LOC/pm.
- Burdened labor rate is $8000 per month.
- Cost per line of code is approximately $13.
- Based on LOC estimates and historical data:

  estimated project cost is $431,000

  estimated effort is 54 person-months

FP-Based Estimation Table

Information domain value	Opt.	Likely	Pess.	Est count.	Weight	F P count
Number of external inputs	20	24	30	24	4	96 (24 × 4 = 96)
Number of external outputs	12	14	22	14	5	70 (14 × 5 = 70)
Number of external enquiries	16	20	28	20	5	100 (20 × 5 = 100)
Number of internal logical files	4	4	5	4	10	40 (4 × 10 = 40)
Number of external interface files	2	2	3	2	7	14 (2 × 7 = 14)
Count Total						320

Table 25.1 Estimating information domain values

FP-Based Estimation

- To compute the FP equation:

$$\mathrm{FP}_{\text{estimated}} = \text{count total} \times \left[0.65 + 0.01 \times \sum (Fi)\right]$$

- For the purposes of this estimate, the complexity weighting factor is assumed to be average and the FP count total from the table is 320.
- Assume the sum of the 14 complexity factors $\sum (Fi)$ is 52.

$$\left[0.65 + 0.01 \times \sum (Fi)\right] = 1.17$$

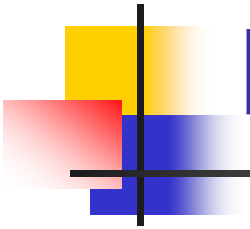
- The estimated number of FP can be computed:

$$\mathrm{Fp}_{\text{estimated}} = \text{count total} \times \left[0.65 + 0.01 \times \sum Fi\right] = 375$$

- If the historic cost per FP is approximately $1,230 then total estimated project cost is $461,000 estimated effort is 58 person-months.

Process-Based Estimation Table

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
DBM				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%		



Process-Based Estimation

- Process-based estimation begins with a delineation of software functions obtained from the project scope.
- A series of framework activities are performed for each function.
- Functions and related framework activities may be represented as part of a table with tasks as columns and rows as functions.
- The effort estimates (for example, person-months) are entered as the matrix cell values.
- Average labor rates (that is, cost/unit effort) are then applied to the effort estimated for each process activity.
- 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 based on the matric entries.

Use Case Point Estimation

Computation of use case points takes the following into account:

- The number and complexity of the use cases in the system.
- The number and complexity of the actors on the system.
- Various nonfunctional requirements not written as use cases.
- The environment in which the project will be developed.

$$UCP = (UUCW + UAW) \times TCF \times ECF$$

UUCW – unadjusted sum of use case weights
UAW – unadjusted sum of actor weight
TCF – technical complexity 13 factors
ECF – environment complexity 8 factors

Use Case Point Estimation Example

The engineering subsystem group is described by 14 average use cases and 8 simple use cases. And the infrastructure subsystem is described with 10 simple use cases.

$$UUCW = (16 \text{ use cases} \times 15) + [(14 \text{ use cases} \times 10) + (8 \text{ use cases} \times 5)] + (10 \text{ use cases} \times 5) = 470$$

There are 8 simple actors, 12 average actors, and 4 complex actors.

$$UAW = (8 \text{ actors} \times 1) + (12 \text{ actors} \times 2) + (4 \text{ actors} \times 3) = 44$$

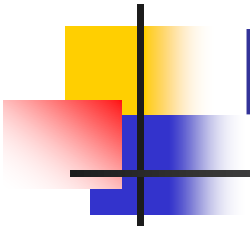
After evaluation of the technology and the environment,

$$TCF = 1.04$$

$$ECF = 0.96$$

$$UCP = (470 + 44) \times 1.04 \times 0.96 = 513$$

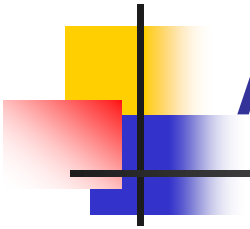
Use Case Point Estimation Example



- Using past project data as a guide, the development group produces 85 LOC per UCP.
- An estimate of the overall size of the CAD project is 43,600 LOC.
- Using 620 LOC/pm as the average productivity for systems of this type and a burdened labor rate of $8,000 per month, and the cost per line of code is approximately $13.
- Based on the use case estimate and the historical productivity data:

  Total estimated project cost is $552,000.

