"Tell me and I forget, teach me and I may remember,
…involve me and I learn"

-Benjamin Franklin

The Software Problem

Atul Gupta

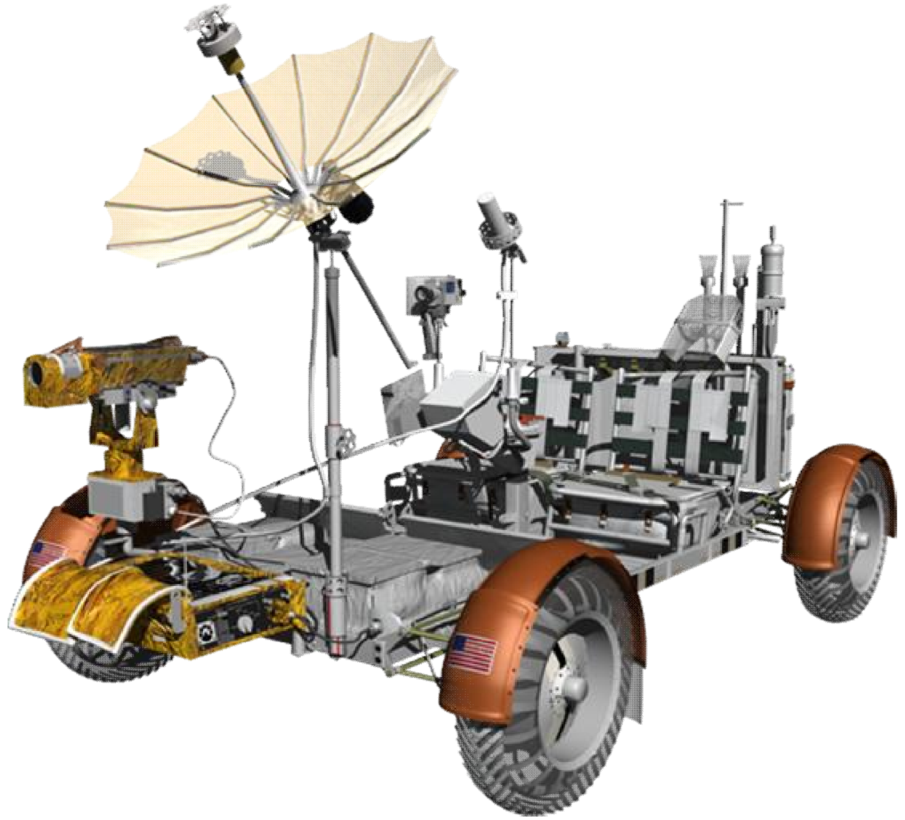
Student’s Software vs. Industrial Software

- Student Software
  - Mostly to demonstrate, no real client/user
  - Mainly coding
  - Business value ?
  - Buggy, Lack of desired quality
  - Ad hoc process followed
  - Not to be maintained or upgraded
- Industrial Software
  - Some definite client/users
  - Coding is a small part of the efforts
  - Business value
  - Have to be of high quality
  - A Systematic, disciplined, and quantifiable process is followed
  - Have to be maintained and upgraded

Software Development in 60's

The moon was promised …..

Software Development in 60's



A lunar rover was built .....

Software Development in 60's

A pair of square wheels were delivered…..

Nature of Software Development: The Past

- IBM survey, 1994
  - 55% of systems cost more than expected
  - 68% overran schedules
  - 88% had to be substantially redesigned
- Advanced Automation System (Federal Aviation Administration, 1982-1994)
  - industry average was $100/line, expected to pay $500/line
  - ended up paying $700-900/line
  - $6B worth of work discarded
- Bureau of Labor Statistics (1997)
  - for every 6 new systems put into operation, 2 cancelled
  - probability of cancellation is about 50% for biggest systems
  - average project overshoots schedule by 50%
  - 3/4 systems were regarded as ‘operating failures’

What’s Wrong?

- Since then, the problem is actively investigated and researched
- Everything has been blamed …
  - Customer
    - What to you mean…I can’t get the moon for $50?
  - ‘Soft’ in the software
    - If I could add one last feature …
  - Processes
  - Young discipline
  - …

What’s Wrong?

Actually, the problem lies in

- Complexity (Lecture #3)
  - The application domain
  - The solution domain
- Change
  - Requirements
  - Changes in technologies
  - Bug fixing

And you need to deliver software…

- Within the budget, the schedule, and user perceived quality…

Magnitude of the Problem: Some Instances

Denver Airport Baggage System

- Approved in 1989
- Aimed to reduce flight delays, shorten waiting times at luggage carousels, and save airlines in labor costs at Denver
- Planned to be operational by the end of 1993
- 53 sq. miles
- 5 runaways, with possibly 12 in future
- 3 landing simultaneously in all weather conditions
- 20 major airlines
- Initial Estimates $186 M
- 4000 telecars carry luggage across 21 miles of track
- Laser scanners read barcodes on luggage tags
- Photocells tracked telecars movement
- Controlled by 300 computers

Denver Airport Baggage System cont.

