Introduction to the Course

Motivations (1/3)

- Many courses in computer science and electronic engineering focus on the technical aspects and on the notations to develop good software (e.g., how to do testing; how to write requirements; how to model systems; UML; Java)
- However, in order to build good software, a well defined and managed process, which organizes the activities in an efficient and controlled way

Motivations (2/3)

- Consider the following:
  - Writing good requirements is no good if you don't have a controlled process, for instance, to accept changes and trace revisions
  - The techniques to do testing are useless if you don't have any time left to do testing (because, for instance, you underestimated the development time and are late with a release)
  - The development of a software system requires to execute, monitor, and control various activities which have little or nothing to do with writing code. Consider, for instance: training users, packaging a product, managing publicity and communication, writing user documentation

Motivations (3/3)

- If you want to deliver on time and within budget a product which has the quality properties agreed upon (be it a software or any other product), you need:
  - A process to define a schedule, a budget, and agree on the (quality) characteristics of a product
  - A list of techniques to define, agree, plan, execute, and monitor: goals, quality, time, and costs

Skills and some goals of the course

- Managing a (software development) project, thus, requires specific competences, skills, and techniques
- Some of the questions you will be able to answer at the end of this course include:
  - How do I estimate how long it will take to complete a task?
  - How much am I going to charge for a project?
  - How do I keep the team motivated and ensure projects are fulfilling and an occasion to learn, grow, and advance in one's career?
  - How do I deal with project risks?
  - How do I assess whether the project is on time, on budget, on schedule?
  - How do I control the quality of the final output?

Software Project Management

The Project Management techniques are intrinsically multidisciplinary …

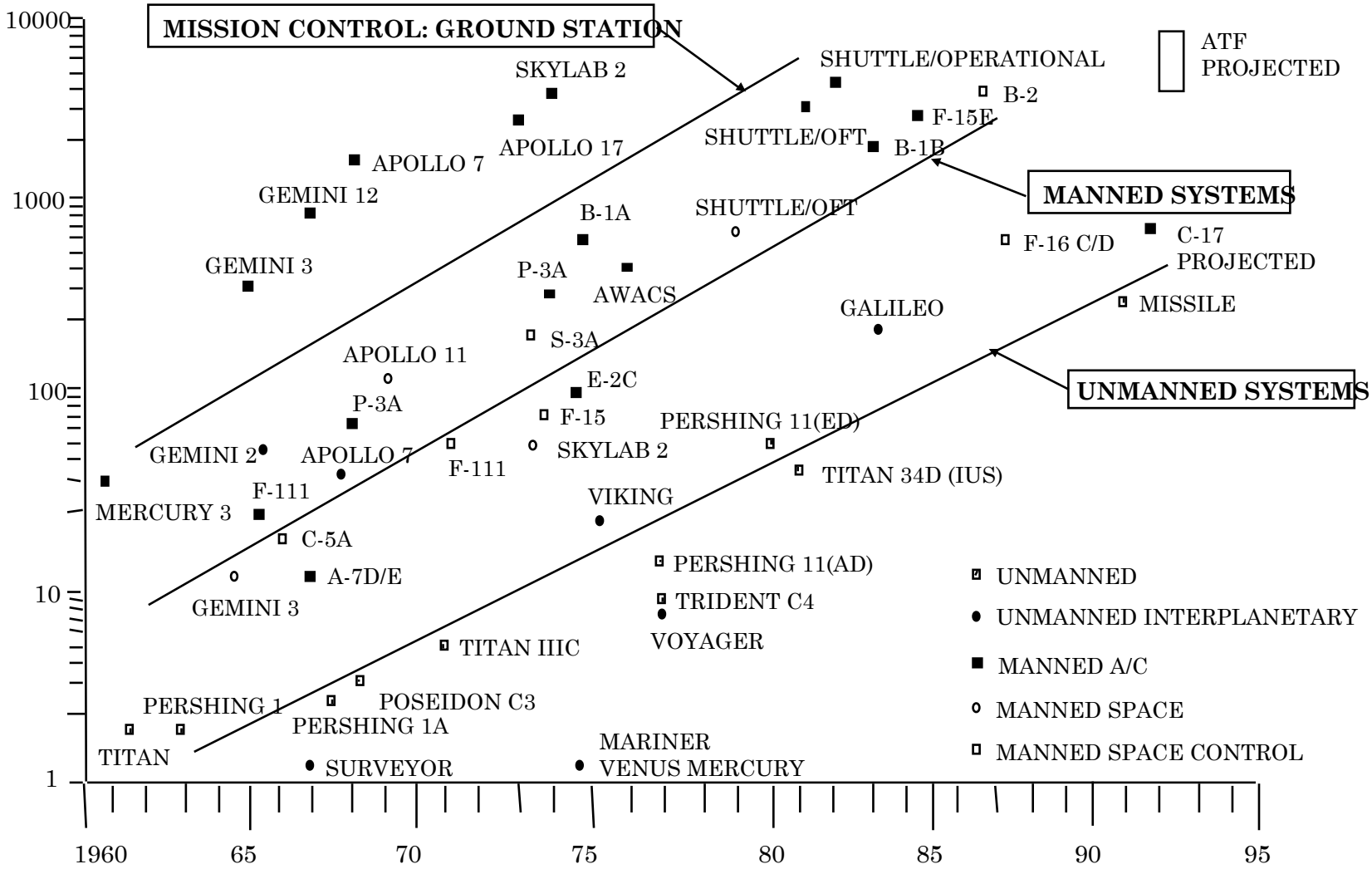
… what you will learn in this course is applicable to virtually any other (engineering) domain.

