

5. Project planning and management

- Role of a manager
- Charts and Critical Path Analysis
- Estimation Techniques
- Monitoring

Role of a manager

- Directs resources for the achievement of goals
- LEADER also provides
 - Vision
 - Inspiration
 - Rises above the usual
- No one right way to manage

Management Continuum

Authoritarian

Democratic

Autocratic

Consultative

Participate



Solves problems alone

Discusses Problems

Chairperson

Dictates decisions

Makes decision

Agrees problem

Creates consensus

Managerial Roles

» (after Henry Mintzberg)

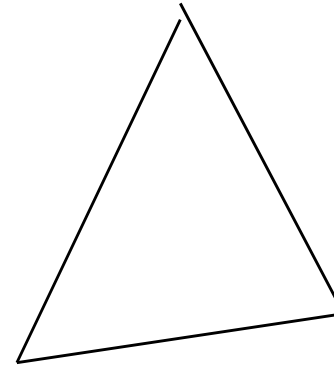
- Interpersonal
 - Figurehead
 - Leader
 - Liaison
- Informational Roles
 - Monitor
 - Disseminator
 - Spokesperson
- Decisional Roles
 - Entrepreneur
 - Resource Allocator
 - Disturbance Allocator
 - Negotiator

Qualities

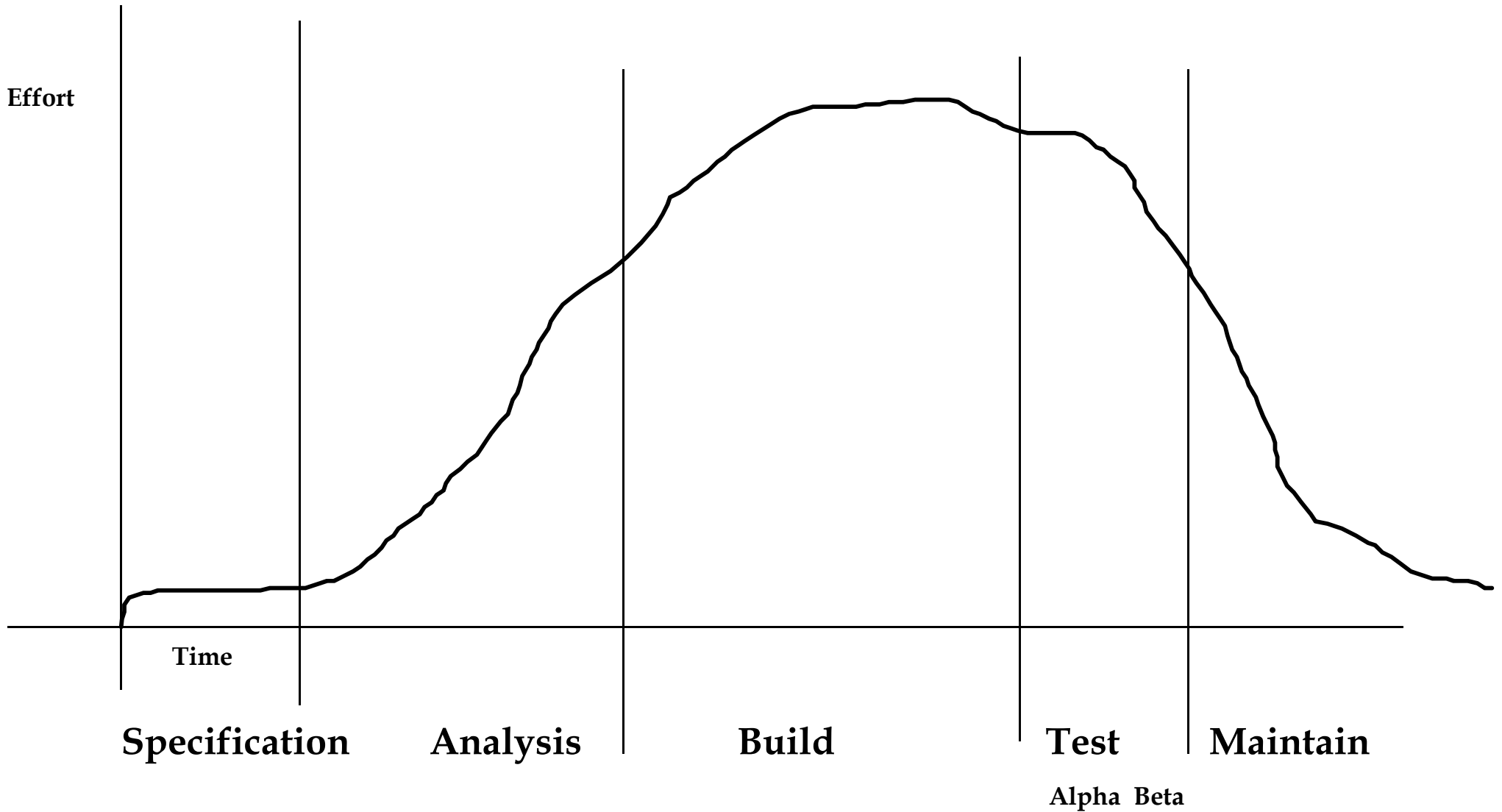
- Technical/Professional knowledge
- Organisational know-how
- Ability to grasp situation
- Ability to make decisions
- Ability to manage change
- Creative
- Mental flexibility - Learns from experience
- Pro-active
- Moral courage
- Resilience
- Social skills
- Self Knowledge

Variables

- Resource
- Time
- Function
- “ You can have any two of quick, good or cheap, but not all three”