  Estimated effort is about 70 person-months.



Agile Project Estimation

1. Each user story is considered separately for estimation purposes.
2. Each user story is decomposed into the set of software engineering tasks that will be required to develop it.

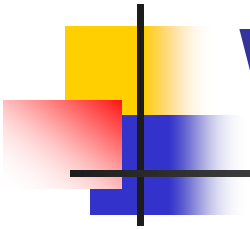
3a. Each task is estimated separately (historic data, empirical model, experience, or planning poker.

3b. Alternatively, the "volume" of the user story can be estimated in LOC, FP, or use case count.

4a. Estimates for each task are summed to estimate the user story.

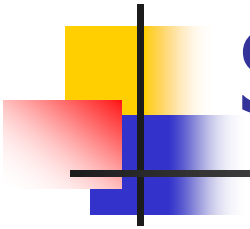
4b. Alternatively, the volume translated into effort using historical data.

5. Effort estimates for all user stories are summed to create effort estimate for the increment.



Why Are Projects Late?

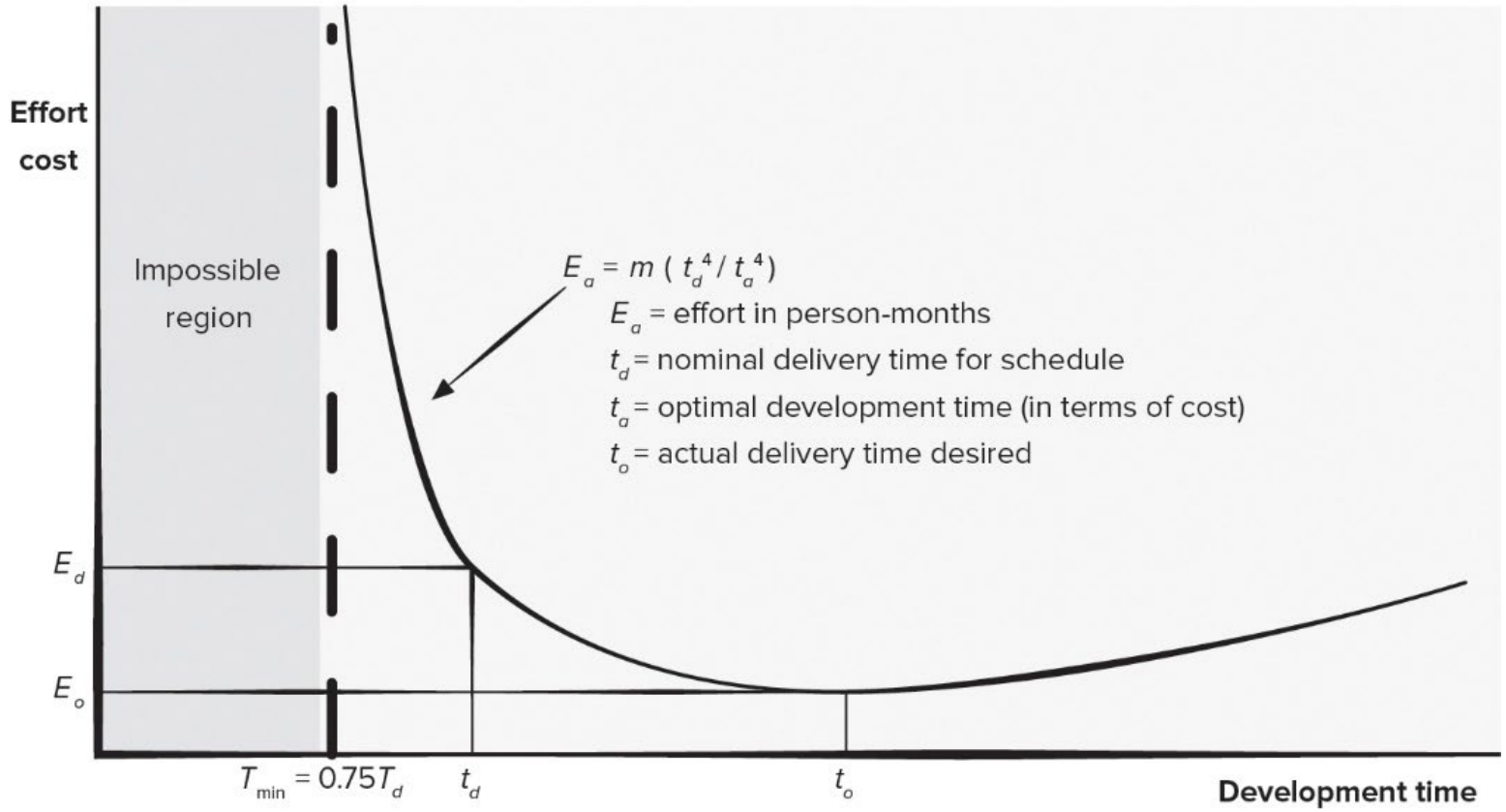
- Unrealistic deadline established by someone outside the software team and forced on managers and practitioners on the group.
- Changing requirements not reflected in schedule changes.
- An honest underestimate of the amount of effort and/or the number of resources that will be required to do the job.
- Predictable and/or unpredictable risks that were not considered when the project commenced.
- Technical difficulties that could not have been foreseen in advance.
- Human difficulties that could not have been foreseen in advance.
- Miscommunication among project staff that results in delays.
- Failure by project management to recognize that the project is falling behind schedule and lack of action to correct the problem.