There are however certain characteristics that make the management of software projects particularly interesting.

Software Project Management

- Software project management is interesting and challenging because:
  - The product is intangible
  - The product is uniquely flexible (e.g. different sizes; different constraints)
  - Many software projects are 'one-off'
  - The development process is uniquely flexible
  - Size and complexity are increasing exponentially
  - Human lives might depend on software running as expected (consider the control system of an airplane)- safety critical systems

Complexity (1/2)



Source: Mars and Beyond: NASA's Software Challenges in the 21st Century

Dr. Michael R. Lowry. NASA Ames Research Center. December 5, 2003
web.cecs.pdx.edu/~black/S3S/MichaelLowry.ppt

Complexity (2/2)

- The entire Saturn V stack, that is, all three stages of the booster plus the command module and the lunar module, had less computing capacity combined than today's typical cell phone.

Source: Apollo
by Charles Murray and
Catherine Bly Cox

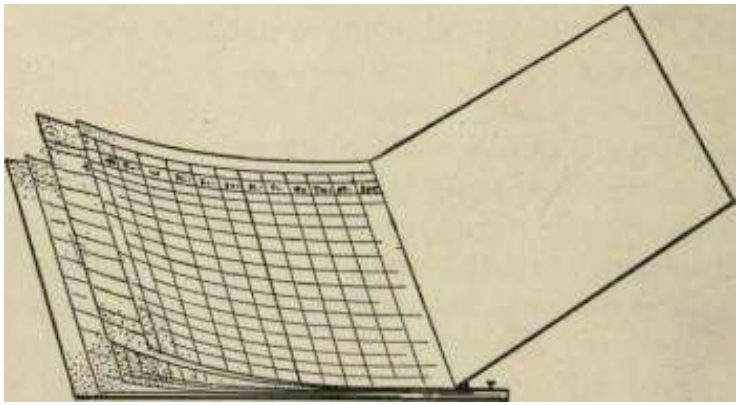
A Brief History of (Software) Project Management

