Software Engineering

Software Configuration

Software Configuration Items

- Project plan
  - the first document
    - states the purpose of the project
  - feasibility study (requirements analysis)
  - tasks, schedules, resources (requirements definition: customer-oriented description of system's functions and operation)

Software Configuration Items

- Requirements specification
  - precise and detailed description of what functionality is required (analysis)
    - primarily from the point of view of the users
  - ought to be complete and consistent
  - includes information of how to verify
  - serves as the basis of a **contract** for the system development

Software Configuration Items

- *A software requirements specification is a document containing a complete description of what the software will do without describing how it will do it* [Alan Davis]

  but:

  one person's how is another person's what

What’s the Correct Answer?

a) 49134

b) 23678

c) -96754

d) 34567

How can you know the answer unless you’ve defined the question?

Questions We Need to Ask

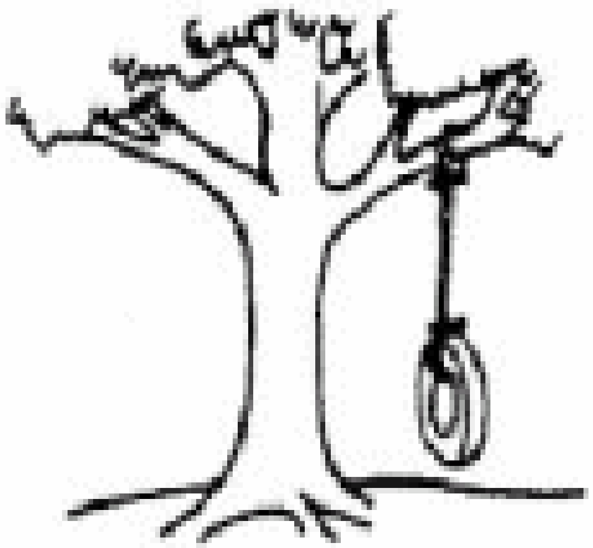
- Why are we doing this task?
- What is this component supposed to do?
- How will we integrate this?
- When can I expect this functionality?
- Where is this request being fulfilled?

What is a Requirement?

- A condition or capability needed by a user to **solve a problem** or achieve an objective
- A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed document

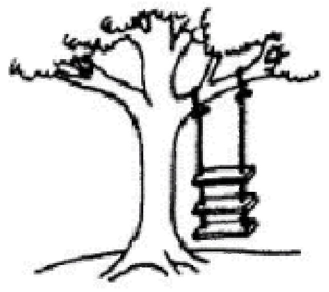
IEEE Standard Glossary of Software Engineering Terminology (1997)

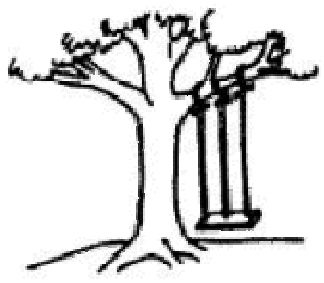
A Simple Example of Requirements Management

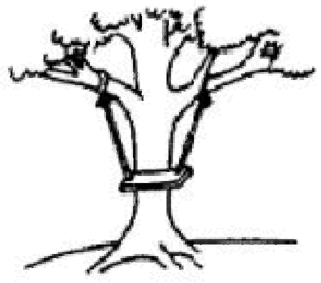


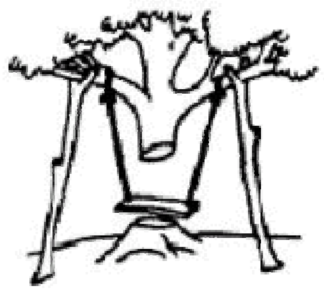
What the customer really needs
What IT really needs to deliver