- Results
  - Crippled by many Software bugs
  - Telecars were misrouted and crashed
  - $1.1 M/day loss due to non operational
  - Airport opened in 1995
  - $88 M spent extra on Baggage system
  - Only one airline used it and that too for outgoing flight
  - Costed $4.2 B
  - A complete failure

Therac-25 (1985-87)

- Radiotherapy machine with software controller
- Software failed to maintain essential invariants: either electron beam mode or stronger beam and plate intervening, to generate X-rays
- Several deaths due to burning
- Programmer had no experience with concurrent programming

  see: *http://sunnyday.mit.edu/therac-25.html*

Ariane-5 (4th June 1996)

- European Space Agency
- Complete loss of unmanned rocket shortly after takeoff
- Failure due to exception thrown in Ada code
- The faulty code was not even needed after takeoff
- It was due to change in physical environment: undocumented assumptions violated
- see: *http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html*

London Ambulance Service (1992)

- loss of calls, double dispatches from duplicate calls
- Media reports - up to 30 people may have died as a result of the chaos.
- poor choice of developer: inadequate experience
- see: *http://www.cs.ucl.ac.uk/staff/A.Finkelstein/las.html*

And the List is endless

- Lost Voyager Spacecraft (one bad line of code caused failure)
- 3 Mile Island in Pennsylvania(poor user interface design)
- Commercial aircraft accidentally shot down during Gulf War (poor user interface)
- …

Sensing the Problem

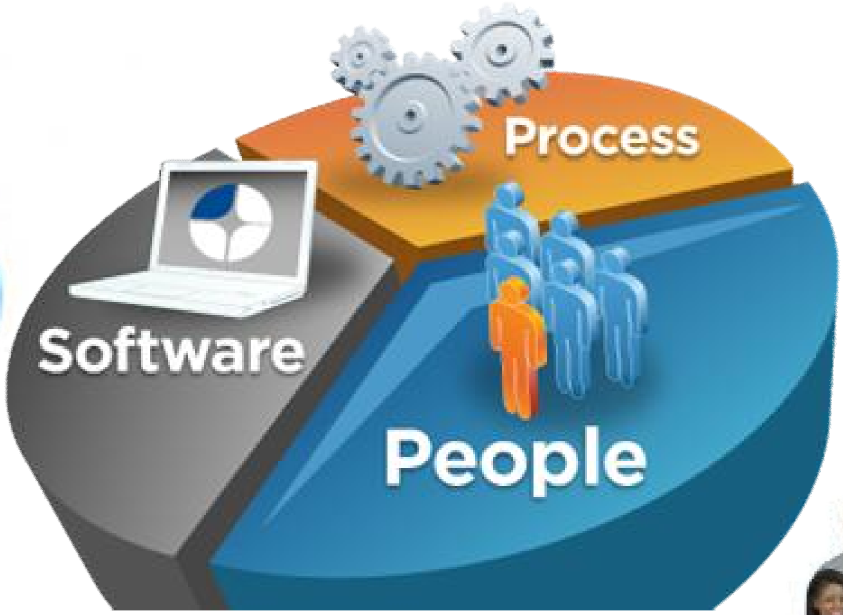
"The demand for software has grown far faster than our ability to produce it. Furthermore, the Nation needs software that is far more usable, reliable, and powerful than what is being produced today. We have become dangerously dependent on large software systems whose behavior is not well understood and which often fail in unpredicted ways."

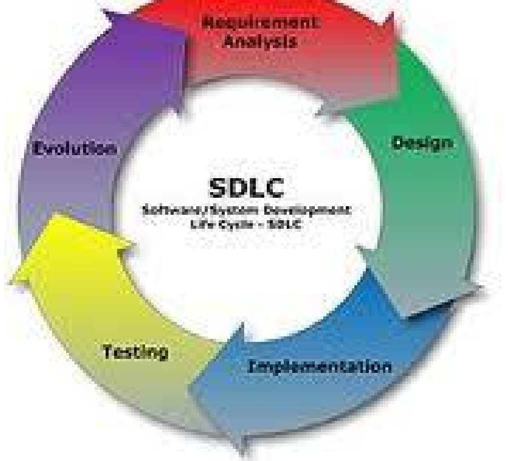
Information Technology Research: Investing in Our Future
President's Information Technology Advisory Committee (PITAC) Report to the President, February 24, 1999