Frederick Winslow Taylor

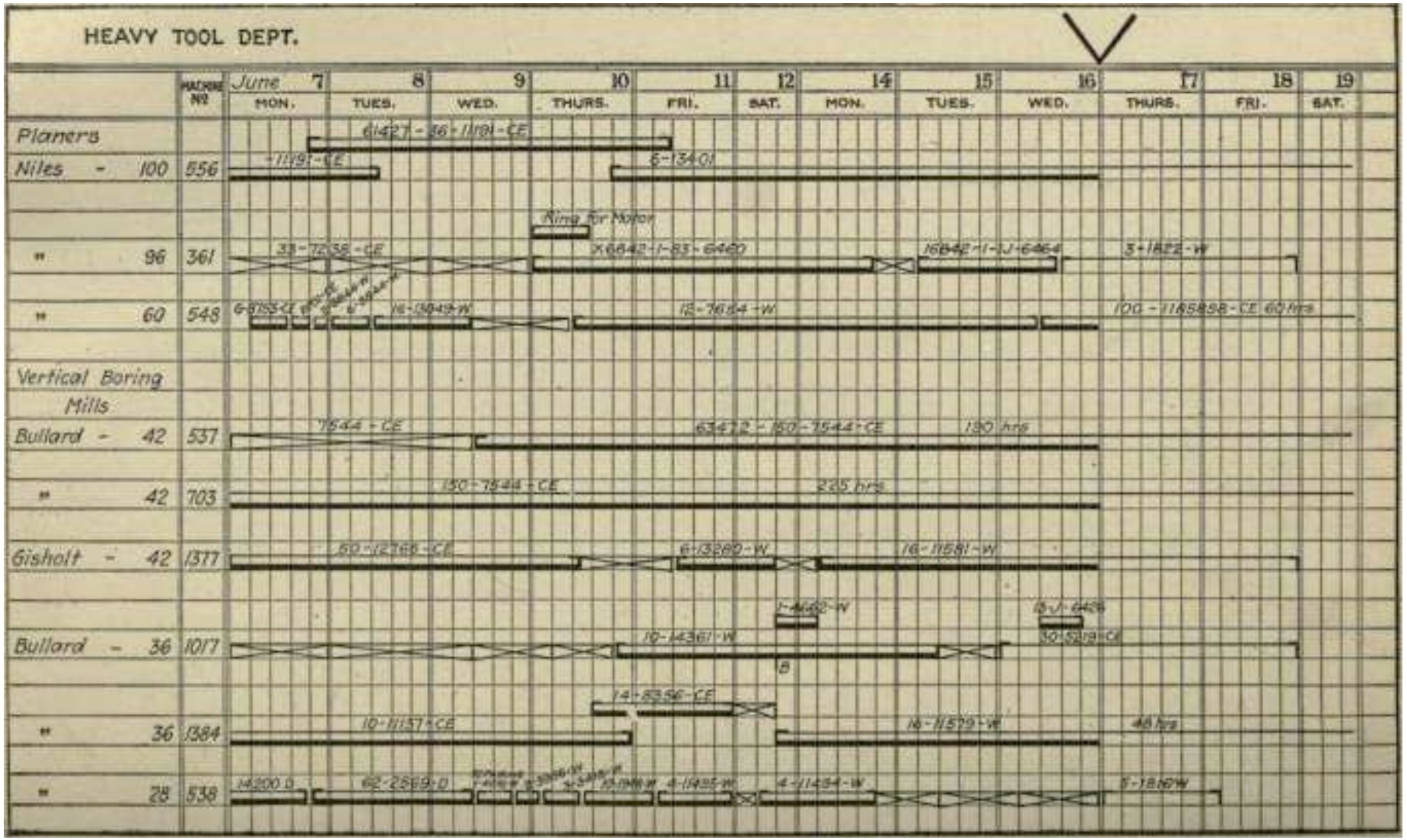
- Taylor's "Scientific Management" is the first theory of work and management
- Beginning of 1900
- A negative view of workers (they perform at the slowest rate which goes unpunished)
- Some more interesting characteristics:
  - Scientific definition of work
  - Scientific selection of personnel
  - Sharing of responsibilities between workers and management
  - Incentives and rest periods (to make workers more efficient)

Henry Gantt

- Gantt's "Gantt chart" notation is still used today to schedule projects
- Defined during the First World War
- First used to schedule and monitor work and progress in ship building: distinction between work and progress
- His book available for download from archive.org



