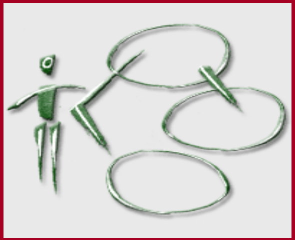


Software Engineering

Basic Information

Basic Information

- Teaching Assistant: **Alessandro Tomasi**
- eMail Teaching Assistant: **alessandro.tomasi@unitn.it**

Basic Information

October Schedule

14-Oct – 2.30 pm – Gantt + UML
16-Oct – 2 pm – Use Case diagrams
21-Oct – 2 pm – Yannis Velegrakis
23-Oct – 2 pm – Use Case diagrams
28-Oct – 2 pm – Yannis Velegrakis

Software Engineering

- Project Management
- Gantt Diagram
- Requirement Analysis
- A quick overview of UML

Software Engineering

Project Management

Project Management

- Goal:
  - organizing and managing resources (e.g. people, hardware, software, …)
  - complete a project within defined scope, quality, time and constraints
- Output
  - Project Management Document
- Diagram for Project Management:
  - Gantt Diagram

Software Engineering

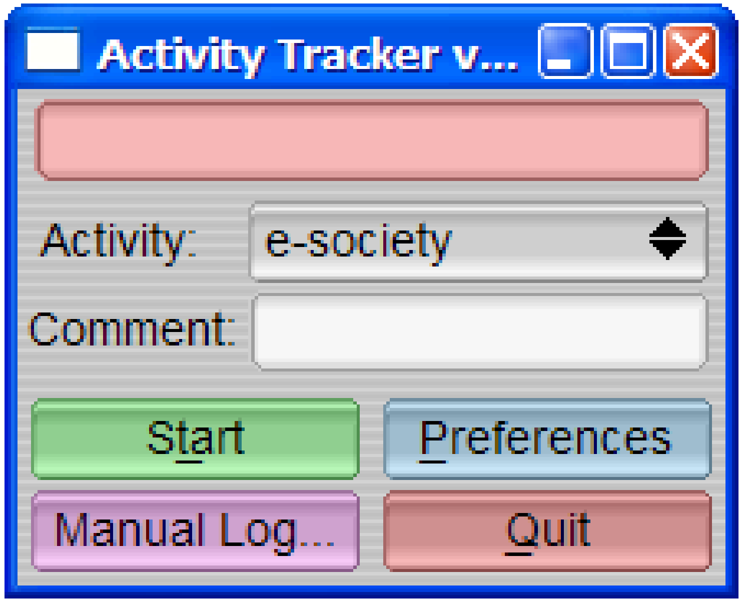
Gantt Diagram

Basic Concepts

- Planned Duration
- Planned Effort
- Resources
- Tasks
- Deliverable/Milestone
- Dependencies
- Actual Duration
- Actual Effort

Example: Time Tracker

We have been contacted by a small firm. They want us to build a system for letting employees track how they spend their time when working on a computer. The idea is that of a stop-watch: the users of the system can start and stop counting the time spent on different activities; the system logs such activities and can be used to produce reports.



The system can also be integrated with a billing system. The billing system receives all the information about the time spent by programmers on the different projects and computes the cost of projects. This information is then used to charge clients.