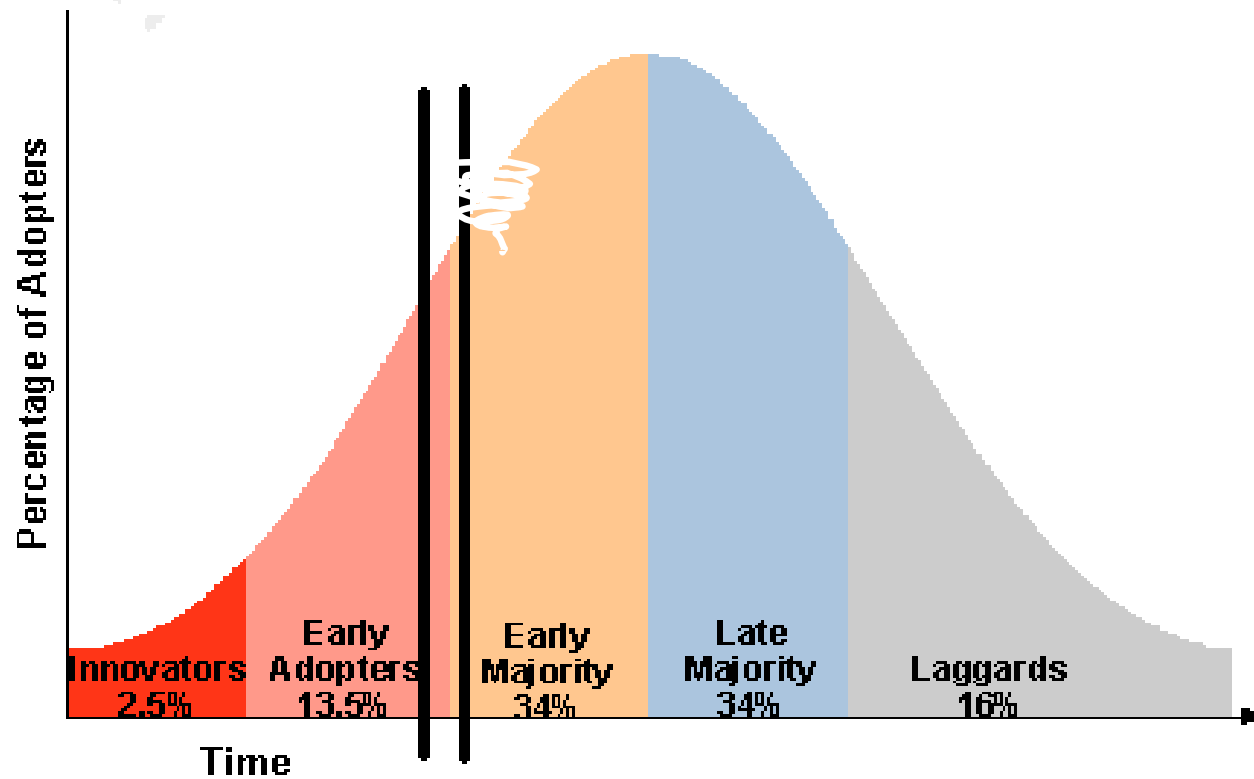


Development cycle:



Crossing the Chasm

- Geoffrey Moore, after Everett Rogers



Approaches and methodologies

- Top Down
 - Waterfall decomposition
- Bottom Up
 - meta machine
- Rapid Prototype
 - successive refinement
 - Agile Engineering
- Muddle through

Agile Engineering

In February 2001, 17 software developers^[5] met at the [Snowbird, Utah](#) resort, to discuss lightweight development methods. They published the *Manifesto for Agile Software Development*^[1] to define the approach now known as agile software development. Some of the manifesto's authors formed the Agile Alliance, a nonprofit organization that promotes software development according to the manifesto's principles.

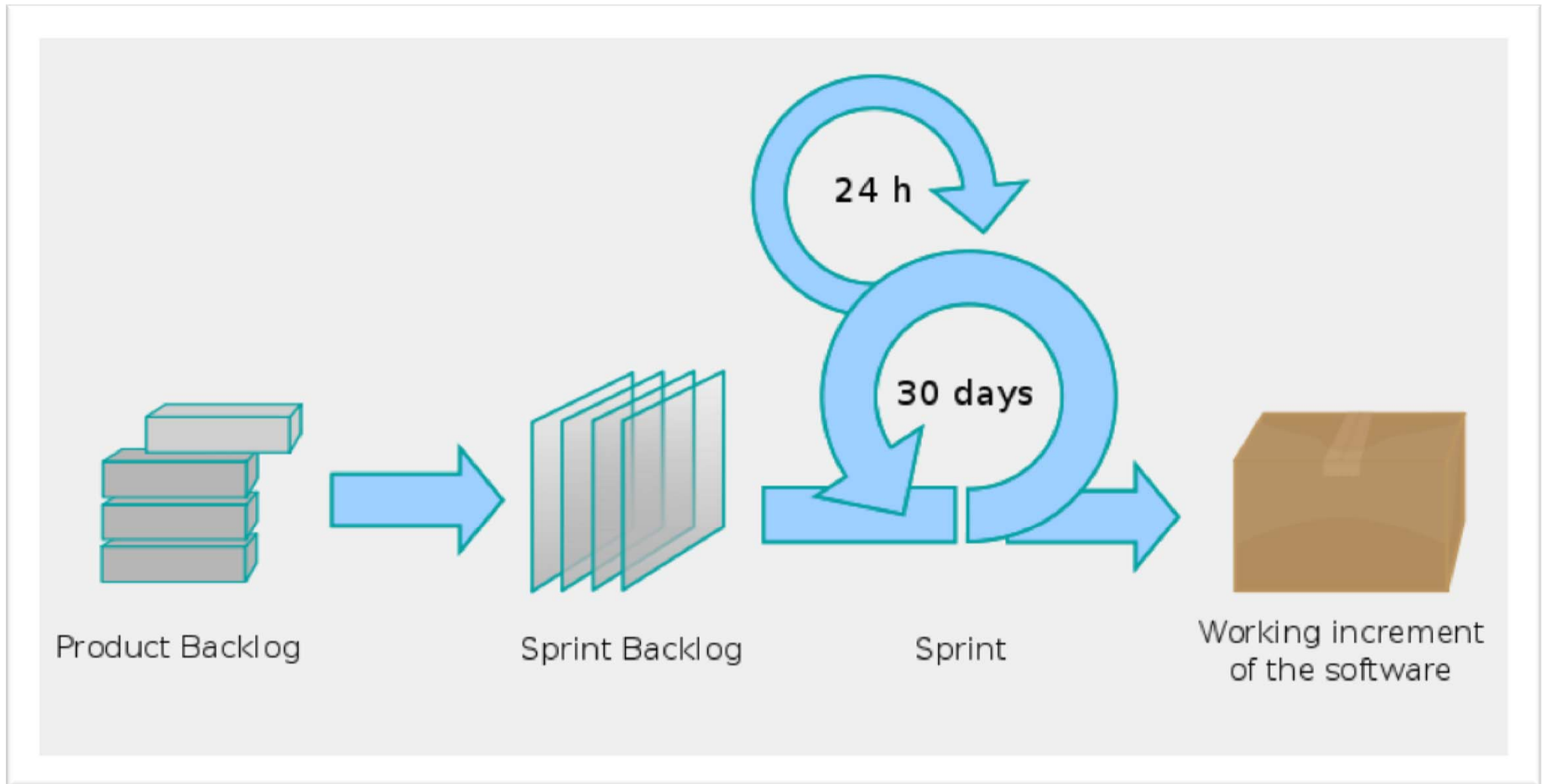
The Agile Manifesto reads, in its entirety, as follows:^[1]

