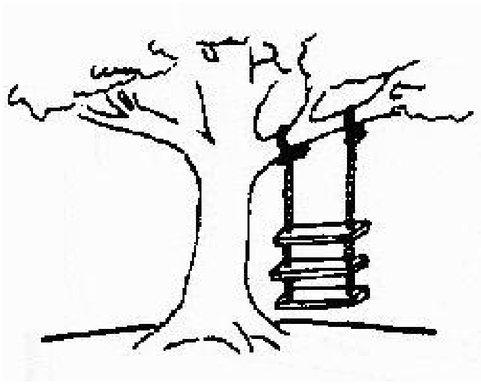
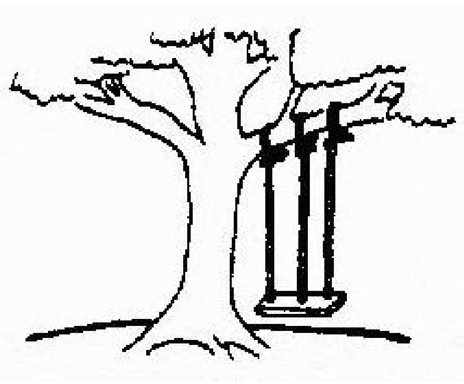
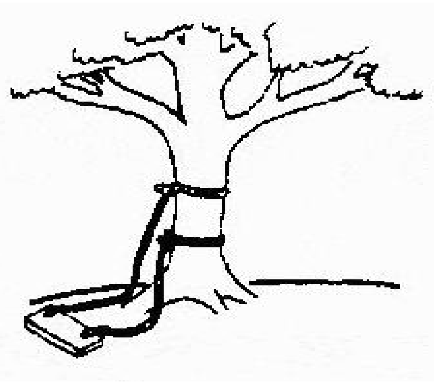
Software Engineering