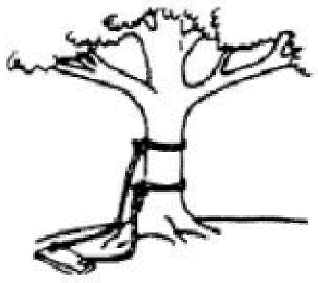
The Communication Challenge

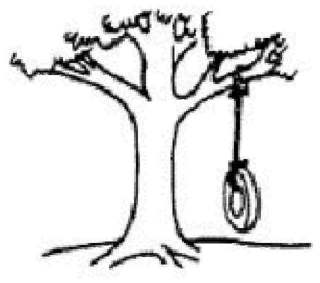

As viewed by Marketing


As viewed by Sales

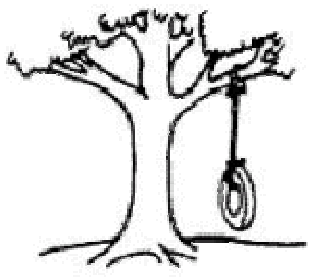

As viewed by IT

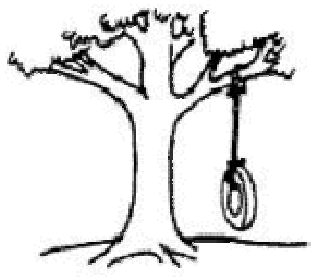

As viewed by Manufacturing

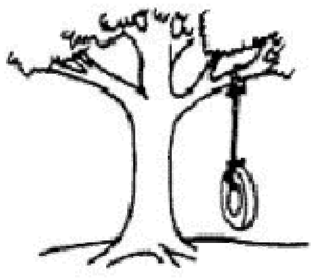

As viewed by Finance

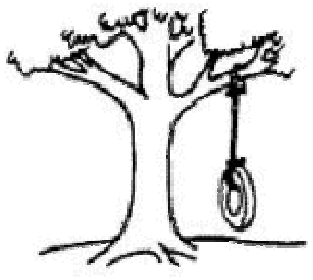

What the business really needs

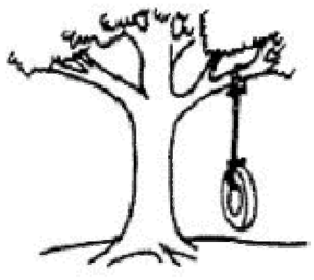
The Goal of Requirements Management

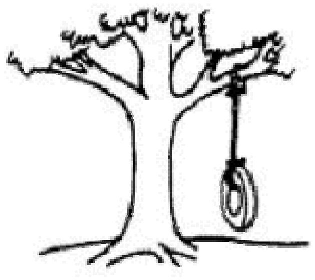

As viewed by
Marketing


As viewed by
Sales


As viewed by
IT


As viewed by
Manufacturing


As viewed by
Finance


What the business
really needs

Software Configuration Items

- Test plan
  - test methods, approaches, procedures (for acceptance, system, integration, and unit tests)

Software Configuration Items

- Data dictionary
  - data, data structures, definition of terms, variables
  - supports design, coding, maintenance
  - presents a shared repository of system information
  - developed during analysis and design

Software Configuration Items

- Design document
  - details behavior and structure (architecture) of the system
  - details the code components of the system
    - data structures, interfaces, algorithms
- User documents
  - user guides, reference guides
- Source code

Software Configuration

- Planning
  - project plan
- Requirement definition
  - requirements specification
  - test plan
  - data dictionary
  - documents

