
Scheduling Principles

- Compartmentalization—define distinct tasks
- Interdependency—indicate task interrelationships, 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

Task Set

A set of tasks that enable the software team to define, develop, and ultimately maintain software.

- No single set of tasks is appropriate for all projects
- An effective software process should define a collection of task sets designed to meet the needs of different types of projects.

The task set is a collection of software engineering work tasks, milestones, and deliverables that must be accomplished to complete a particular project.

Defining Task Sets

Determine type of project

Assess the degree of rigor required

- identify adaptation criteria
- compute task set selector (TSS) value
- interpret TSS to determine degree of rigor

Select appropriate software engineering tasks

Five common types of task sets

- **Concept Development Projects**-- initiated to explore some new business concept or application of some new technology
- **New Application Development Projects**-- undertaken as a consequence of a specific customer request
- **Application Enhancement Projects**-- occur when existing software undergoes major modifications to function, performance, or interfaces that are observable by the end user
- **Application Maintenance Projects**-- correct, adapt, or extend existing software in ways that may not be immediately obvious to the end user
- **Reengineering Projects**-- undertaken with the intent of rebuilding an existing (legacy) system in whole or in part

Example

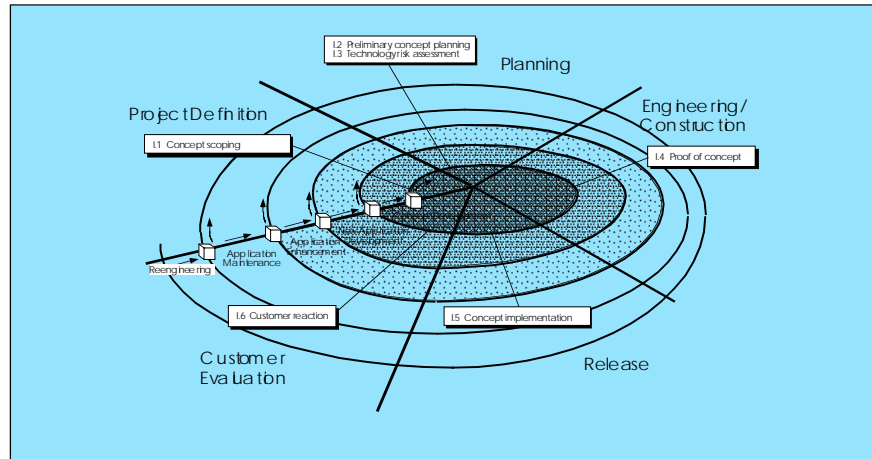
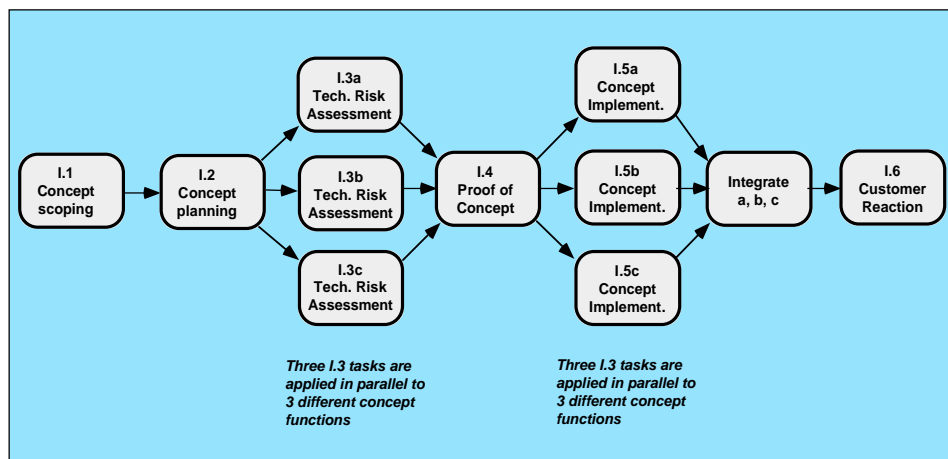


Figure 7.2 Concept development tasks using an evolutionary model

SOE2000

7

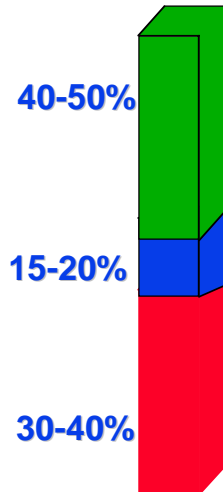
Define a Task Network



SOE2000

8

Effort Allocation



“front end” activities

- customer communication
- analysis
- design
- review and modification

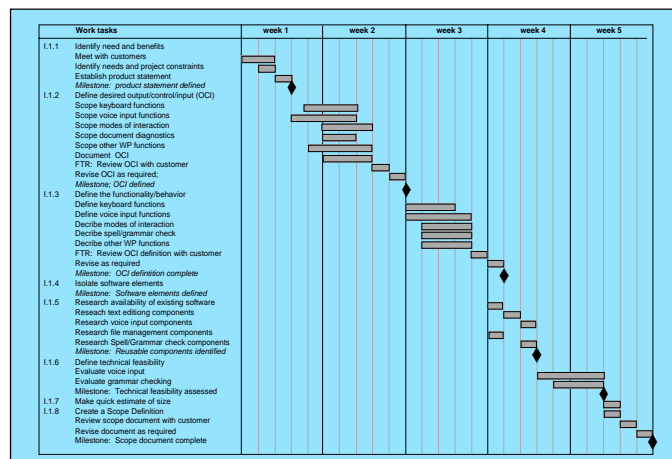
construction activities

- coding or code generation

testing and installation

- unit, integration
- white-box, black box
- regression

Use Automated Tools to Derive a Timeline Chart



Scheduling Methods: Strengths

- These methods are continuously useful to project managers prior to and during a project.
- They are straightforward in concept and are supported by software.
- Their graphical representation of the project's tasks help to show the task interrelationships.
- Their ability to highlight the project's critical path and task slack time allows the project manager to focus more attention on the critical aspects of the project-time, costs and people.
- The project management software that creates the network usually provides excellent project tracking documentation.
- These methods are applicable in a wide variety of projects.