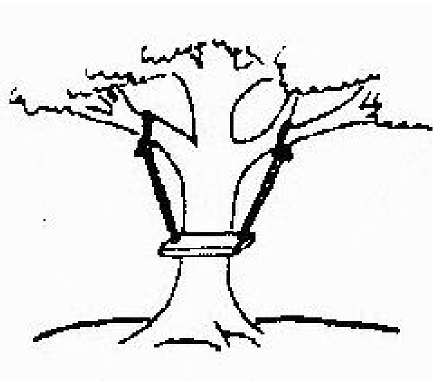
Compiled by Elizabeth George

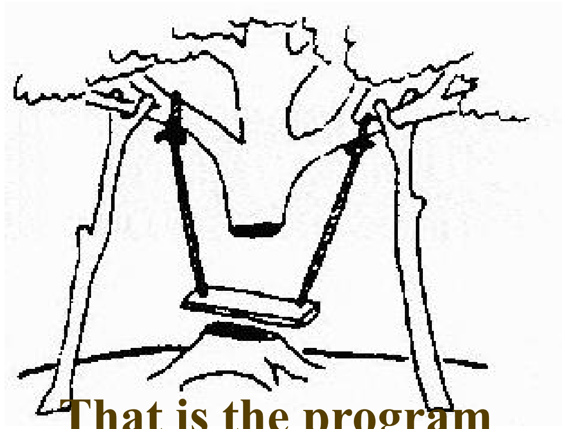
How is Software usually Constructed …

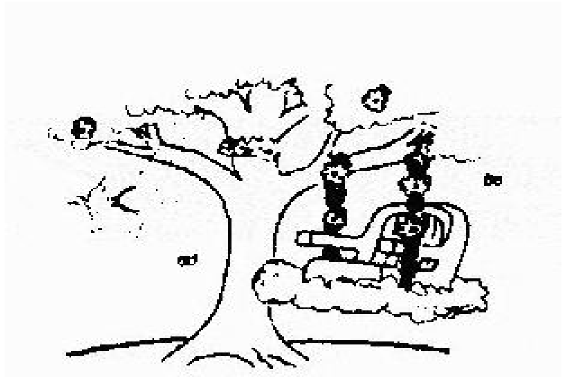

The requirements specification was defined like this

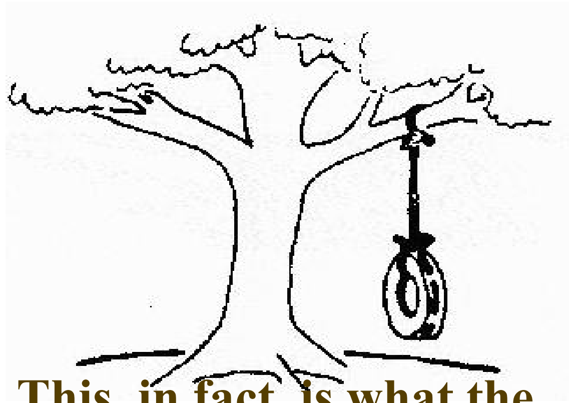

The developers understood it in that way


This is how the problem was solved before.