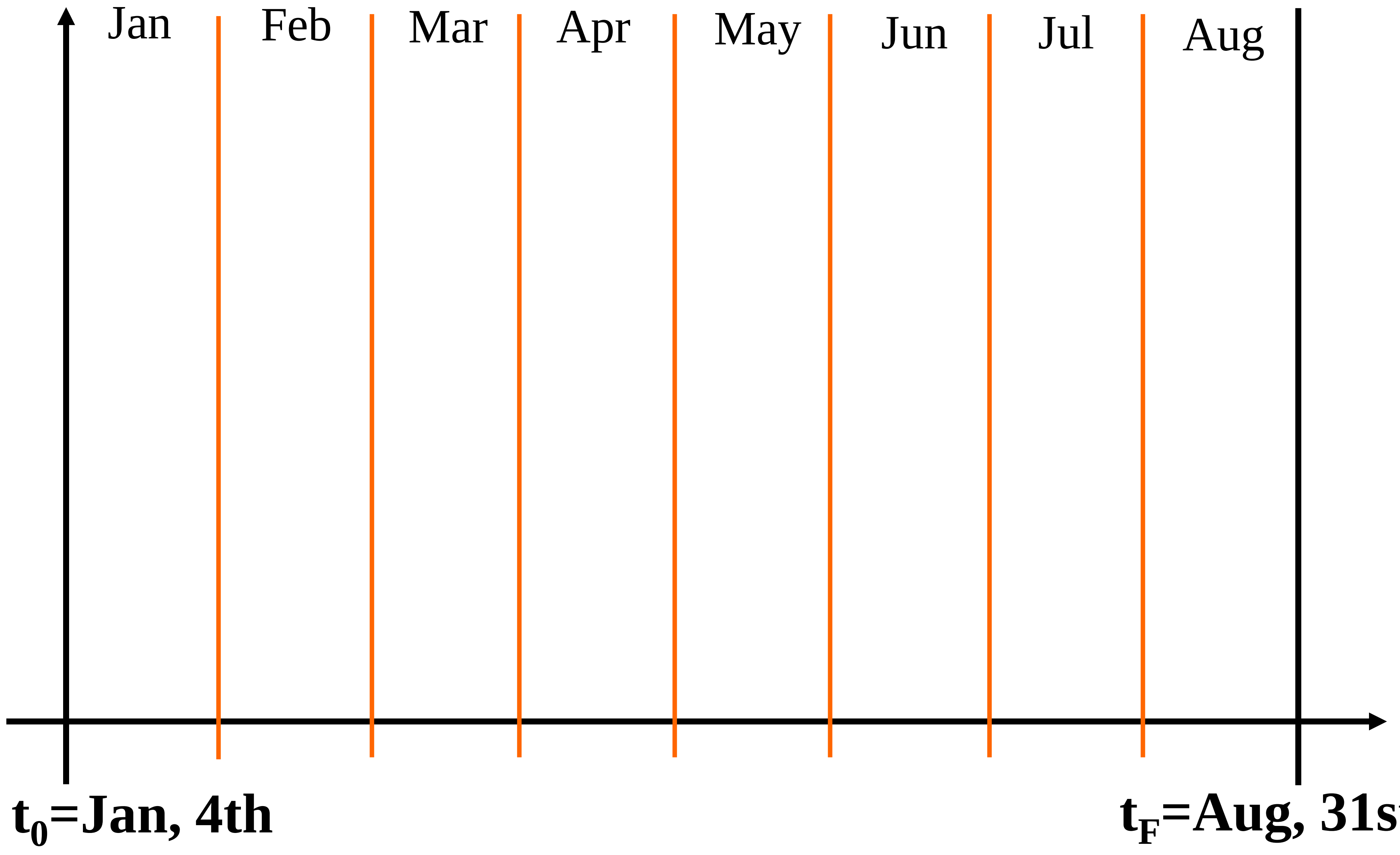
Parameters 1/6

Kickoff-Meeting := January, 4th 2014

Delivery := August, 31st 2014

Team :=	1 project manager (full time)
	1 analyst (full time)
	1 system architect (full time)
	1 programmer (full time)

Gantt Diagram



Parameters 2/6

Elapsed Time:= t_f - t_0= 240 days

Milestone:= 31st, August

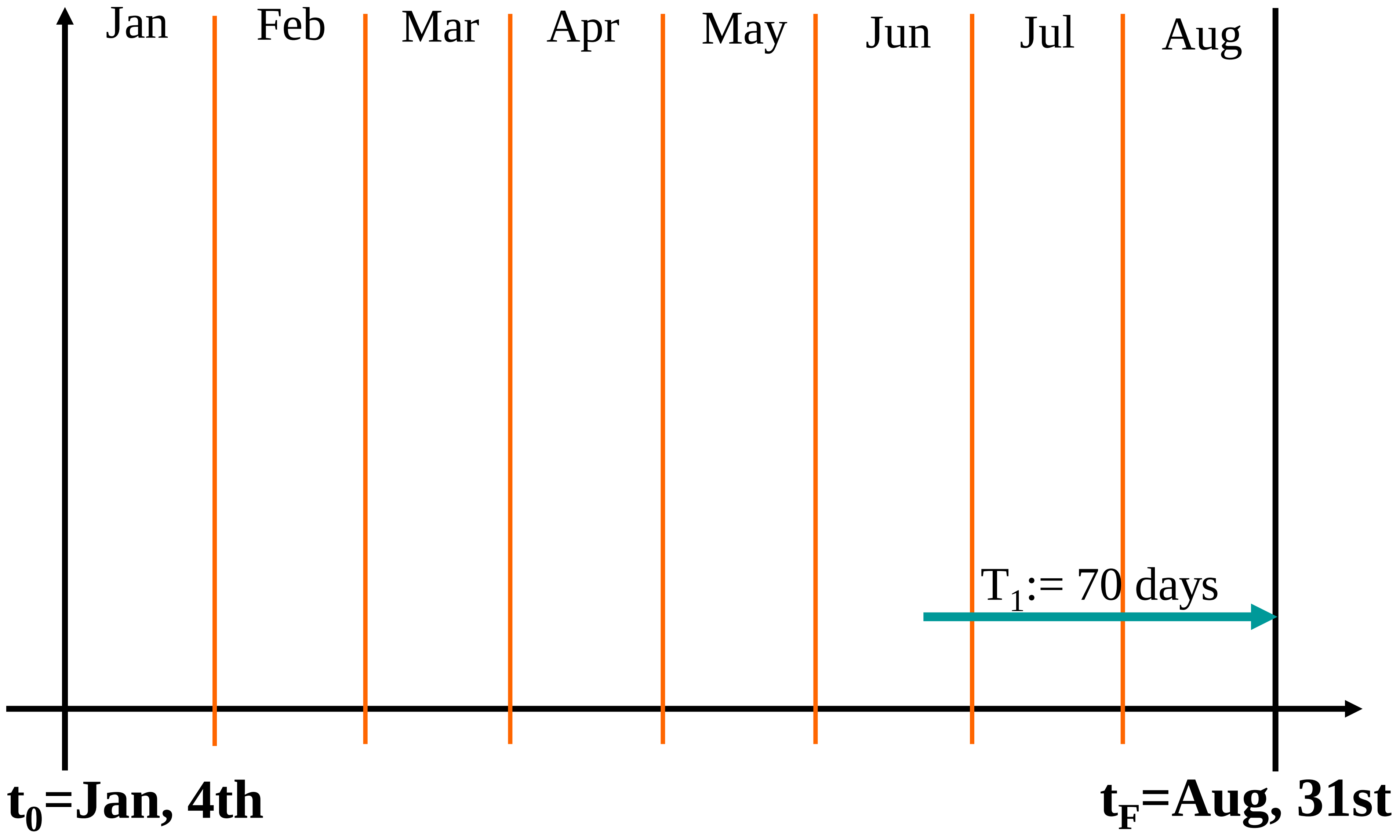
Deliverable:= Software

Development Processes:= Waterfall

Parameters 3/6

	Activity	Days	Milestone	Deliverable
T_4				
T_3				
T_2				
T_1	V&V	70	August, 31st	Software