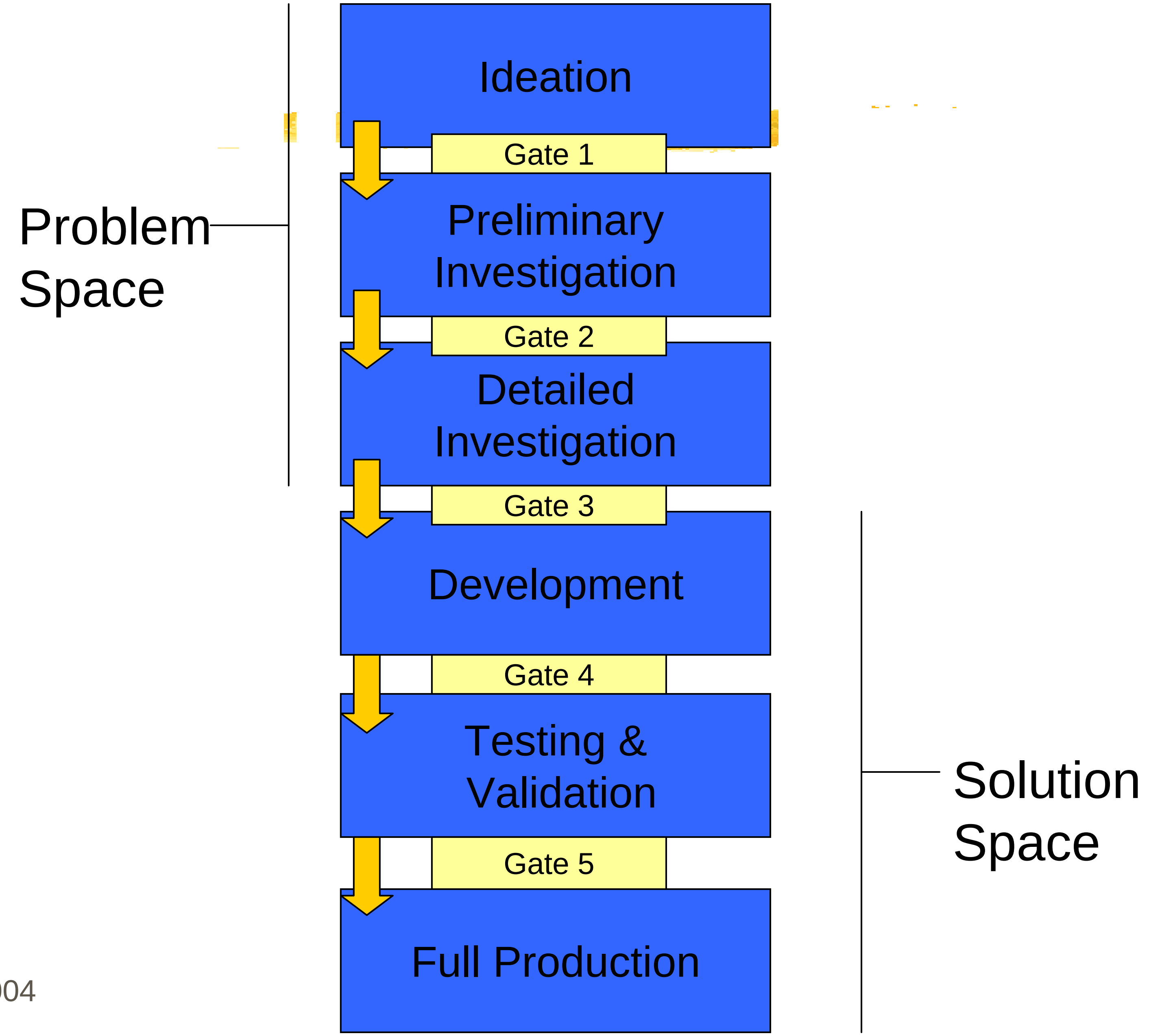
Software Configuration

- Design
  - test plan
- Coding and testing
- Delivery
  - user documents
- Maintenance
  - corrective
  - adaptive
  - perfective
  - preventive

Ideation / Fulfillment Matrix

Interrogations	Deliverable	Plan	Analyze	Design	Build	Test
Why?	**Idea / Need**	Ideation	Preliminary Investigation	Detailed Investigation	Statement of Needs	User Acceptance Test
What?	**Requirement**	Elicitation	Analysis	Decompose/ Derive	Represention of Requirements	Systems Test
How?	**Specification**	Derivation	Tradeoff Analysis	Architect	Specification of Solution	Integration Test
When?	**Task**	Allocation	Estimation / Resourcing	Planning	Management of Plan, Resources, etc.	Project QA
Where?	**Configuration Items**	Solution	Identification	Assignation	Tangible Fulfillment of Need	Unit Test