This is how the problem is solved now


That is the program after debugging


This is how the program is described by marketing dept.


This, in fact, is what the customer wanted …

What is Software?

Software is:

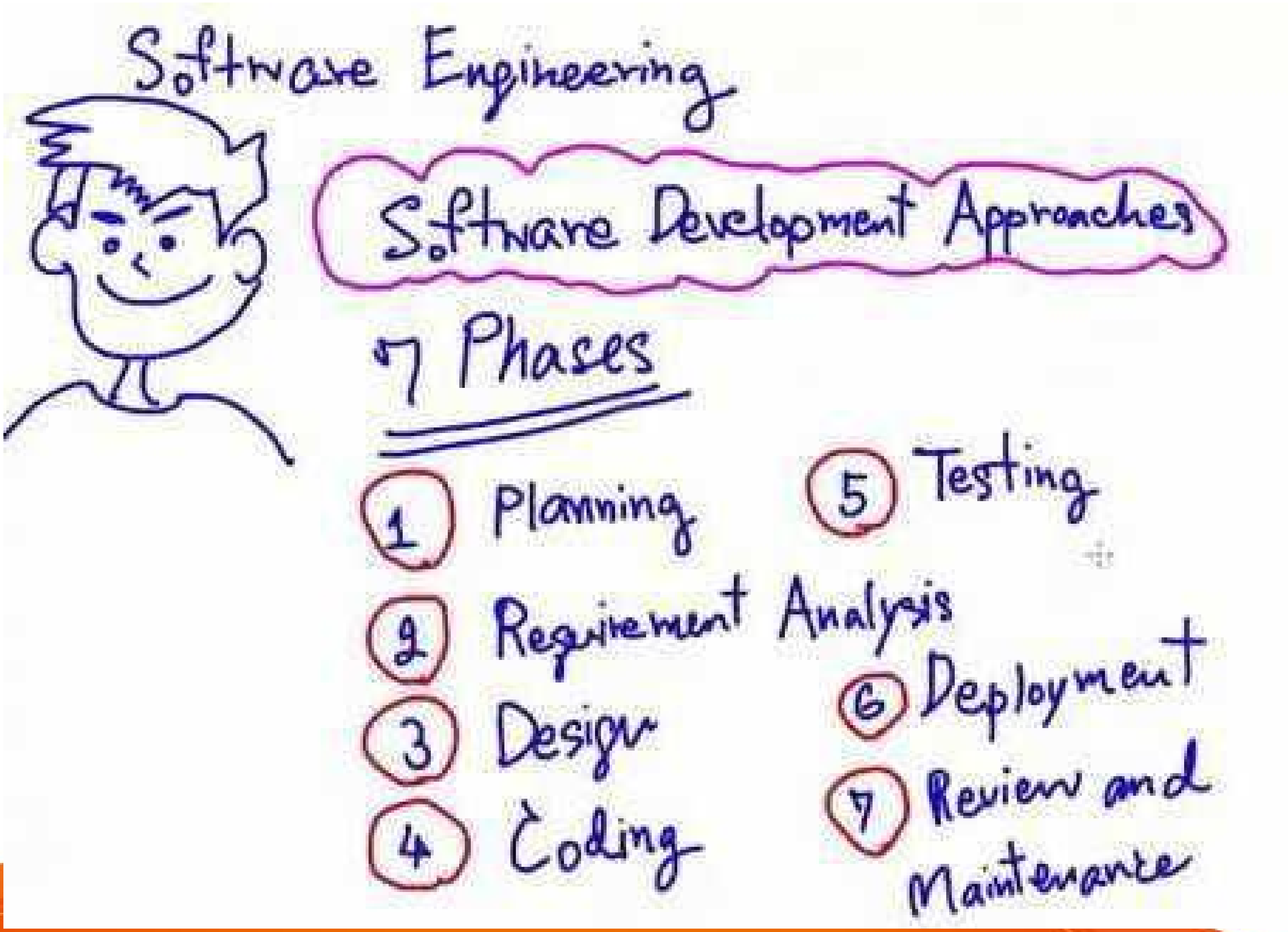
Software = Computer Programs (Instructions) + Procedures + Rules + Associated documents + Data