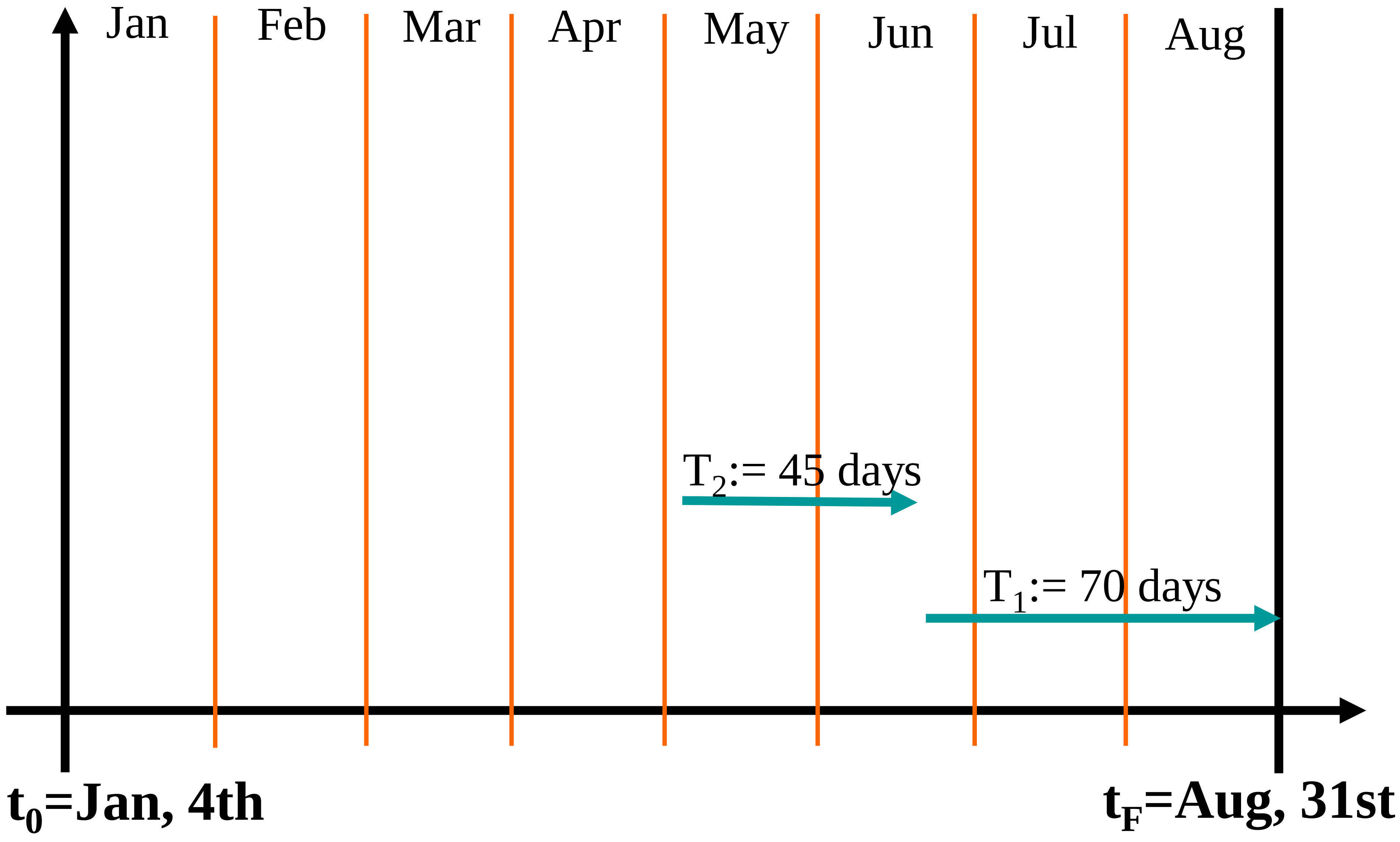
Gantt 3/6



Parameters 4/6

	Activity	Days	Milestone	Deliverable
T_4				
T_3				
T_2	Implementation	45	June, 22nd	Code
T_1	V&V	70	August, 31st	Software

