CS 61C: Great Ideas in Computer Architecture (Machine Structures) Course Introduction

Instructors:
Krste Asanovic
Randy H. Katz
http://inst.eecs.Berkeley.edu/~cs61c/F12

Agenda

- Great Ideas in Computer Architecture
- Administrivia
- PostPC Era: From Phones to Datacenters

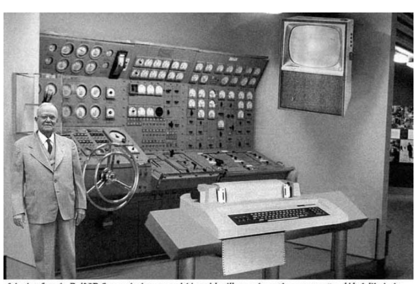
Agenda

- Great Ideas in Computer Architecture
- Administrivia
- PostPC Era: From Phones to Datacenters

CS61c is NOT really about C Programming

- It is about the hardware-software interface
  - What does the programmer need to know to achieve the highest possible performance
- Languages like C are closer to the underlying hardware, unlike languages like Scheme!
  - Allows us to talk about key hardware features in higher level terms
  - Allows programmer to explicitly harness underlying hardware parallelism for high performance

Old School CS61c

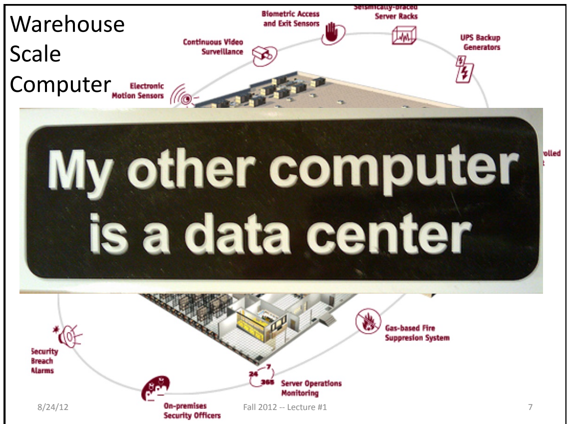


Scientists from the RAND Corporation have created this model to illustrate how a "home computer" could look like in the year 2004. However the needed technology will not be economically feasible for the average home. Also the scientists readily admit that the computer will require not yet invented technology to actually work, but 50 years from now scientific progress is expected to solve these problems. With teletype interface and the Fortran language, the computer will be easy to use.

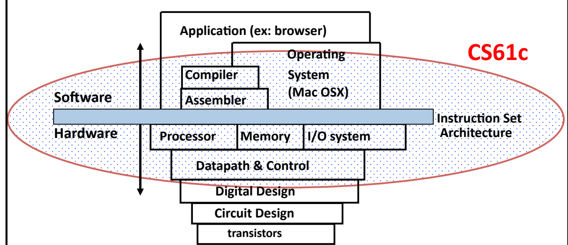
Personal Mobile Devices

New School CS61c

Warehouse Scale Computer



Old-School Machine Structures

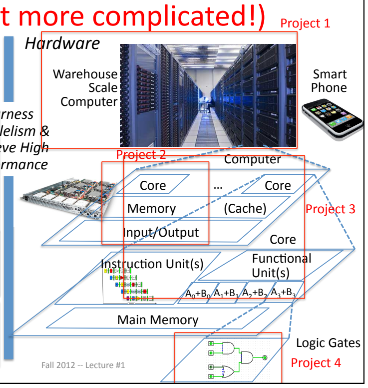


New-School Machine Structures (It's a bit more complicated!)

Software

- Parallel Requests
  Assigned to computer
  e.g., Search "Katz"
- Parallel Threads
  Assigned to core
  e.g., Lookup, Ads
- Parallel Instructions
  >1 instruction @ one time
  e.g., 5 pipelined instructions
- Parallel Data
  >1 data item @ one time
  e.g., Add of 4 pairs of words
- Hardware descriptions
  All gates functioning in parallel at same time

Harness Parallelism & Achieve High Performance



6 Great Ideas in Computer Architecture

1. Layers of Representation/Interpretation
2. Moore's Law
3. Principle of Locality/Memory Hierarchy
4. Parallelism
5. Performance Measurement & Improvement
6. Dependability via Redundancy

Great Idea #1: Levels of Representation/Interpretation

High Level Language Program (e.g., C)

```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;
```

Compiler

Assembly Language Program (e.g., MIPS)

```
lw   $t0, 0($2)
lw   $t1, 4($2)
sw   $t1, 0($2)
sw   $t0, 4($2)
```

Anything can be represented as a *number*, i.e., data or instructions

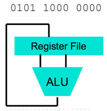
Assembler

Machine Language Program (MIPS)

```
0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111
```