1. *instructions (computer programs) that when executed provide desired features, function, and performance;*
2. *data structures that enable the programs to adequately manipulate information and*
3. *documentation that describes the operation and use of the programs.*
4. Procedures and rules include step-by-step instructions of how to install (deploy) and use the software.

Cont..

- *Software is developed or engineered, it is not "manufactured "*
- *Software doesn't "wear out." (kharaab nai hota)*
- *Although the industry is moving toward component-based construction, most software continues to be custom-built.*
- *Individual programs perform an operation; collectively they form software and perform a function for the user.*
- *Software is classified into two types based on the areas they command, namely application software and system software.*

Phases in Software Development



Software Engineering

Software Development Approaches

7 Phases

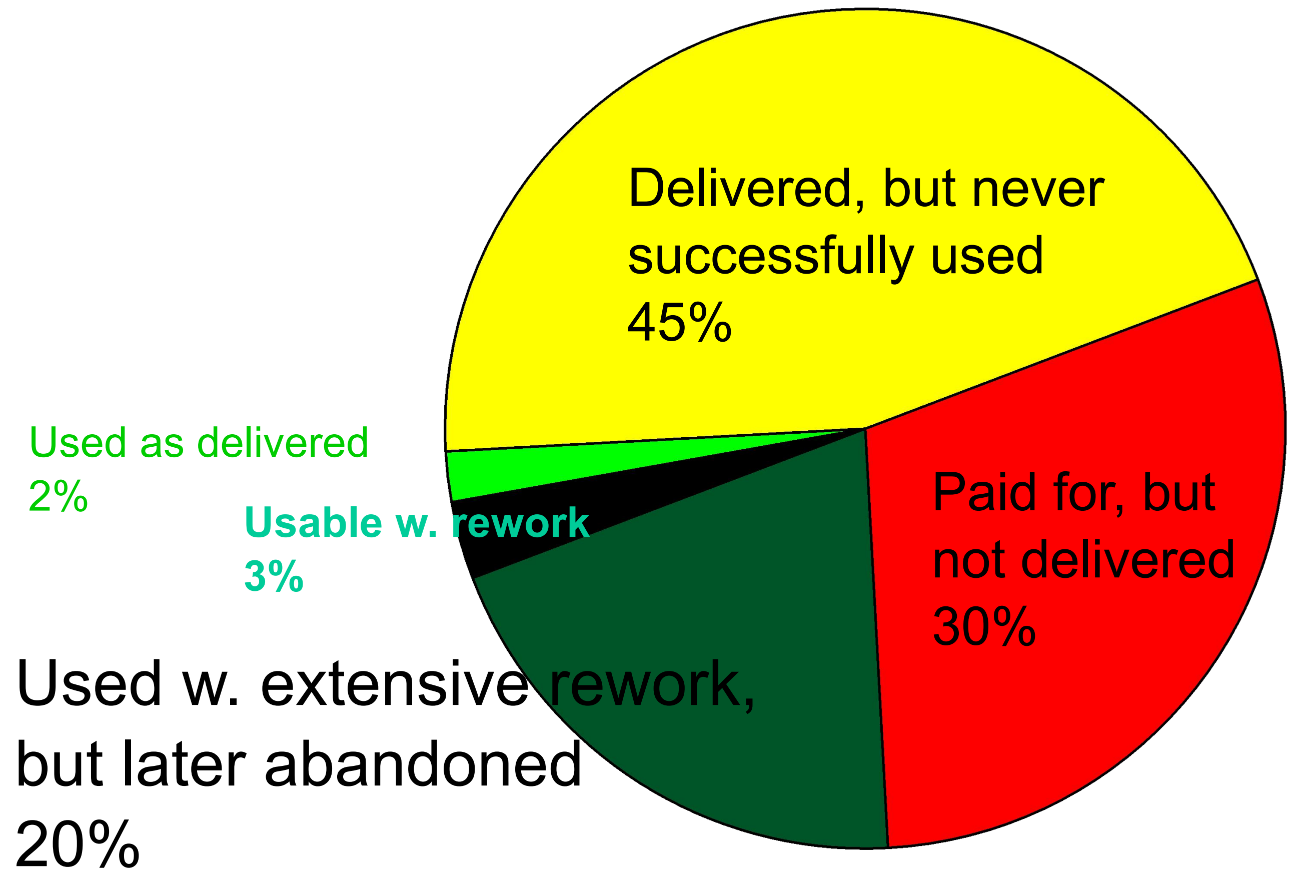
1. Planning
2. Requirement Analysis
3. Design
4. Coding
5. Testing
6. Deployment
7. Review and Maintenance