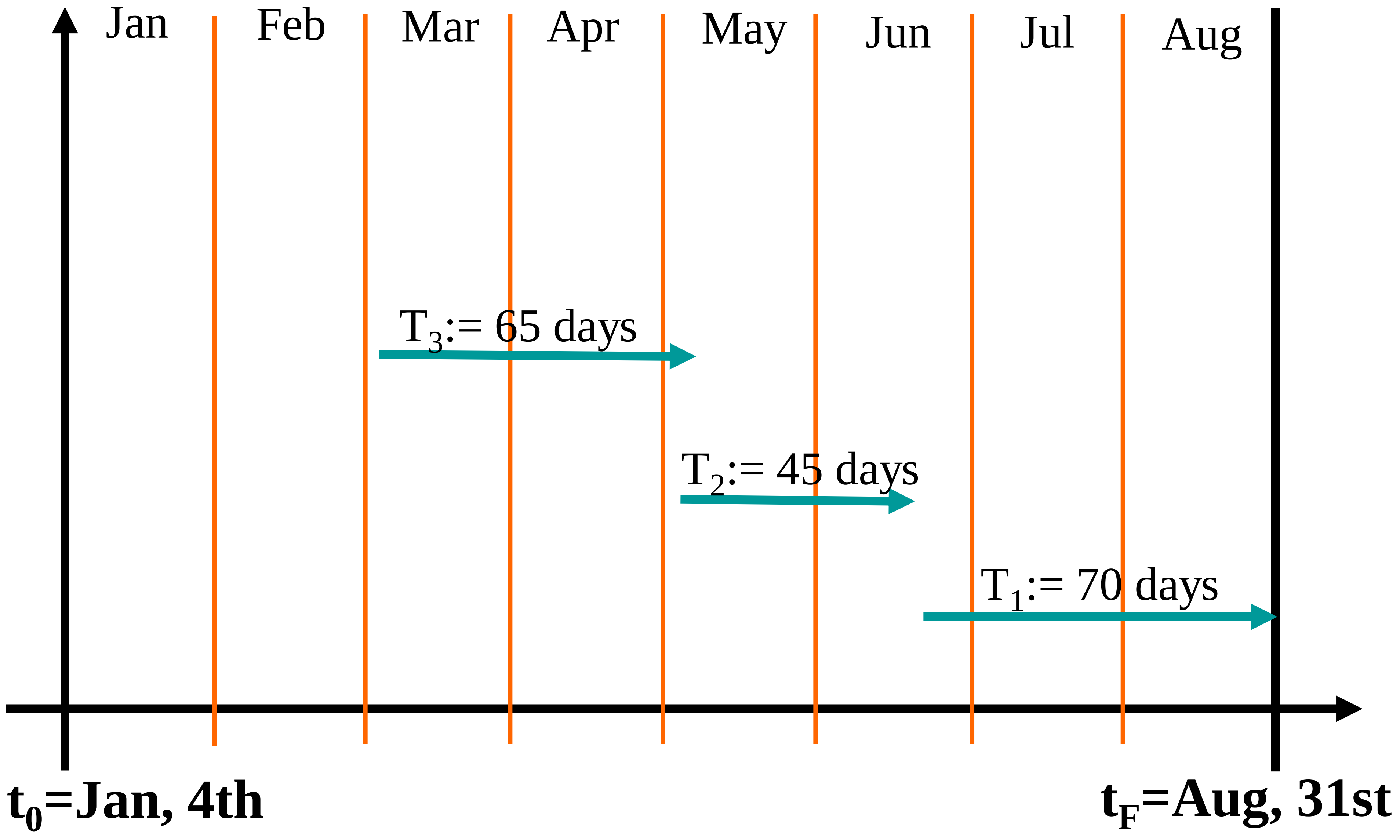
Gantt 4/6


Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
T_2:= 45 days
T_1:= 70 days
t_0=Jan, 4th
t_F=Aug, 31st

Parameters 5/6

	Activity	Days	Milestone	Deliverable
T_4				
T_3	Design	65	May, 8th	Architec.
T_2	Implementation	45	June, 22nd	Code
T_1	V&V	70	August, 31st	Software

Gantt 5/6

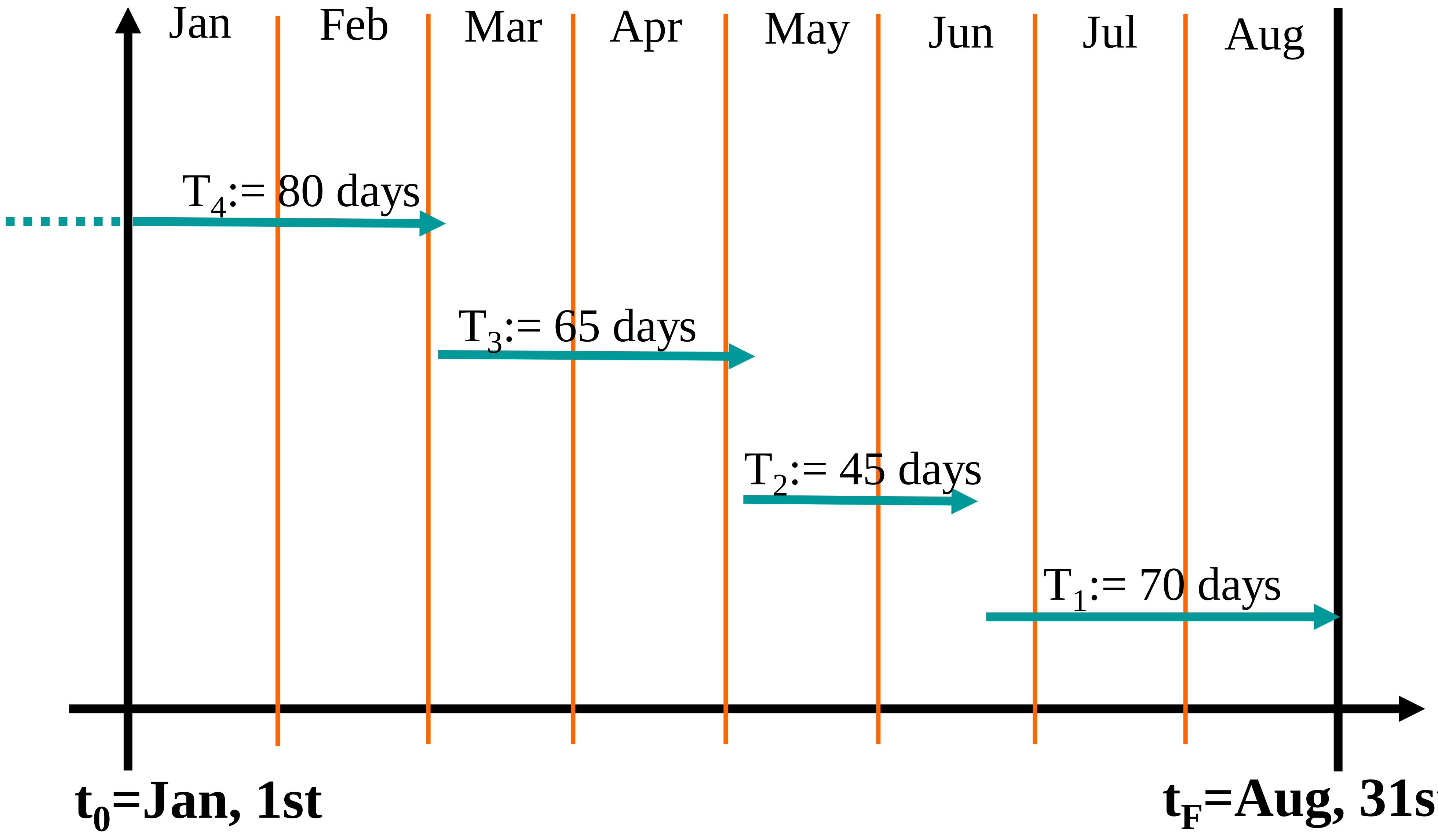

Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
T_3:= 65 days
T_2:= 45 days
T_1:= 70 days
t_0=Jan, 4th
t_F=Aug, 31st

Parameters 6/6

	Activity	Days	Milestone	Deliverable
T_4	Requirements	80	March, 04th	Requirem.
T_3	Design	65	May, 08th	Architec.
T_2	Implementation	45	June, 22nd	Code
T_1	V&V	70	August, 31st	Software

… there are only 60 days between January the 4th and March the 4th!!!