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more

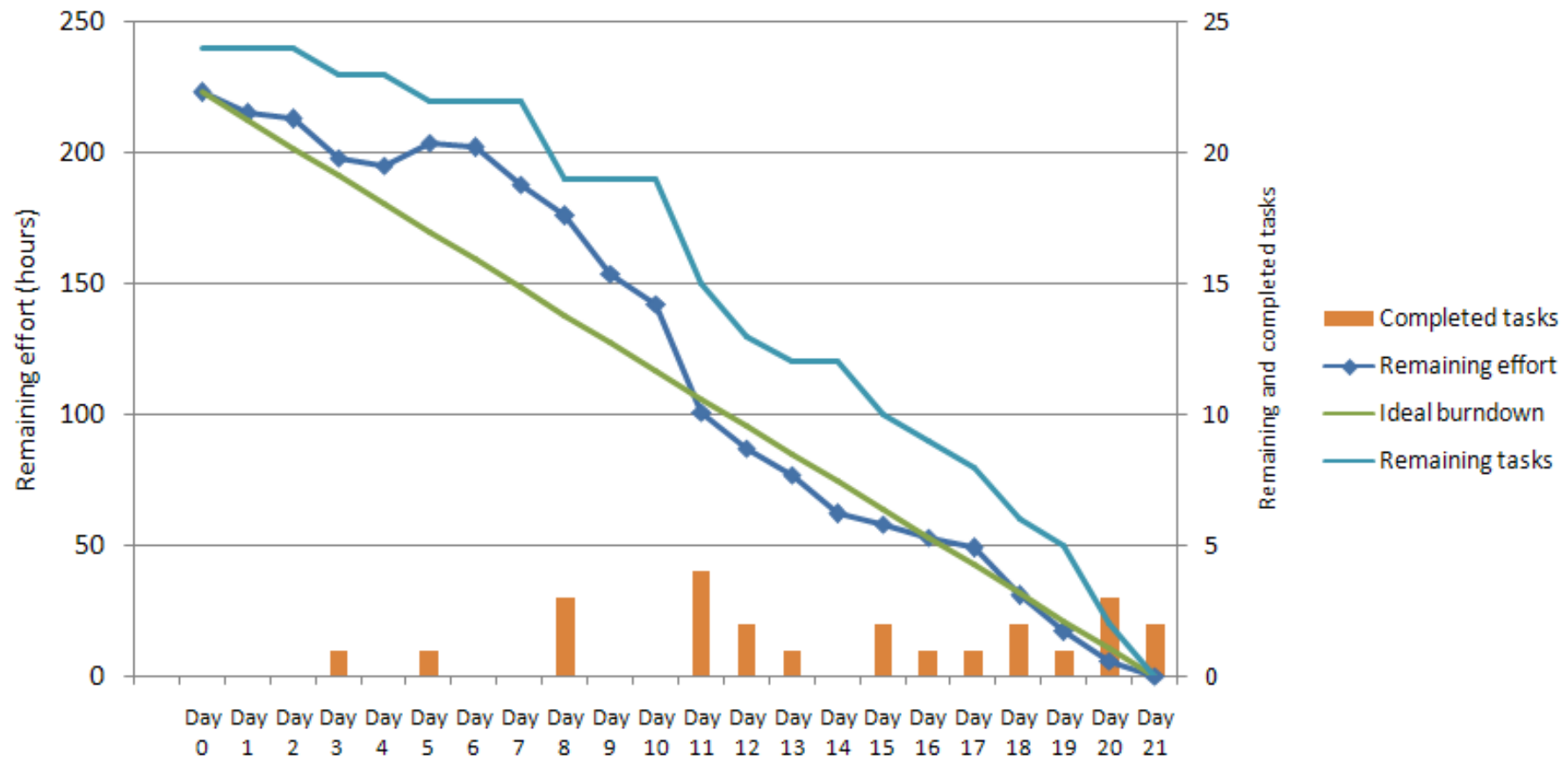
Sprints, Scrums, Timeboxes



DeGrace, Peter; Stahl, Leslie Hulet (1990-10-01). [Wicked problems, righteous solutions](#). Prentice Hall. [ISBN 978-0-135-90126-7](#).

Adapted from [http://en.wikipedia.org/wiki/Scrum_\(development\)](http://en.wikipedia.org/wiki/Scrum_(development))