Software Engineering

- The application of a systematic, disciplined, quantifiable **approach** to the development, operation, and maintenance of software; i.e., the application of engineering to software.
  - IEEE

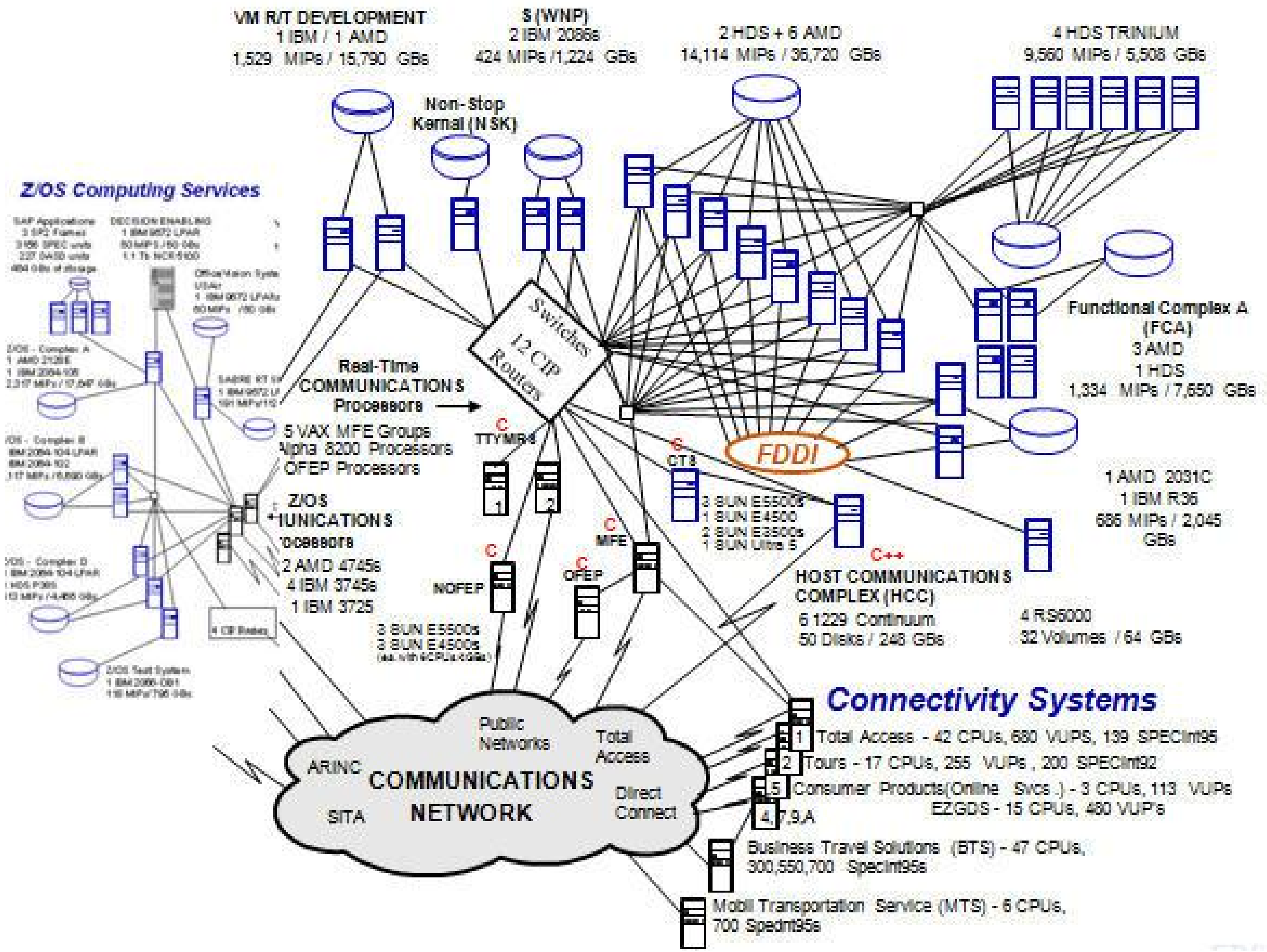
Why Software Engineering?