Gantt 6/6


Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
T_4:= 80 days
T_3:= 65 days
T_2:= 45 days
T_1:= 70 days
t_0=Jan, 1st
t_F=Aug, 31st

Solutions

1. End the project 20 days later
2. Start the project 20 days earlier

3. Increase resource allocation (overtime – it does not always work… eXtreme Programming)
4. Assign more resources to certain tasks (it does not always work)
5. Try and break task dependencies (e.g. 80%, …)
6. Make it a parallel process by splitting the system in independent tasks

Solutions

Solution 1: Obvius

Solution 2: Obvius

Solution 3: Obvius

Solution 4: Obvius

Solution 5: Start an activity before the ending of the previous

Independent sub-systems

Solution 6:

Step 1: Find independent sub-systems

Sub-system 1: tracking system

Sub-system 2: billing system interface

Independent sub-systems

Step 2:

Re-plan exploiting parallel development of independent sub-systems.

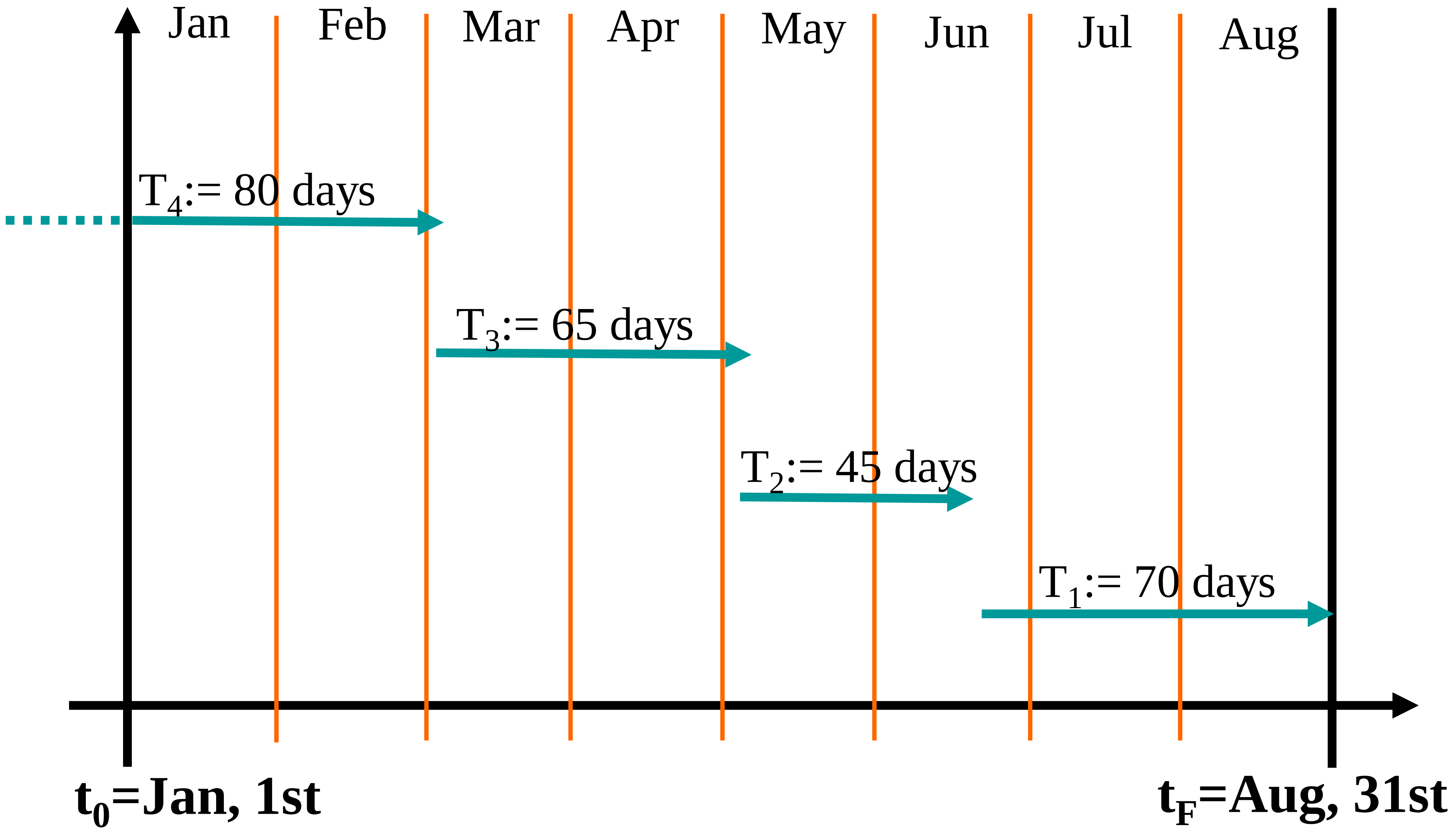
Example:

Implementation of the first subsystem can be performed before the design of the architecture of the second subsystem has ended.