Scheduling Methods: Weaknesses

- In order for these methods to be useful, project tasks have to be clearly defined as well as their relationships to each other.
- These methods do not deal very well with task overlap. They assume the following tasks begin after their preceding tasks end.
- They are only as good as the time estimates that are entered by the project manager.
- By design, the project manager will normally focus more attention on the critical path tasks than other tasks, which could be problematic for near-critical path tasks if overlooked.

Software Project Tracking and Oversight Activities Performed

- | | |
|---------------|--|
| Goal 1 | Actual results and performances are tracked against the software plans. |
| Activity 5 | The size of the software work products (or size of the changes to the software work products) are tracked, and corrective actions are taken as necessary. |
| Activity 6 | The project's software effort and costs are tracked, and corrective actions are taken as necessary. |
| Activity 7 | The project's critical computer resources are tracked, and corrective actions are taken as necessary. |
| Activity 8 | The project's software schedule is tracked, and corrective actions are taken as necessary. |
| Activity 9 | Software engineering technical activities are tracked, and corrective actions are taken as necessary. |
| Activity 10 | The software risks associated with cost, resource, schedule, and technical aspects of the project are tracked. |

Software Project Tracking and Oversight Activities Performed

- | | |
|---------------|---|
| Goal 1 | Actual results and performances are tracked against the software plans . |
| Activity 1 | A documented software development plan is used for tracking the software activities and communicating status. |
| Activity 11 | Actual measurement data and replanning data for the software project are recorded. |
| Activity 12 | The software engineering group conducts periodic internal reviews to track technical progress, plans, performance, and issues against the software development plan. |
| Activity 13 | Formal reviews to address the accomplishments and results of the software project are conducted at selected project milestones according to a documented procedure. |

Software Project Tracking and Oversight Activities Performed

Goal 2	Corrective actions are taken and managed to closure when actual results and performance deviate significantly from the software plans.
Activity 2	The project's software development plan is revised according to a documented procedure.
Activity 5	The size of the software work products (or size of the changes to the software work products) are tracked, and corrective actions are taken as necessary.
Activity 6	The project's software effort and costs are tracked, and corrective actions are taken as necessary.
Activity 7	The project's critical computer resources are tracked, and corrective actions are taken as necessary.
Activity 8	The project's software schedule is tracked, and corrective actions are taken as necessary.
Activity 9	Software engineering technical activities are tracked, and corrective actions are taken as necessary.
Activity 10	The software risks associated with cost, resource, schedule, and technical aspects of the project are tracked.

Software Project Tracking and Oversight Activities Performed

Goal 3	Changes to software commitments are agreed to by the affected groups and individuals.
Activity 3	Software project commitments and changes to commitments made to individuals and groups external to the organization are reviewed with senior management according to a documented procedure.
Activity 4	Approved changes to commitments that affect the software project are communicated to the members of the software engineering group and other software-related groups.

Tracking: Elementary Metrics

Unit of measure	Characteristics addressed
Counts of physical source lines of code	Size, progress, reuse
Counts of staff-hours expended	Effort, cost, resource allocations
Calendar dates	Schedule
Counts of software problems and defects	Quality, readiness for delivery, improvement trends

Project Tracking - Manpower & Effort

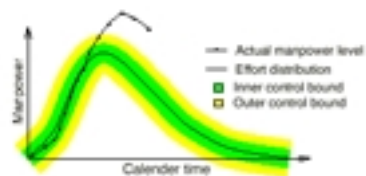


Figure 13: Rate curve. The actual effort values are plotted against the distribution.

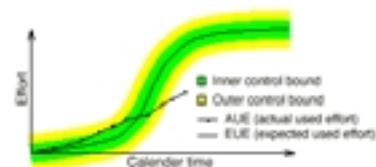


Figure 14: Cumulative curve. The actual effort values are plotted against the distribution.

Steen Andersen, Peter Stegenborg Larsen,
Carsten Lindholst: *Evaluation and Evolution of
Navi - a Web Based Tool for Project Planning
and Tracking*, Masters Thesis, Computer Science,
Aalborg University, 1998.

Project Tracking - Lines of Code & Defects

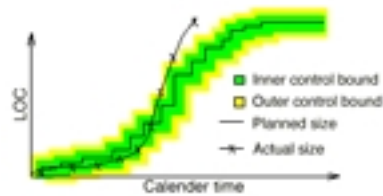


Figure 18: The actual size in LOC plotted against the planned size

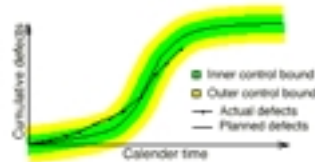


Figure 19: Tracking cumulative defect arrival against a planned Rayleigh distribution.

Steen Andersen, Peter Stegenborg Larsen,
Carsten Lindholst: *Evaluation and Evolution of
Navi - a Web Based Tool for Project Planning
and Tracking*, Masters Thesis, Computer Science,
Aalborg University, 1998.