9 software projects totaling $96.7 million: Where The Money Went [Report to Congress, Comptroller General, 1979]



Take a look at the Standish Report (The "Chaos" Report)

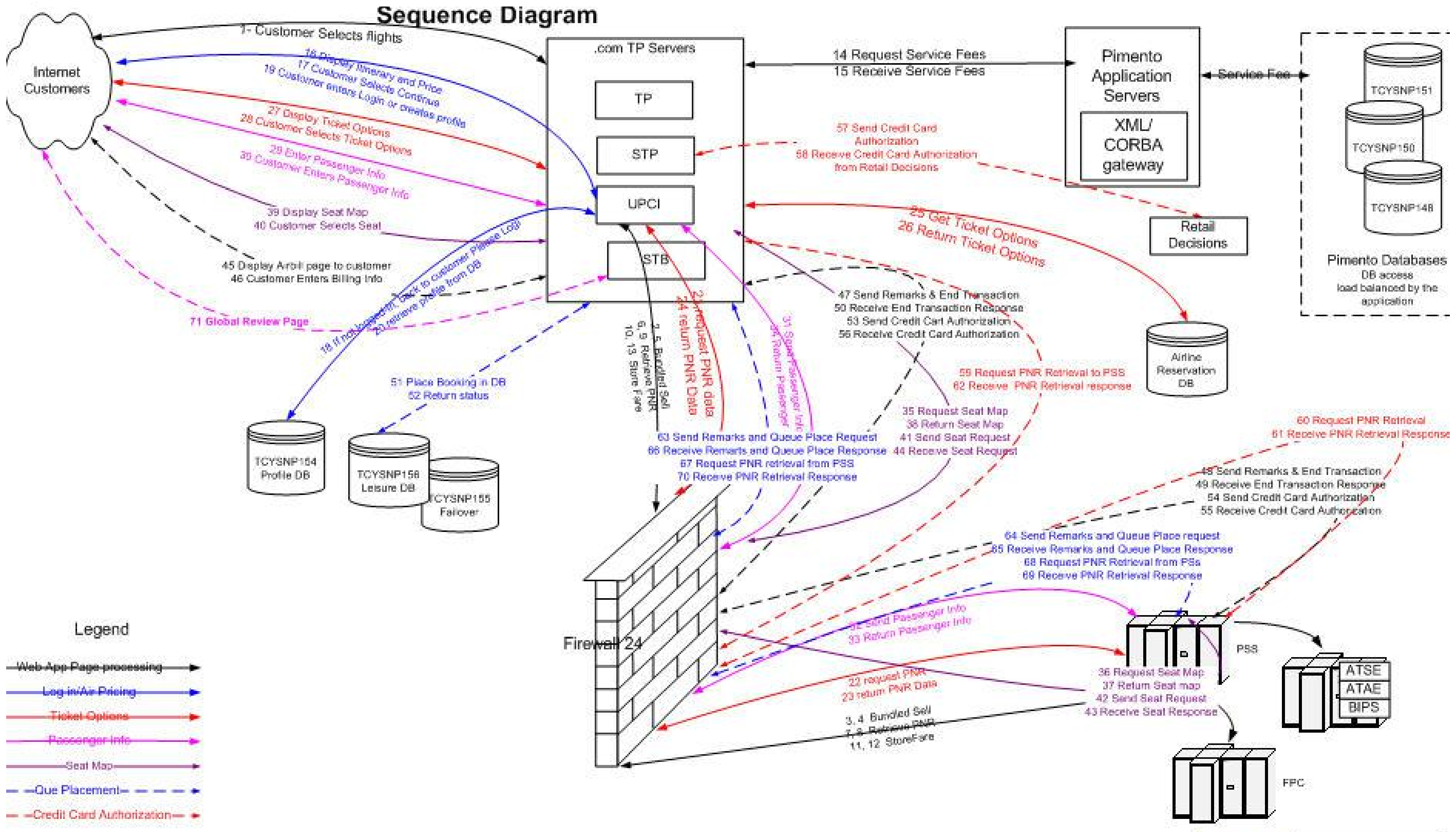
Complex Server Connections



*Factors affecting the **quality** of a software system*

- **Complexity:**
  - The system is so complex that no single programmer can understand it anymore
  - The introduction of one bug fix causes another bug
- **Change:**
  - The “Entropy” of a software system increases with each change: Each implemented change erodes the structure of the system which makes the next change even more expensive (“Second Law of Software Dynamics”).
  - As time goes on, the cost to implement a change will be too.