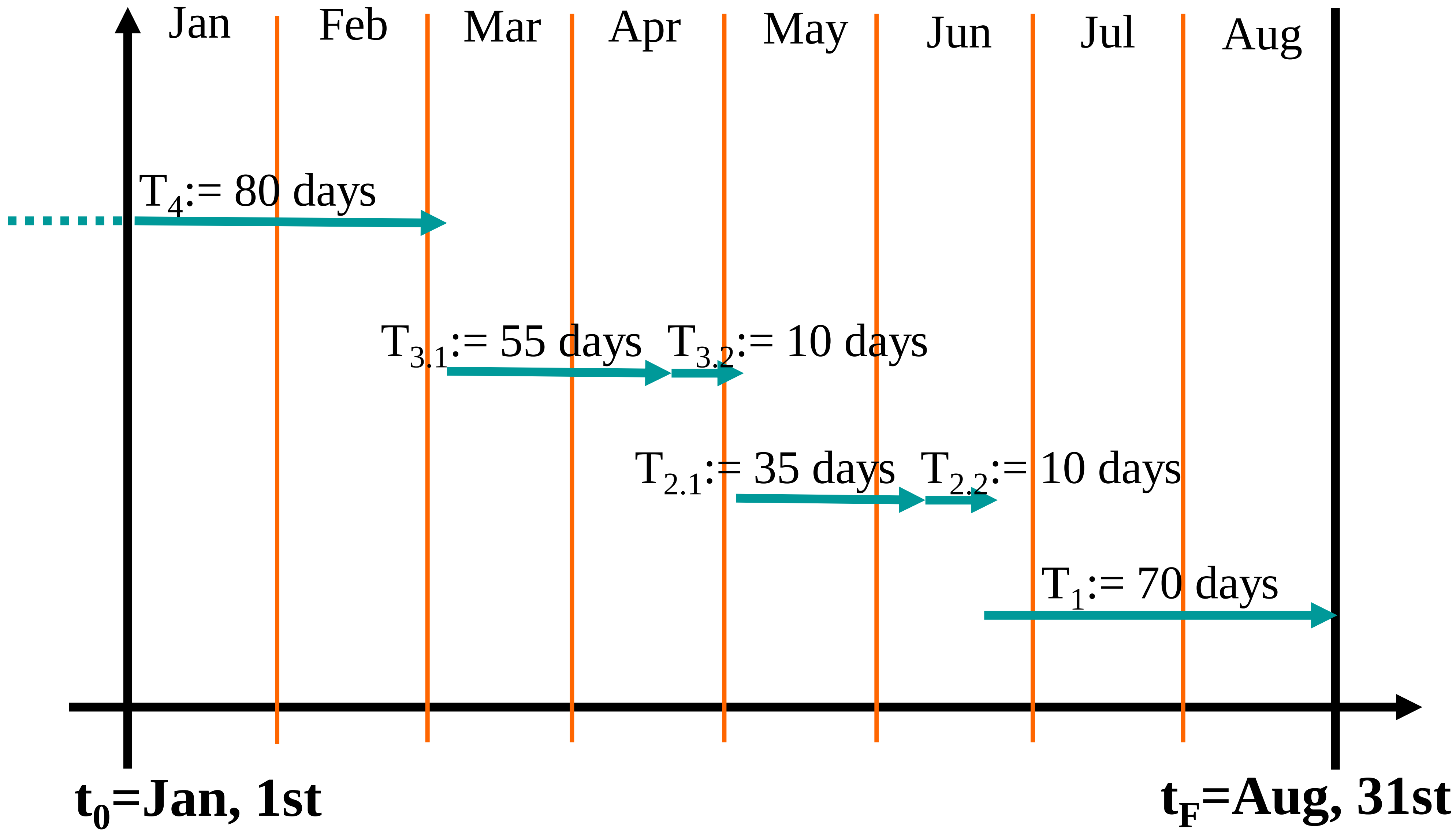
Parameters

	Activity	Days	Milestone	Deliverable
T_4	Requirements	80	March, 24th	Requirem.
$T_{3.2}$	**Subsystem 2 Design**	65	May, 08th	Architec.
$T_{3.1}$	**Subsystem 1 Design**			Architec.
$T_{2.2}$	**Subsystem 2 Impl.**	45	June, 22nd	Code
$T_{2.1}$	**Subsystem 1 Impl.**			Code
T_1	V&V	70	August, 31st	Software