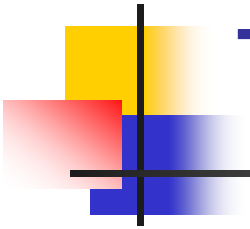
Scheduling Principles

- Compartmentalization. The project must be compartmentalized by decomposing the product and the process.
- Interdependency. The interdependency of each compartmentalized activity or task must be determined.
- Time allocation. Each task must be allocated some number of work units and assigned a start date and a completion date.
- Effort validation. Ensure that no more than the allocated number oFPeople has been scheduled at any given time.
- Defined responsibilities. Every task that is scheduled should be assigned to a specific team member.
- Defined outcomes. Every task should have a defined outcome.
- Defined milestones. Every task should be associated with a project milestone.

Relationship Between Effort and Delivery Date



Concept Development Task Set



1.1 Concept scoping determines the overall scope of the project.

1.2 Preliminary concept planning establishes the organization's ability to undertake the work implied by the project scope.

1.3 Technology risk assessment evaluates the risk associated with the technology to be implemented as part of the project scope.

1.4 Proof of concept demonstrates the viability of a new technology in the software context.

1.5 Concept implementation implements the concept representation in a manner that can be reviewed by a customer and is used for "marketing" purposes when a concept must be sold to other customers or management.

1.6 Customer reaction to the concept solicits feedback on a new technology concept and targets specific customer applications.