Example of Gantt Chart



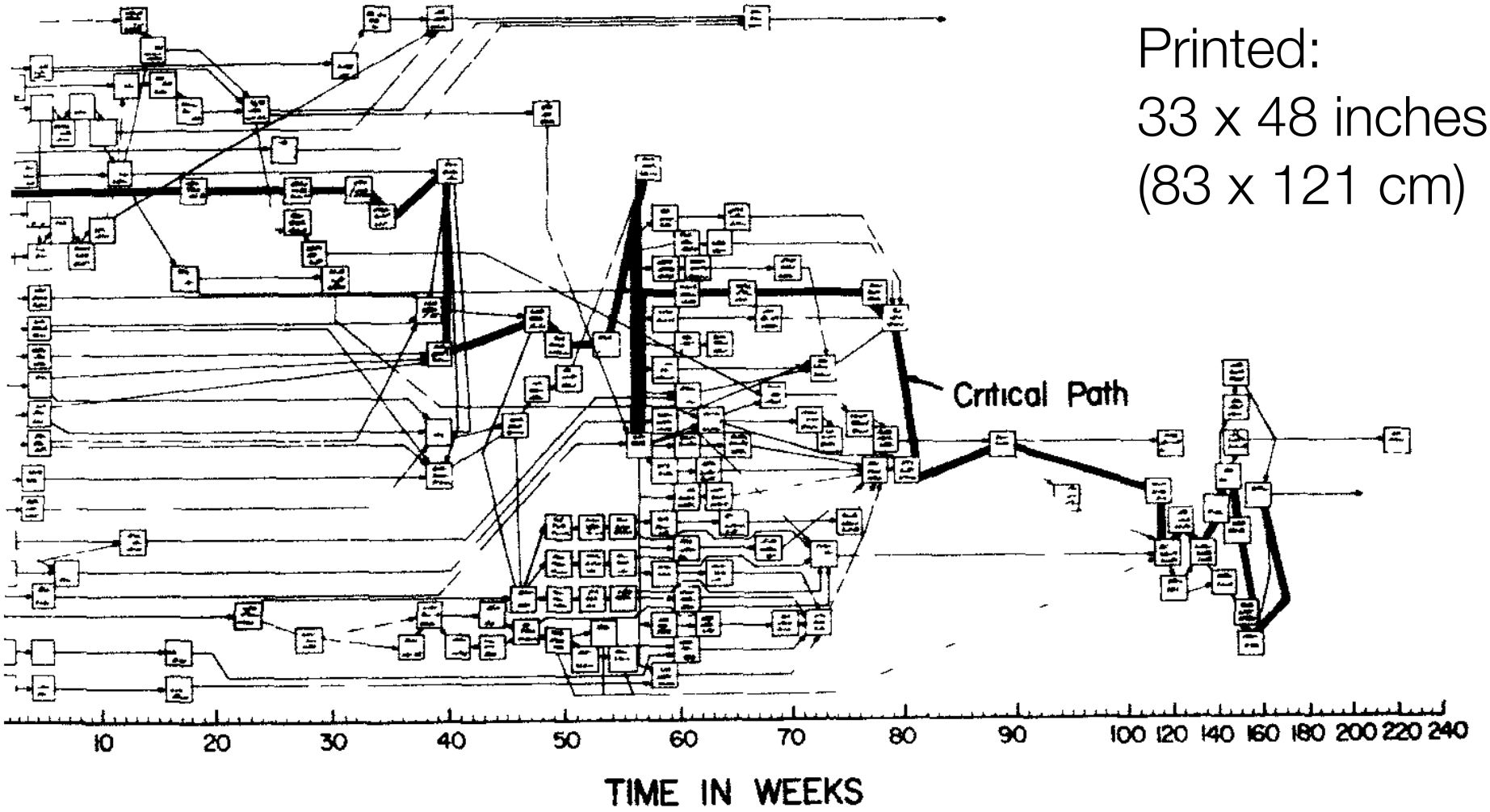
Source: The Gantt chart, a working tool of management
Clark, Wallace and Gantt, Henry

Fifties and Sixties

- 2nd World War:
  - Manhattan Project: process flow diagrams
- 1957 **CPM** (Critical Path Method)
  - Mathematically based algorithm for scheduling a set of project activities, used to plan maintenance activities in plants
  - Dupont + Remington Rand UNIVAC team
  - No fundamental changes to date
- 1958 **PERT** (Program Evaluation and Review Technique)
  - U.S. Navy Polaris missile program (Booz Allen & Hamilton (management consulting firm) working as ORSA team for Lockheed Missile System)

Motivation for CPM and PERT

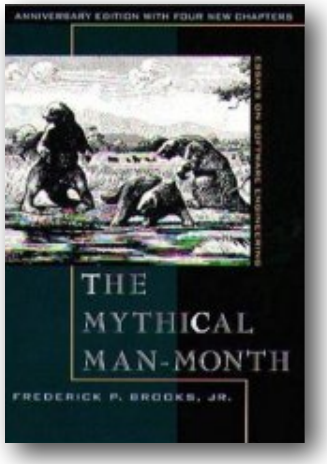
What activities could delay the project delivery of the following plan?



Printed:
33 x 48 inches
(83 x 121 cm)

Fifties and Sixties

- **1960's** Big Government contracts (Vietnam, nuclear power plants, NASA Apollo): standardization and automation
  - PERT/COST and WBS become compulsory in Government's sponsored projects
  - Earned Value Analysis (EVA) is defined
  - Configuration management
  - Project organizations (PMI, IPMA) promote profession and techniques
  - (1961) IBM uses PM commercially



- **1970** Software development gets into the equation
  - EVA developed for monitoring schedule and cost
  - Waterfall model for software development
  - The mythical man-month highlights many pitfalls of software development

The Computer Revolution

- **1980's**
  - Hardware and software proliferation make PM tools accessible to smaller firms
  - Hardware capacity grows exponentially (Moore's Law) and so does software
  - Estimation models (FP and COCOMO) are introduced to predict software complexity
- **1990's**
  - Total quality
  - Leaner, quicker, more responsive organizations
- **Today**
  - Web application and new application distribution models
  - Development with components and frameworks
  - Agility, quick interaction, constant feedback