Gantt 6/6

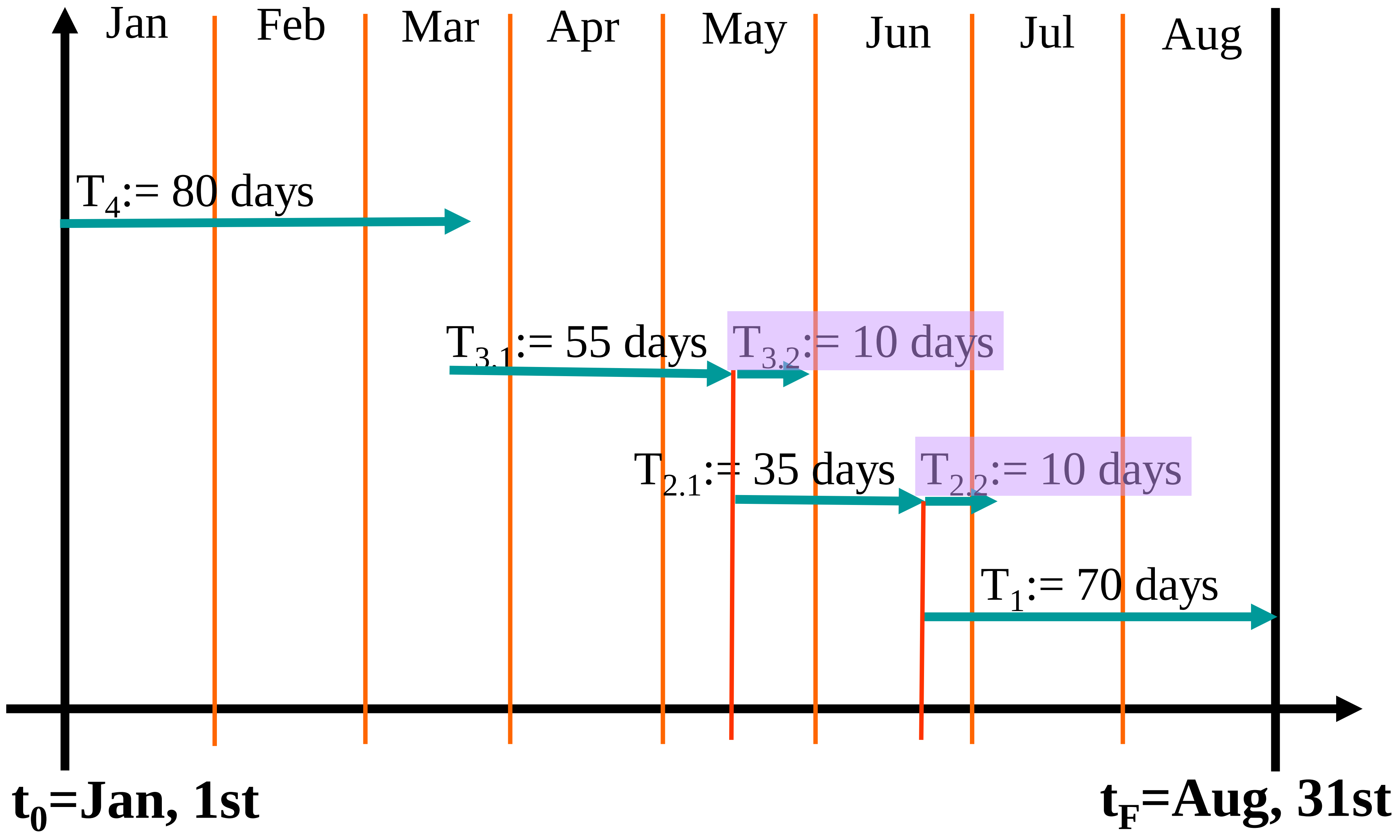

Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
T_4:= 80 days
T_3:= 65 days
T_2:= 45 days
T_1:= 70 days
t_0=Jan, 1st
t_F=Aug, 31st

New Gantt



Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
T_4:= 80 days
$T_{3.1}$:= 55 days
$T_{3.2}$:= 10 days
$T_{2.1}$:= 35 days
$T_{2.2}$:= 10 days
T_1:= 70 days
t_0=Jan, 1st
t_F=Aug, 31st

New Gantt


Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
T_4:= 80 days
$T_{3.1}$:= 55 days
$T_{3.2}$:= 10 days
$T_{2.1}$:= 35 days
$T_{2.2}$:= 10 days
T_1:= 70 days
t_0=Jan, 1st
t_F=Aug, 31st

New Parameters

	Activity	Days	Milestone	Deliverable
T_4	Requirements	80	March, 24th	Requirem.
$T_{3.2}$	**Subsystem 1 Design**	55	May, 18th	Architec.
$T_{3.1}$	**Subsystem 2 Design**	10	May, 28th	Architec.
$T_{2.2}$	**Subsystem 1 Impl.**	35	June, 22nd	Code
$T_{2.1}$	**Subsystem 2 Impl.**	10	July, 02nd	Code
T_1	V&V	70	August, 31st	Software

Resource Allocation

	Activity	From	To
Project Manager	Check the project	January, 04th	August, 31st
Analyst	Requirements	January, 04th	March, 24th
System Architet	Subsystem 1 Design	March, 25rd	May, 18th
System Architet	Subsystem 2 Design	May, 19th	May, 28th
Programmer 1	Subsystem 1 Impl.	May, 19th	June, 22nd
Programmer 2	Subsystem 2 Impl.	June, 23rd	July, 2nd
Programmer 1	V&V	June, 22nd	August, 31st

Software Engineering

Requirement Analysis

September, 28

Requirements Analysis

- **Standard approach:**
  - Use natural language specifications written with a Word Processor (e.g. MS Word)
- **Standard problems:**
  - Difficulties in understanding how the system works
  - Ambiguities/Incompleteness in the specifications
  - Requirements Management (impact of requirements, traceability, …)

Requirements Analysis

- Goal:
  - Define the requirements of the product
- Output
  - Requirements Document with tables and diagrams
- Diagrams for Requirements Analysis:
  - Use Case Diagram
  - Sequence Diagram

Functional and non-functional requir.

Functional requirements:

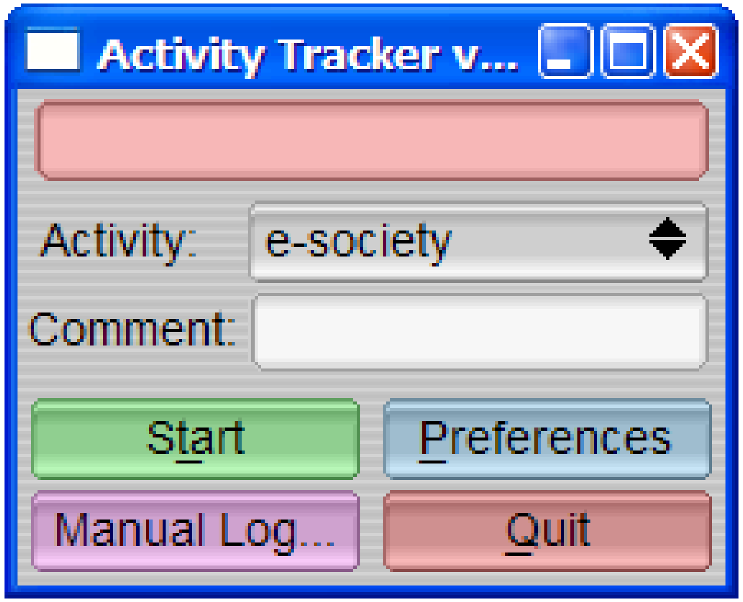
define what a system is supposed to do (functions of the software)

Non-functional requirements:

define how a system is supposed to be

Example: requirements Time Tracker

We have been contacted by a small firm. They want us to build a system for letting employees track how they spend their time when working on a computer. The idea is that of a stop-watch: the users of the system can start and stop counting the time spent on different activities; the system logs such activities and can be used to produce reports.