Complex Message Flow

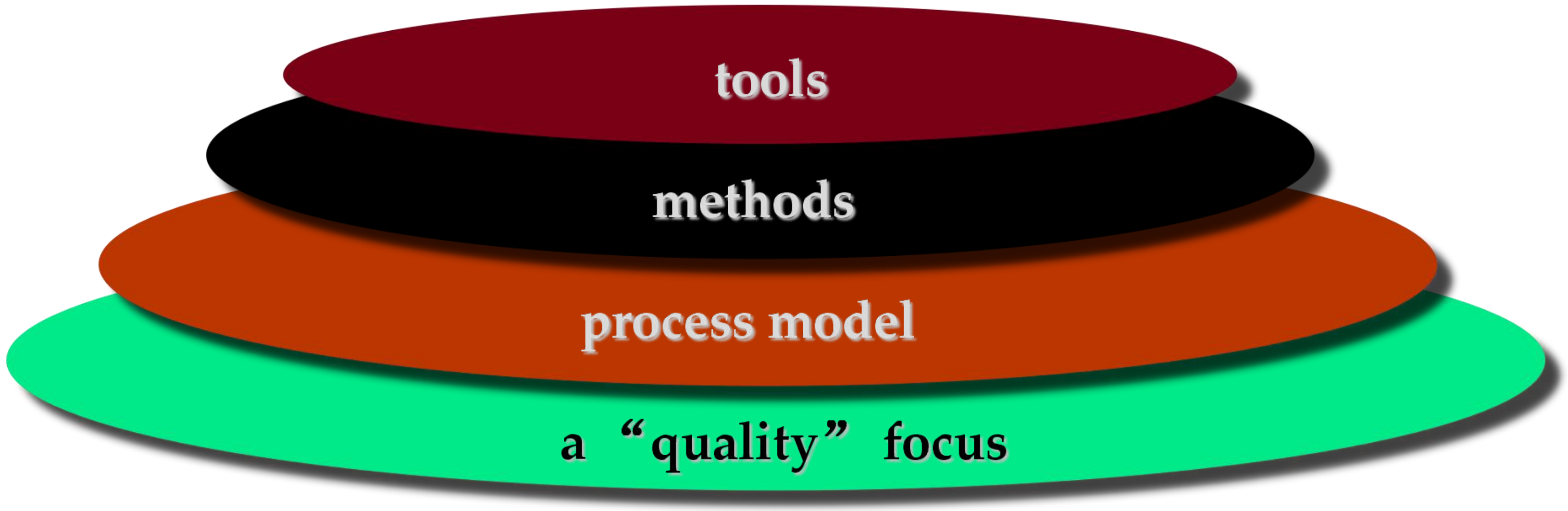


Sequence Diagram
Internet Customers
1- Customer Selects flights
16 Display Itinerary and Price
17 Customer Selects Continue
19 Customer enters Login or creates profile
27 Display Ticket Options
28 Customer Selects Ticket Options
29 Enter Passenger Info
30 Customer Enters Passenger Info
39 Display Seat Map
40 Customer Selects Seat
45 Display Arbil page to customer
46 Customer Enters Billing Info
71 Global Review Page
18 If not logged in, back to customer Please Log
20 retrieve profile from DB
.com TP Servers
TP
STP
UPCI
STB
14 Request Service Fees
15 Receive Service Fees
Pimento Application Servers
XML/ CORBA gateway
Service Fee
TCYSNP151
TCYSNP150
TCYSNP148
Pimento Databases
DB access load balanced by the application
57 Send Credit Card Authorization
58 Receive Credit Card Authorization from Retail Decisions
Retail Decisions
25 Get Ticket Options
26 Return Ticket Options
47 Send Remarks & End Transaction
50 Receive End Transaction Response
53 Send Credit Card Authorization
56 Receive Credit Card Authorization
Airline Reservation DB
59 Request PNR Retrieval to PSS
62 Receive PNR Retrieval response
60 Request PNR Retrieval
61 Receive PNR Retrieval Response
2, 5 Bundled Sell
6, 9 Retrieve PNR
10, 13 Store Fare
21 Request PNR data
24 return PNR Data
31 Send Passenger Info
34 Return Passenger Info
51 Place Booking in DB
52 Return status
TCYSNP154 Profile DB
TCYSNP156 Leisure DB
TCYSNP155 Failover
63 Send Remarks and Queue Place Request
66 Receive Remarks and Queue Place Response
67 Request PNR retrieval from PSS
70 Receive PNR Retrieval Response
35 Request Seat Map
38 Return Seat Map
41 Send Seat Request
44 Receive Seat Request
48 Send Remarks & End Transaction
49 Receive End Transaction Response
54 Send Credit Card Authorization
55 Receive Credit Card Authorization
64 Send Remarks and Queue Place request
65 Receive Remarks and Queue Place Response
68 Request PNR Retrieval from PSs
69 Receive PNR Retrieval Response
32 Send Passenger Info
33 Return Passenger Info
22 request PNR
23 return PNR Data
3, 4 Bundled Sell
7, 8 Retrieve PNR
11, 12 StoreFare
36 Request Seat Map
37 Return Seat map
42 Send Seat Request
43 Receive Seat Response
Firewall 24
PSS
ATSE
ATAE
BIPS
FPC
Legend
Web App Page processing
Log in/Air Pricing
Ticket Options
Passenger Info
Seat Map
Que Placement
Credit Card Authorization

Importance of Software Engineering

- Large software
- Scalability
- Cost
- Dynamic nature
- Quality Management

Software Engineering: A Layered technology



- Any engineering approach must rest on organizational commitment to **quality** which fosters a continuous process improvement culture.
- **Process** layer as the foundation defines a framework with activities fo effective delivery of software engineering technology.
- **Method** provides technical how-to’ s for building software.
- **Tools** provide automated or semi-automated support for the process and methods.

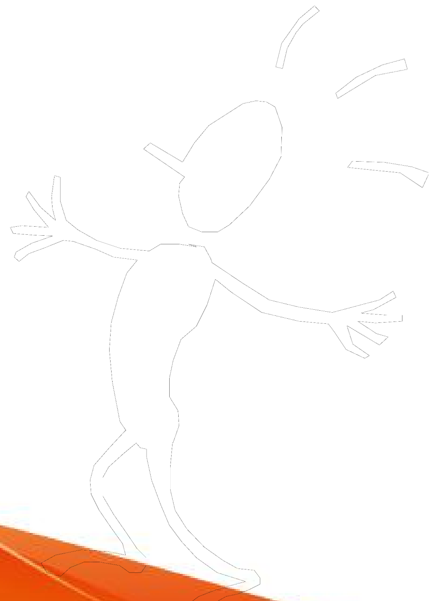
Software Myths (Management Perspectives)

- When my schedule slips, what I have to do is to start a fire-fighting operation: add more software specialists, those with higher skills and longer experience - they will bring the schedule back on the rails!

Software Myths (Management Perspectives

As long as there are good standards and clear procedures in my company, I shouldn't be too concerned.

But the proof of the pudding
is in the eating;
not in the Recipe !