Sample Burndown Chart



Spiral Methodology

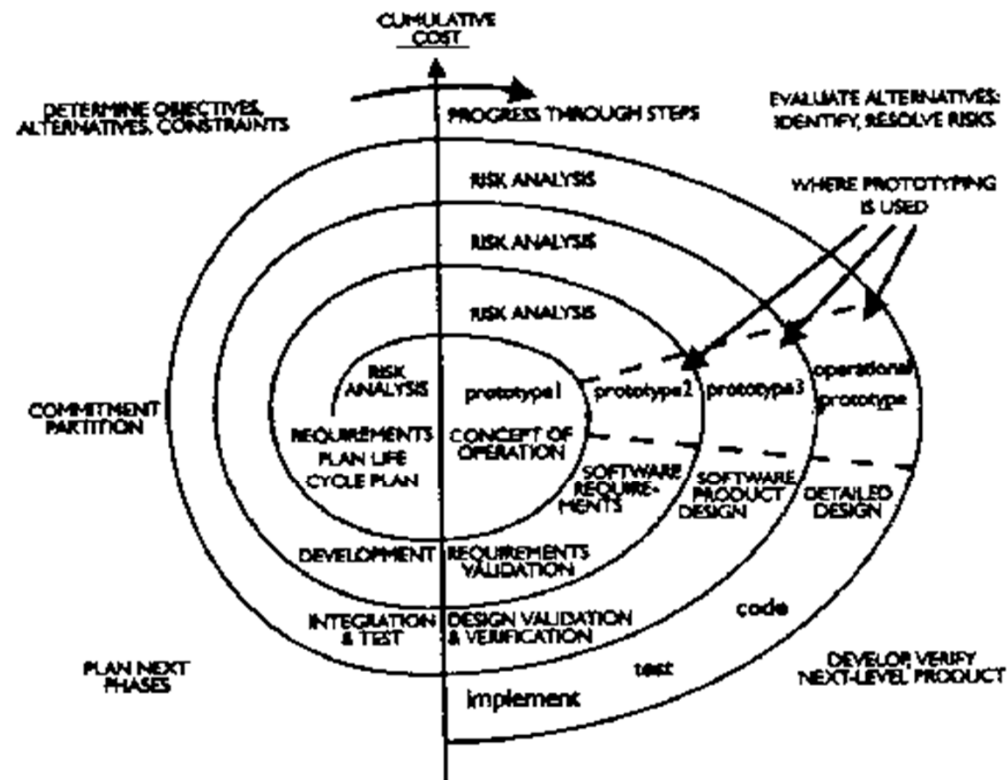


FIGURE 4: BOEHM SPIRAL MODEL (REPRINTED WITH PERMISSION FROM [10]).

Phase	Deliverables
Envisioning – agreeing the overall direction and the contents of this phase	Vision/Scope document Risk assessment Project structure
Planning – design for this phase.	Functional specification Risk assessment Project schedule
Developing – the actual build	Frozen functional specification Risk management plan Source code and executables Performance support elements Test specification and test cases Master project plan and master project schedule
Stabilising – test, debug, rework.	Golden release Release notes Performance support elements

Meetings

- Daily Scrum
 - Each day during the sprint, a project status meeting occurs. This is called a *daily scrum*, or *the daily standup*. This meeting has specific guidelines: The meeting starts precisely on time. All are welcome, but normally only the core roles speak. The meeting is [timeboxed](#) to 15 minutes.
- Scrum of scrums
- Each day normally after the daily scrum. These meetings allow clusters of teams to discuss their work, focusing especially on areas of overlap and integration. A designated person from each team attends.
- Sprint Planning Meeting
 - At the beginning of the sprint cycle (every 7–30 days), a “Sprint Planning Meeting” is held. Select what work is to be done. Prepare the Sprint Backlog that details the time it will take to do that work, with the entire team. Identify and communicate how much of the work is likely to be done during the current sprint. Eight hour time limit
 - (1st four hours) Product Owner + Team: dialog for prioritizing the Product Backlog
 - (2nd four hours) Team only: hashing out a plan for the Sprint, resulting in the Sprint Backlog
- At the end of a sprint cycle:
- Sprint Review Meeting
 - Review the work that was completed and not completed. Present the completed work to the stakeholders (a.k.a. “the demo”). Incomplete work cannot be demonstrated. Four hour time limit
- [Sprint Retrospective
 - All team members reflect on the past sprint. Make continuous process improvements. Two main questions are asked in the sprint retrospective: What went well during the sprint? What could be improved in the next sprint? Three hour time limit

Pert and Gantt Charts

- Visual representation of project
- Microsoft Project

Example: Getting up in the morning

Task	Duration (mins)
1 Alarm rings	0
2. Wake Up	3
3. Get out of bed	5
4. Wash	5
5. Get dressed	5
6. Put kettle on	2
7 Wait for kettle to boil	5
8 Put toast on	2
9 Wait for Toast	3
10 Make coffee	3
11 Butter Toast	2
12 Eat Breakfast	10
13 Leave for Lectures	0