The system can also be integrated with a billing system. The billing system receives all the information about the time spent by programmers on the different projects and computes the cost of projects. This information is then used to charge clients.

Functional and non-functional requir.

Functional requirement:

- users can start and stop counting the time spent on activities
- the system logs such activities
- logs can be used to produce reports
- …

Non-functional requirement:

- logs need to be updated in real time on a server (or not?)
- this software has to work on Windows, Linux and MacOS
- …

Software Engineering

Overview of UML

UML: What is it and what it is not

Software Engineering is about:

- Process

How you solve problem and deliver a solution

Waterfall, …

- Notations/Languages

How you represent the problem and the solution

Unified Modeling Language, …

- Tools

How you write the problem and deliver the solution

IBM **R**ational **S**oftware **A**rchitect, …

Why UML (and the rest of the story)

- Simple and intuitive
- Support whole cycle
- Widely used
- ISO standard

... and the rest of the story:

UML is not the only specification language

Examples: DFD, SDL, MSC, Statecharts, B, Z.

Why RSA (and the rest of the story)

- Good Support of UML
- More than a drawing program
- Widely used
- We are a IBM-Rational competence center (http://dit.unitn.it/ibm/)

... and the rest of the story:

RSA is not the only tool supporting UML

Examples: Enterprise Architect, Telelogic Tau, Magic Draw UML, Argo UML, Poseidon UML, Dia, Jude, Visual Paradigm, Visio, Visual Studio, ...

UML Tools: a link with some examples

http://www.jeckle.de/umltools.htm

http://en.wikipedia.org/wiki/List_of_UML_tools

UML: Risk and Danger

Notations/Languages are not enough…

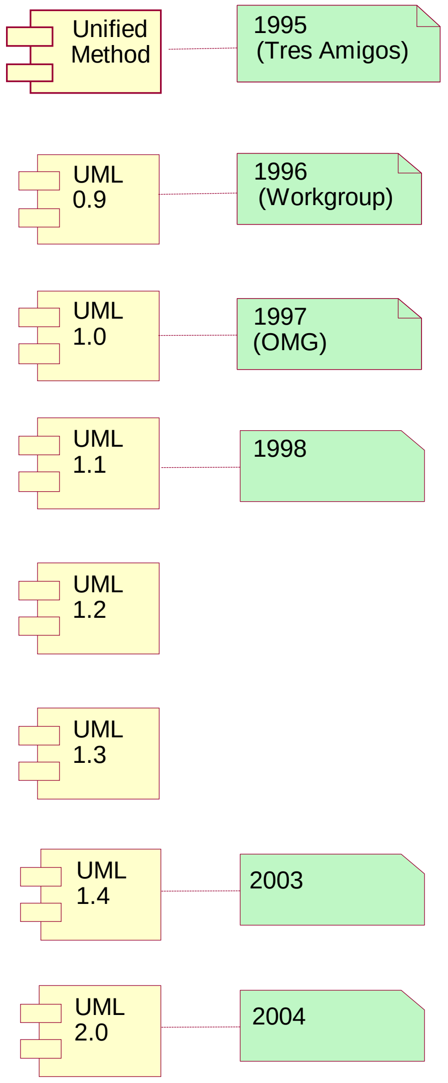
… not even for representing
a problem and its solution.

You need (at least):

- to have a full comprehension of the problem and of the solution
- to be able to communicate such knowledge to other people (diagrams, documents, informal descriptions)

History of UML

- Tres Amigos: Booch, Rumbaugh, Jacobson
- **OMG**: not for profit consortium, 800 members, that produces and maintains standards (e.g. Corba)
- UML Resource Page: http://www.uml.org/



UML Diagrams

- **Static**
  - **Package Diagram**
  - **Use case diagram**
  - **Class diagram and Object diagram**
  - **Component diagram**
  - **Deployment diagram**
- **Dynamic**
  - **Interaction diagrams**
    - **Sequence diagram**
    - **Collaboration diagram**
  - **State diagram**
  - **Activity diagram**

UML Diagrams: Main Uses

- **Static**
  - **Package Diagram**: Help you organize your model
  - **Use case diagram**
    - Business Modeling: Processes
    - Requirements: Functional Requirements
  - **Class diagram and Object diagram**
    - Business Modeling: Organization, Entities
    - Analysis And Design: Logical Architecture

UML Diagrams: Main Uses

- **Static**
  - **Component diagram**
    - Design:
      - Physical Architecture
      - DB Structure
      - Structuring in Files
  - **Deployment diagram**
    - Design: Deployment of Executable in Complex Networks

UML Diagrams

- **Dynamic**
  - **Interaction diagrams**
    - Business Modeling, Requirements: Interaction User-system
    - Analysis And Design: Interaction Among Elements of The System
    - Requirements: Functional Requirements (Sequence Diagram)
  - **State diagram**
    - Analysis and Design: Specification of the Behaviour of the System
  - **Activity diagram**
    - Business Modeling: Workflow
    - Analysis and Design: Specification of the Behaviour of the System

UML Diagrams

- Static
  - Use case diagram
  - Class diagram and Object diagram
  - Component diagram
  - Deployment diagram
- Dynamic
  - Interaction diagrams
    - Sequence diagram
    - Collaboration diagram
  - State diagram
  - Activity diagram

Homework

Make a Gantt Diagram of your student career.

(You can free download GanttProject tool from http://ganttproject.biz)