Task 1.1 Refinement

Task definition: Task 1.1 Concept Scoping

1.1.1 Identify need, benefits and potential customers;

1.1.2 Define desired output/control and input events that drive the application;

Begin Task 1.1.2

1.1.2.1 FTR: Review written description of need indicates that a FTR is to be conducted.

1.1.2.2 Derive a list of customer visible outputs/inputs

1.1.2.3 FTR: Review outputs/inputs with customer and revise as required;

endtask Task 1.1.2

1.1.3 Define the functionality/behavior for each major function;

Begin Task 1.1.3

1.1.3.1 FTR: Review output and input data objects derived in task 1.1.2;

1.1.3.2 Derive a model of functions/behaviors;

1.1.3.3 FTR: Review functions/behaviors with customer and revise as required;

endtask Task 1.1.3

1.1.4 Isolate those elements of the technology to be implemented in software;

1.1.5 Research availability of existing software;

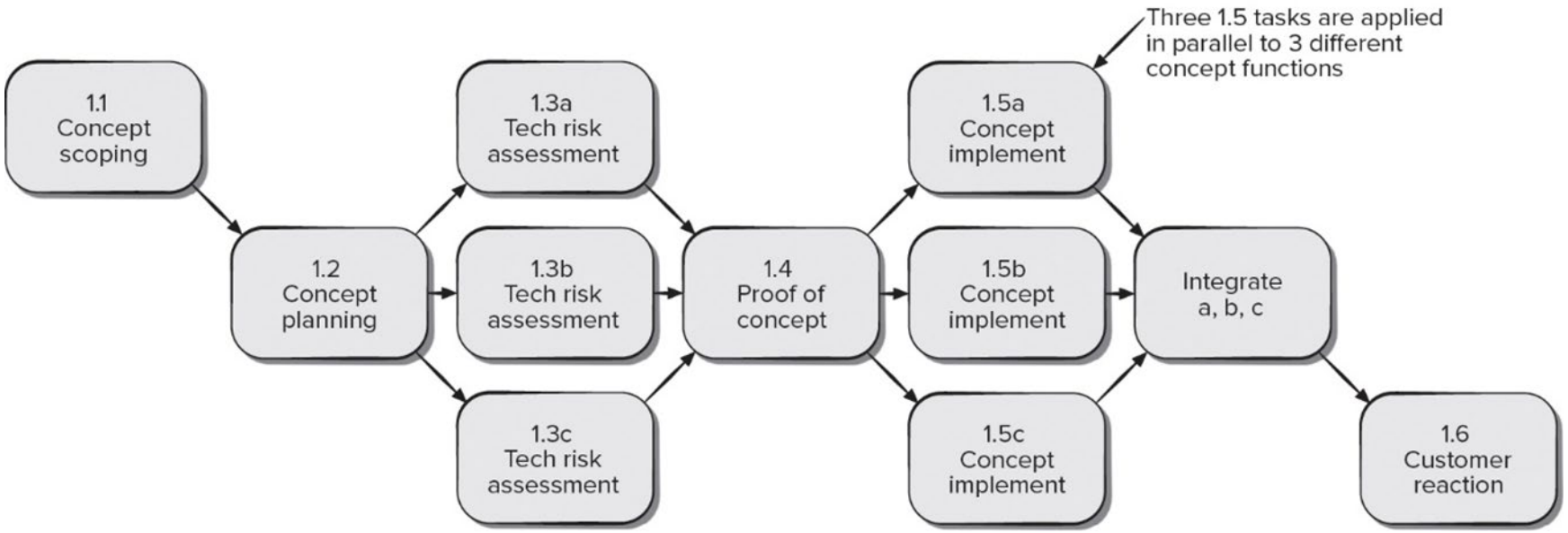
1.1.6 Define technical feasibility;