The Product Development Process



Software Myths

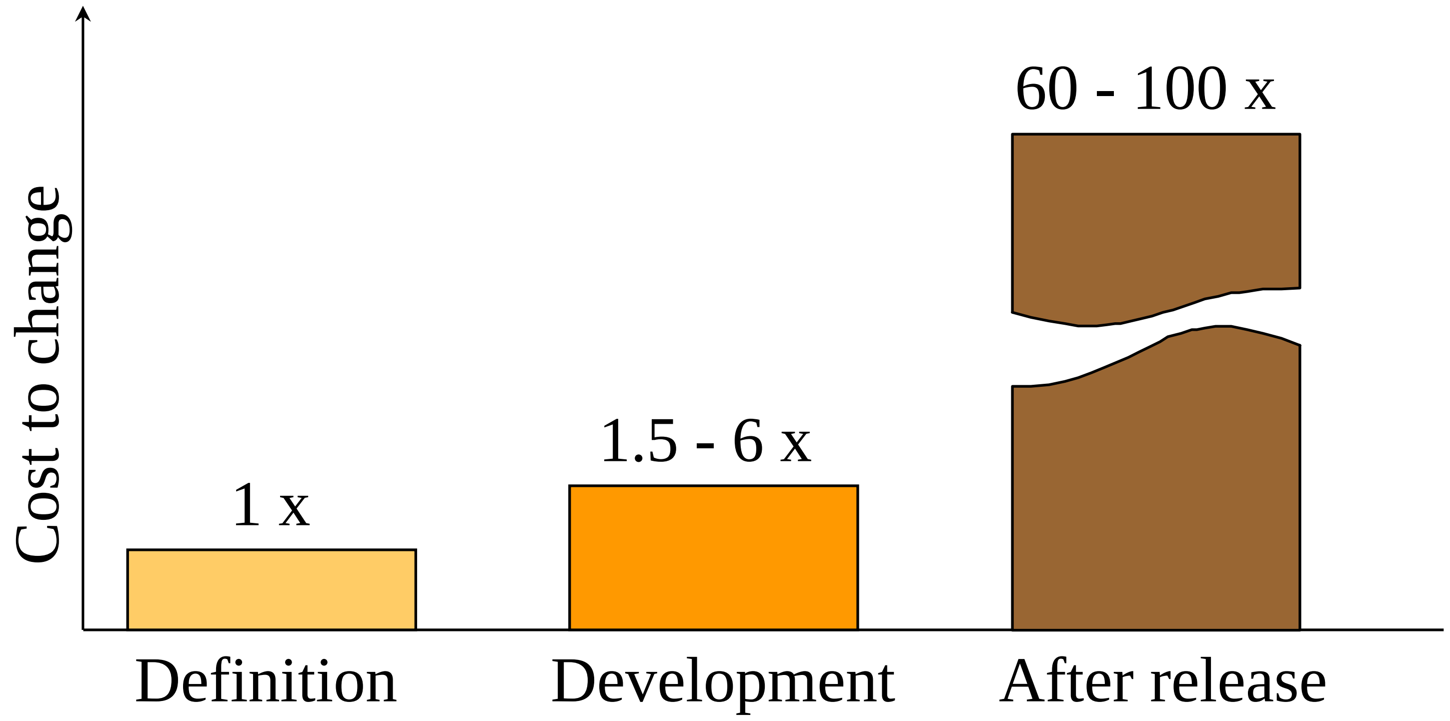
Customer Software Myths

- In order to begin writing programs it is enough to generally state what is wanted (we can fill the details later)
- The truth
  - poor up-front definition is the major cause of failed software efforts
  - thorough communication between customer and developer is needed
  - a formal and detailed statement of function, performance, interfaces, design constraints, and validation criteria is essential

Customer Software Myths

- Software is flexible hence it is easy to change at any stage of software's life
- The truth
  - changes happen as a fact of life
  - late changes are expensive

The Impact of Change



Developer Software Myths

- Job is finished when a program works
- The truth
  - a working program is only one part of a system that includes all elements of a software product
    - recall: what are the additional deliverables?
  - a software life cycle exists: initial work concentrates on planning, subsequent work focuses on development, and ongoing work is required to maintain the software

Developer Software Myths

- The only delivery is a working program
- The truth
  - documentation
    - users
    - maintenance

Manager Software Myths

- When deadline approaches and project gets behind, programmers can be added
- The truth
  - adding manpower to a late software project makes it later
    - training needed
    - integration

Manager Software Myths

- In house tools (state-of-the-art tools) are sufficient
- The truth
  - a fool with a tool is still a fool

Manager Software Myths

- Once software is working, maintenance is minimal and can be done ad hoc
- The truth
  - to the dismay of many managers, over 50% of budget is typically expended on maintenance
  - software maintenance should be organized, planned, and controlled as if it were the largest endeavor within an organization

General Software Myths

- Requirements continually change, but change is easily accommodated since software is malleable
- The truth
  - requirements do change, but the impact of change varies with the type of change and the time at which it is introduced
  - changes requested late in the project may be many times more expensive than the same changes requested earlier in development