- Available at *http://www.ccic.gov/ac/report/*

The Software Problem

- The **Product**





- The **Process**

- The **People**

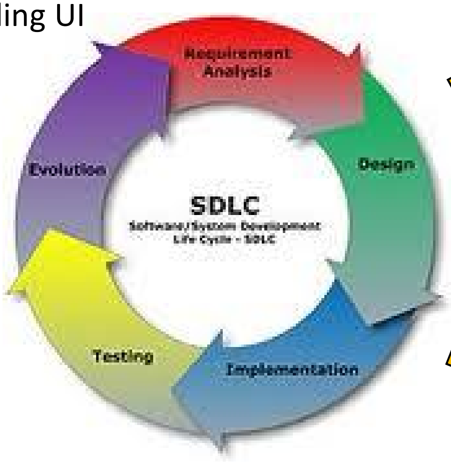
… and managing them all…

The Product

- Software is soft, abstract, not a concrete object like a bridge
- Software is Developed rather than "Manufactured" in the classical sense
- Costs are almost all human
- Easy to fix bugs, but hard to test and validate
- Software typically never wears out...but the hardware/OS platforms it runs on do

The Process

- A phased development
  - Requirements Specification
    - Functional and Non-Functional Req. (Testable)
    - Additional Constraints
    - Interfaces to other systems including UI
  - Analysis Modeling
    - Use cases
    - Interaction Models
    - Object Model (Problem Domain)
  - Design
    - System Design
    - Object Design
  - Construction (Coding)
  - Testing
    - At Different levels – Unit, Integration, System, User
    - Regression Testing
    - Specialized Testing like performance, security, load, Stress testing
  - Operation and Maintenance
    - Deployment
    - Change Management



RUP

XP

SCRUM

The People

- Technical Skills
- Inter-personal-Skills
  - Communication
  - Feedback
  - Courage
  - Motivation
  - Analytical
  - Accountable
  - Discipline
- Team Skills

The People

- Roles
  - Product Manager (Business Analyst)
  - Project Manager (Tech Lead)
  - Product Designer (Technical Architect)
  - Developer
  - QA Person (Tester)

Product Manager (Business Analyst)

The person who makes sure that we build the right product for our users and for company…"

- is going to provide business/market validation for large product and business decisions
- is making sure that what we build has a realistic product and market fit
- comes up with ideas that facilitate the success of our business model through useful features
- mandates creating new, removing old, updating existing features
- is making sure our product strategy has a vision, time-line and plan

Product Designer (Technical Architect)

The Person who make sure the product is actually good, looks great, and works well…”

– works out about how users are actually going to use our features and what those features will look like

– is going to make/take wireframes and turn them into something someone can actually use

– is going to design a product that is visually/functionally comparable to the industry standards/competitors

The People (Project Manager)

The person who make sure that the project we are engineering actually gets successfully finished within all constraints…”

- is making project estimations, to give realistic timelines for launch
- is protecting the time of the engineers, keeping the development train moving and free from endless planning meetings
- is keeping track of the overall progress of the engineering efforts
- is ensuring that the 2 months planned for testing hasn’t been shrunk to 1 week due to unrealistic deadlines
- keeps engineering efforts focused and on task when inevitable distractions arise

The Developer

The Tester

What is Software Engineering?

"Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software."

[IEEE'90]

"Software Engineering is an act of applying a collection of techniques, methodologies and tools that help with the production of a high quality software system, …
with a given budget, before a given deadline,
while change occurs."

[Textbook]

"Software Engineering is intended to mean the best-practice processes used to create and/or maintain software, whether for groups or individuals."

[SEYP]

What is high quality software?

- It must be useful (to the original customer)
- It must be portable (work at all of the customer’s sites)
- It must be maintainable
- It must be secure
- It must have high integrity (produces correct results, with a high degree of accuracy).
- It must be efficient
- It must have consistency of function (it does what the user would, reasonably expect it to do)
- It must have good human engineering (easy to learn and easy to use)
- It must be error-free

“By failing to prepare, you are preparing to fail”.

"When you're finished changing, you're finished.“

Either write something worth reading or do something worth writing.

“It is the first responsibility of every citizen to question authority.”

“I am for doing good to the poor, but...I think the best way of doing good to the poor, is not making them easy in poverty, but leading or driving them out of it. I observed...that the more public provisions were made for the poor, the less they provided for themselves, and of course became poorer. And, on the contrary, the less was done for them, the more they did for themselves, and became richer.”
— Benjamin Franklin

“We must all hang together, or assuredly we shall all hang separately.”
— Benjamin Franklin