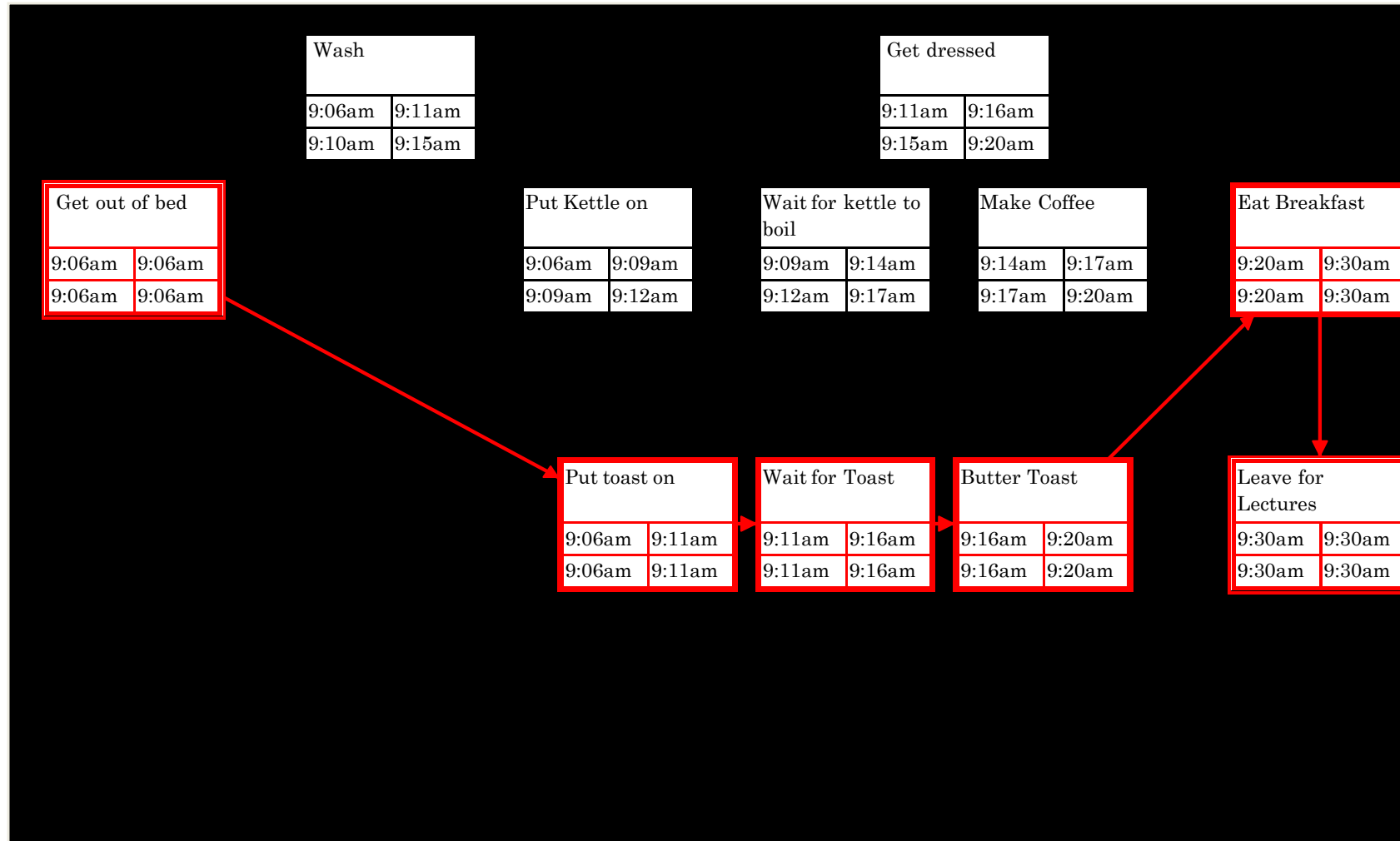
Pert Chart



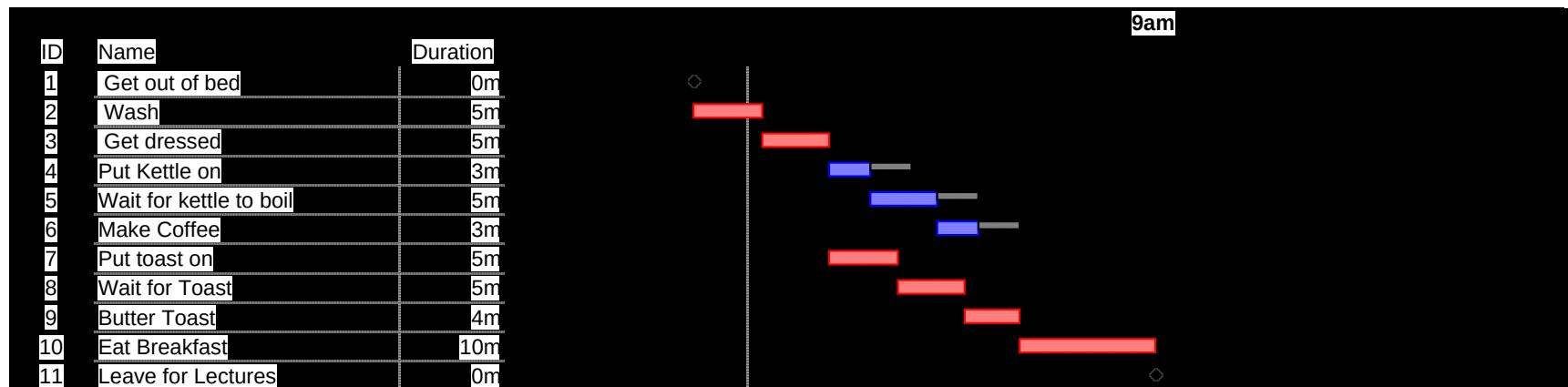
Critical Path Analysis

- Compute earliest and latest start/finish for each task
- The difference is the *slack*
- The Critical Path joins the tasks for which there is no slack
- Any delay in tasks on the on the critical path affects the whole project