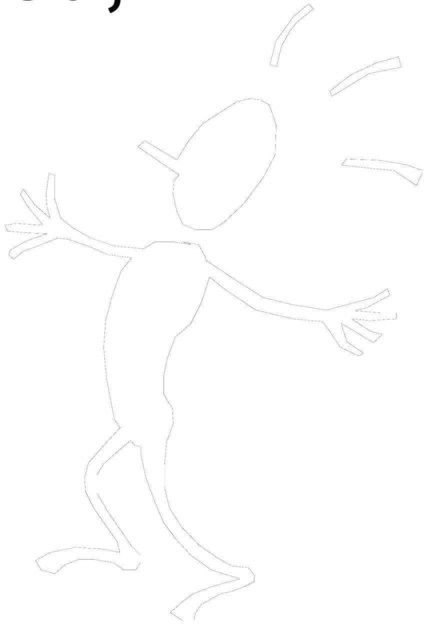
Software Myths (Management Perspectives)

- As long as my software engineers(!) have access to the fastest and the most sophisticated computer environments and state-of-the-art software tools, I shouldn’t be too concerned.

Software Myths (Developer Perspectives)

Once the software is demonstrated, the job is done.

Usually, the problems just begin!



Software Myths
(Developer Perspectives)

- **Until the software is coded and is available for testing, there is no way for assessing its quality.**

Misplaced Assumptions

- All requirements can be pre-specified
- Users are experts at specification of their needs
- Users and developers are both good at visualization
- The project team is capable of unambiguous communication

Ref: Larry Vaughn

Unique Characteristics of Software

- Software is malleable
- Software construction is human-intensive
- Software is intangible and hard to measure
- Software problems are usually complex
- Software directly depends upon the hardware
  - It is at the top of the system engineering “food chain”
- Software doesn’t wear out but will deteriorate
- Software has discontinuous operational nature

Why Study SE?

- It is the basic of any software development
- It helps to produce reliable, reusable and quality software
- Software engineering understands the customer needs and develop the software
- It helps the developers to understand the disciplines of software life cycle
- It helps to develop software in efficient way

Why Study SE?

- Software engineering teaches the best practices of software development
- Software engineering helps to write software, which can be integrated easily with other software

A Process Framework

Process framework

Framework activities

work tasks work products milestones & deliverables QA

Umbrella Activities

Umbrella Activities

Framework Activities

- Communication
- Planning
- Modeling
  - Analysis of requirements
  - Design
- Construction
  - Code generation
  - Testing
- Deployment

Umbrella Activities

- Software project management
- Formal technical reviews
- Software quality assurance
- Software configuration management
- Work product preparation and production
- Reusability management
- Measurement
- Risk management

The Essence of Practice

1. *Understand the problem* (communication and analysis).
2. *Plan a solution* (modeling and software design).
3. *Carry out the plan* (code generation).
4. *Examine the result for accuracy* (testing and quality assurance).

Understand the Problem

- *Who has a stake in the solution to the problem?* That is, who are the stakeholders?
- *What are the unknowns?* What data, functions, and features are required to properly solve the problem?
- *Can the problem be compartmentalized?* Is it possible to represent smaller problems that may be easier to understand?
- *Can the problem be represented graphically?* Can an analysis model be created?

Plan the Solution

- *Have you seen similar problems before?* Are there patterns that are recognizable in a potential solution? Is there existing software that implements the data, functions, and features that are required?
- *Has a similar problem been solved?* If so, are elements of the solution reusable?
- *Can subproblems be defined?* If so, are solutions readily apparent for the subproblems?
- *Can you represent a solution in a manner that leads to effective implementation?* Can a design model be created?

Carry Out the Plan

- *Does the solution conform to the plan?* Is source code traceable to the design model?
- *Is each component part of the solution provably correct?* Has the design and code been reviewed, or better, have correctness proofs been applied to algorithm?

Examine the Result

- *Is it possible to test each component part of the solution?* Has a reasonable testing strategy been implemented?
- *Does the solution produce results that conform to the data, functions, and features that are required?* Has the software been validated against all stakeholder requirements?

Hooker's General Principles

- *The Reason It All Exists*
- *Keep It Simple, Stupid!*
- *Maintain the Vision*
- *What You Produce, Others Will Consume*
- *Be Open to the Future*
- *Plan Ahead for Reuse*
- *Think!*