1.1.7 Make quick estimate of size;

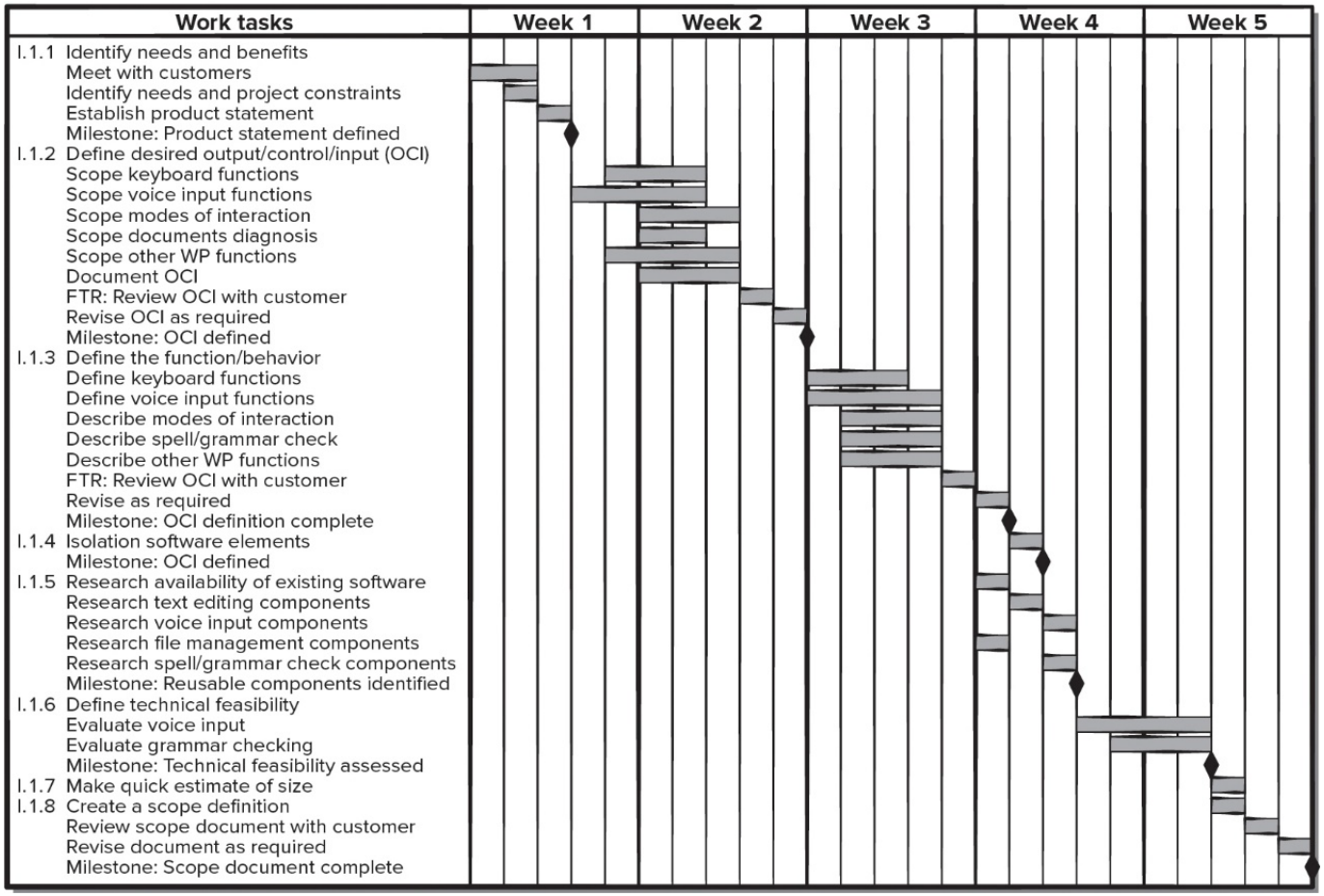
1.1.8 Create a Scope Definition;

endTask definition: Task 1.1

Task Network (Activity Network)



Timeline Chart (Gantt Chart)



Project Table for Project Tracking

Work tasks	Planned start	Actual start	Planned complete	Actual complete	Assigned person	Effort allocated	Notes
I.1.1 Identify needs and benefits							Scoping will require more effort/time
Meet with customers	wk1, d1	wk1, d1	wk1, d2	wk1, d2	BLS	2 p-d	
Identify needs and project constraints	wk1, d2	wk1, d2	wk1, d2	wk1, d2	JPP	1 p-d	
Establish product statement	wk1, d3	wk1, d3	wk1, d3	wk1, d3	BLS/	1 p-d	
Milestone: Product statement defined	wk1, d3	wk1, d3	wk1, d3	wk1, d3			
I.1.2 Define desired output/control/input (OCI)							
Scope keyboard functions	wk1, d4	wk1, d4	wk2, d2		BLS	1.5 p-d	
Scope voice input functions	wk1, d3	wk1, d3	wk2, d2		JPP	2 p-d	
Scope modes of interaction	wk2, d1		wk2, d3		MLL	1 p-d	
Scope documents diagnosis	wk2, d1		wk2, d2		BLS	1.5 pd	
Scope other WP functions	wk1, d4	wk1, d4	wk2, d3		JPP	2 p-d	
Document OCI	wk2, d1		wk2, d3		MLL	3 p-d	
FTR: Review OCI with customer	wk2, d3		wk2, d3		all	3 p-d	
Revise OCI as required	wk2, d4		wk2, d4		all	3 p-d	
Milestone: OCI defined	wk2, d5		wk2, d5				
I.1.3 Define the Function/behavior							

Schedule Tracking

- Conducting periodic project status meetings in which each team member reports progress and problems.
- Evaluating the results of all reviews conducted throughout the software 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 project task listed in the project resource table.
- Meeting informally with practitioners to obtain their subjective assessment of progress to date and problems on the horizon.
- Tracking the project velocity, which is a way of seeing how quickly the development team is clearing the user story backlog.