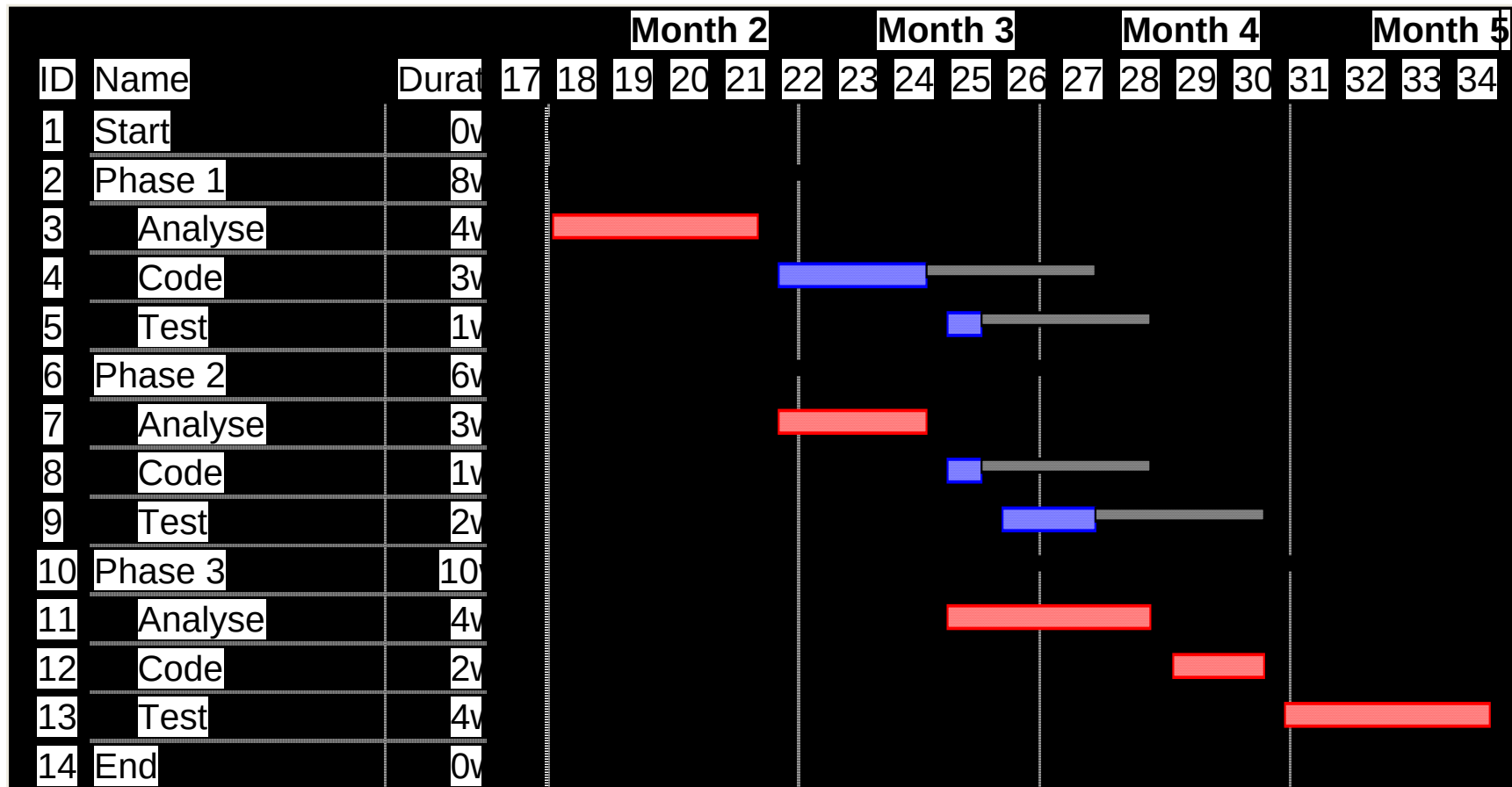
Pert Chart



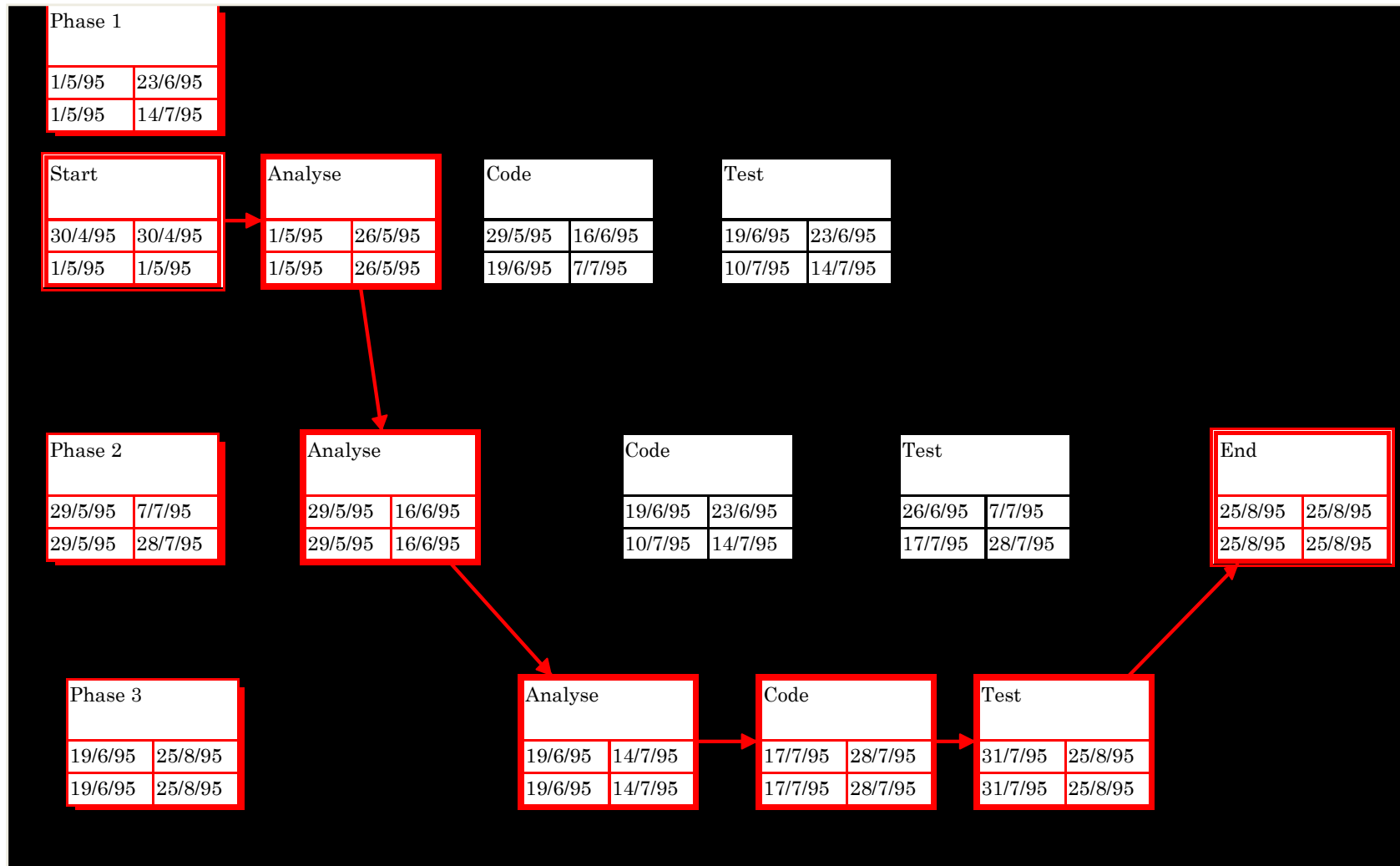
Gantt Chart

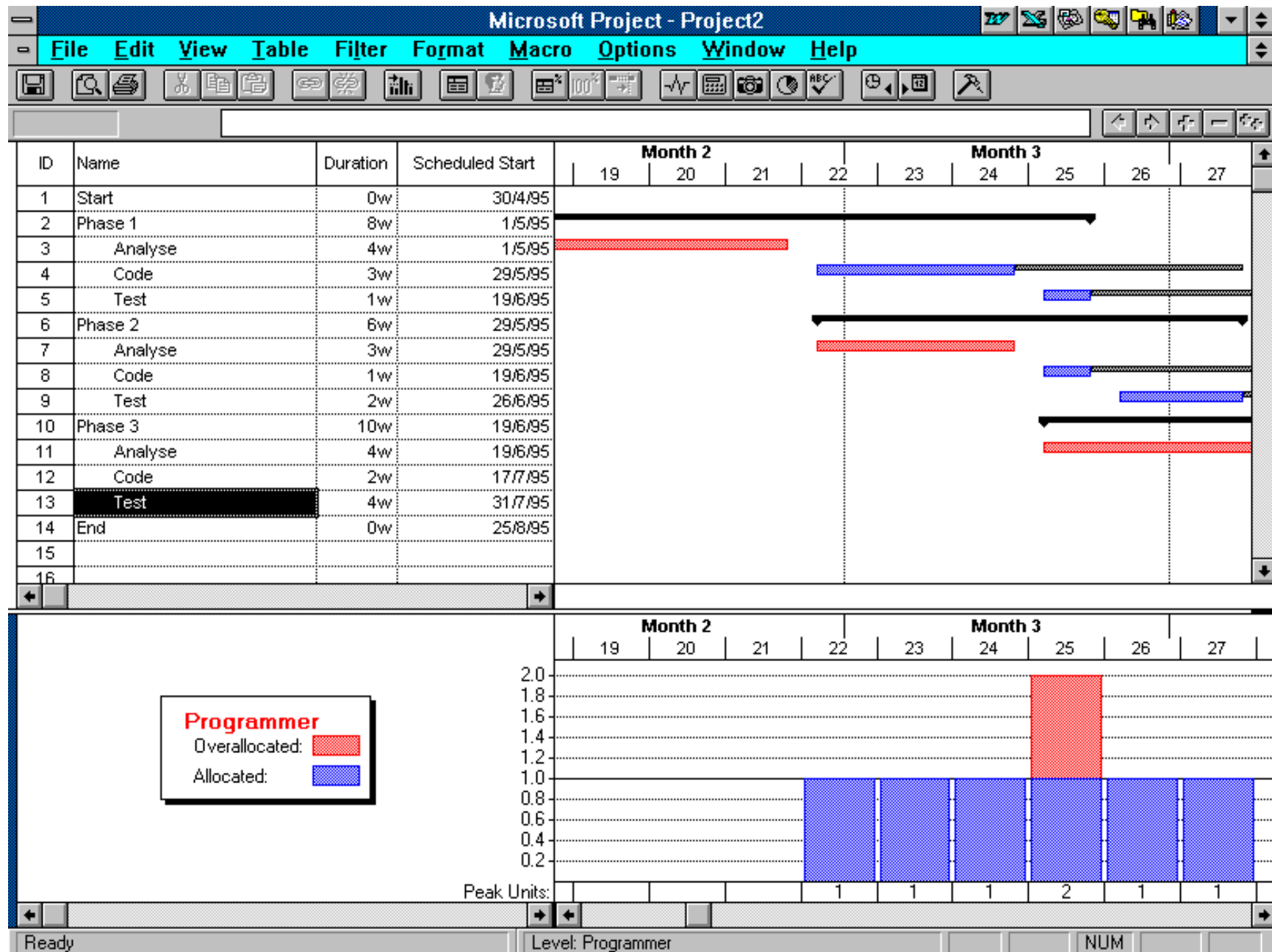


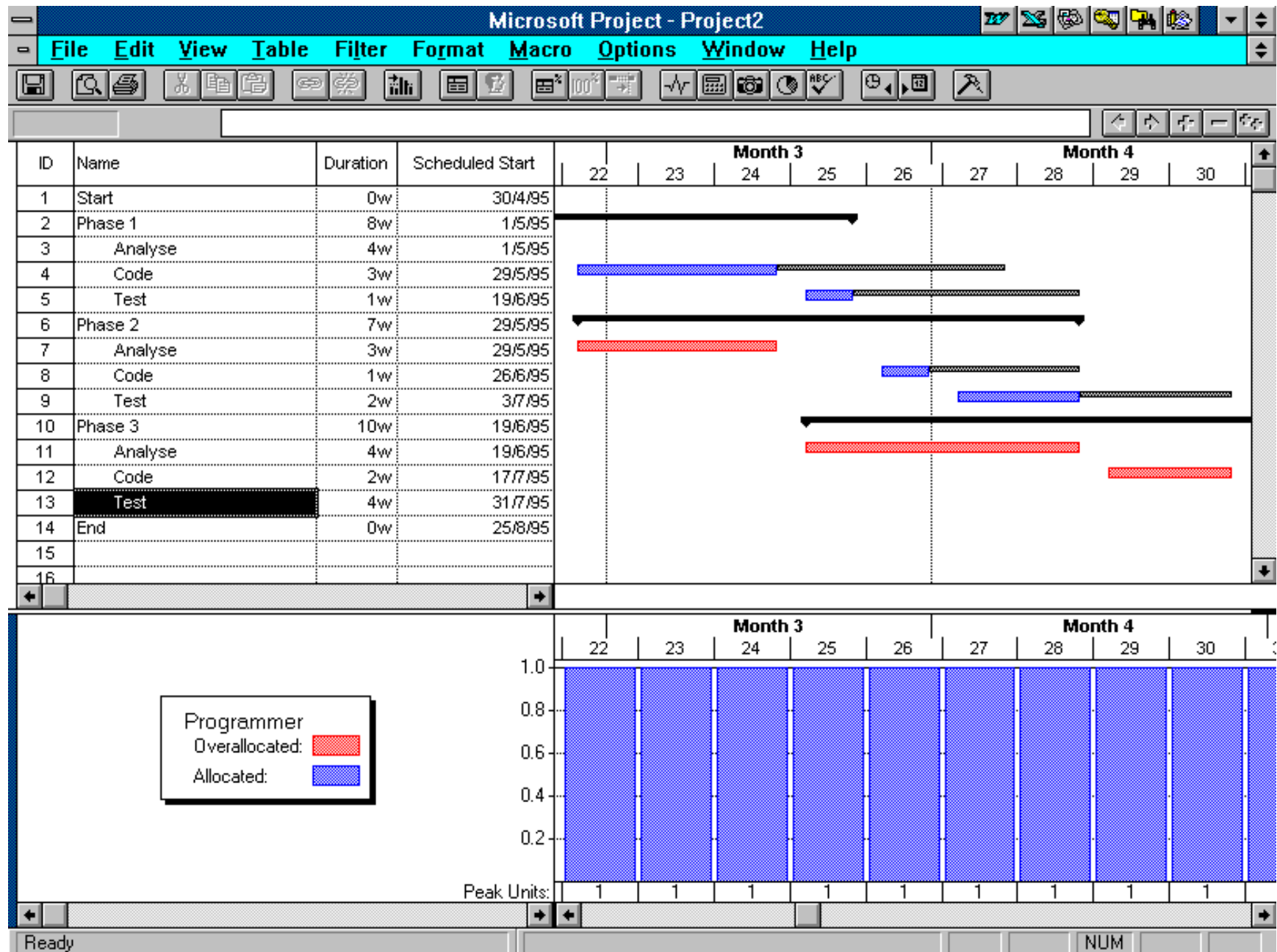
Example



Example Pert



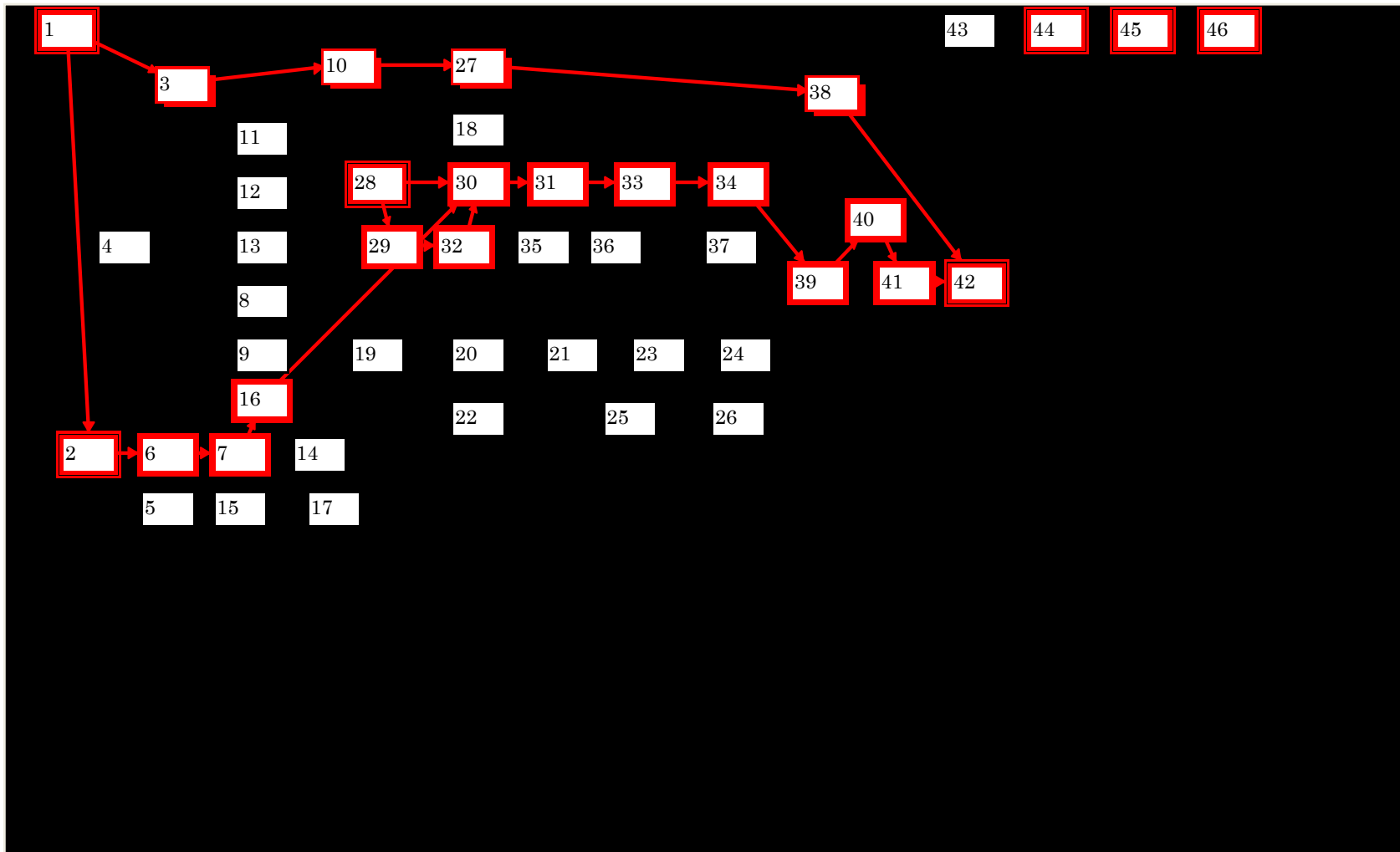




Levelling

- Adjust tasks to match resources available
- Automatic systems available, but do not always give an optimum result
- Tasks may be delayed within slack without affecting project dates
- Otherwise consider extending project, or using more resource
- Adding resource to late project may cause *RECURSIVE COLLAPSE*
 - consider carefully whether the benefits outweigh the additional learning delays and overheads
- Derive costings

Larger example



Estimation Techniques

- Experience
- Comparison with similar tasks
 - 20 lines of code/day
 - can vary by 2 orders of magnitude
- Decomposition
- Plan to throw one away
- 20 working days per month BUT 200 per year

Rules of Thumb

- Software projects:
 - estimate 10 x cost and 3 x time
- 1:3:10 rule
 - 1: cost of prototype
 - 3: cost of turning prototype into a product
 - 10: cost of sales and marketing
 - >>Product costs are dominated by cost of sales
- Hartree's Law
 - The time to completion of any project, as estimated by the project leader, is a constant (Hartree's constant) *regardless of the state of the project*
 - A project is 90% complete 90% of the time
- 80% Rule
 - Don't plan to use more than 80% of the available resources
 - Memory, disc, cycles, programming resource....

Cynic's Project Stages

- Enthusiasm
- Disillusionment
- Panic
- Persecution of the innocent
- Praise of the bystander