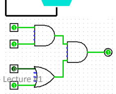
Machine Interpretation

Hardware Architecture Description (e.g., block diagrams)



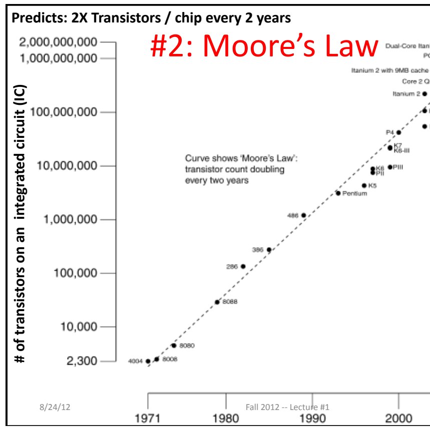
Architecture Implementation

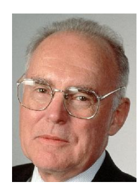
Logic Circuit Description (Circuit Schematic Diagrams)



Predicts: 2X Transistors / chip every 2 years

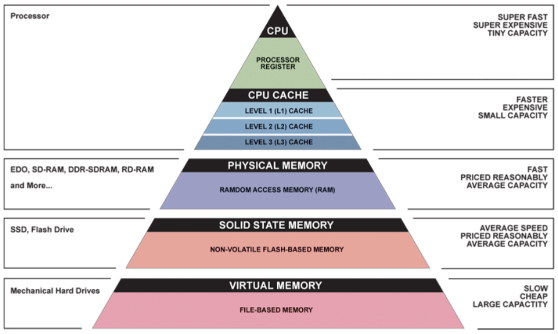
#2: Moore's Law



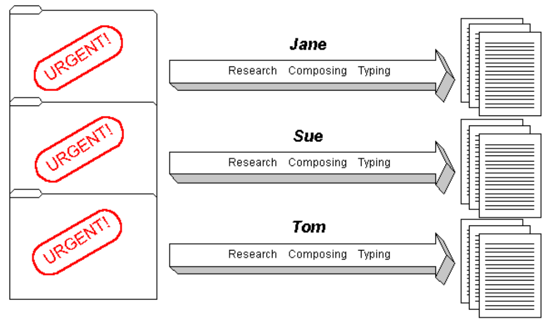


Gordon Moore
Intel Cofounder
B.S. Cal 1950!

Great Idea #3: Principle of Locality/ Memory Hierarchy



Great Idea #4: Parallelism



Great Idea #5: Performance Measurement and Improvement

- Matching application to underlying hardware to exploit:
  - Locality
  - Parallelism
  - Special hardware features, like specialized instructions (e.g., matrix manipulation)
- Latency
  - How long to set the problem up
  - How much faster does it execute once it gets going
  - It is all about *time to finish*

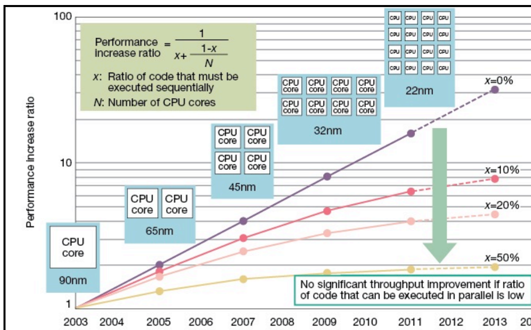
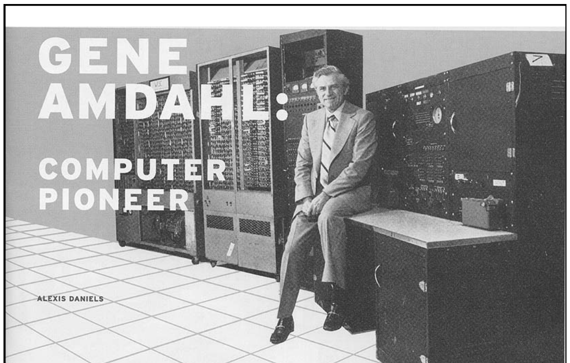
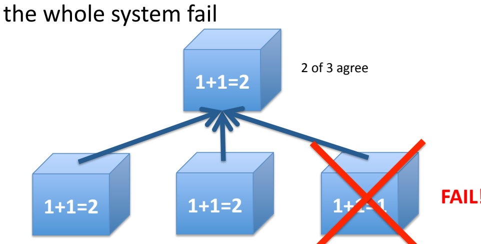


Fig 3 Amdahl's Law an Obstacle to Improved Performance Performance will not rise in the same proportion as the increase in CPU cores. Performance gains are limited by the ratio of software processing that must be executed sequentially. Amdahl's Law is a major obstacle in boosting multicore microprocessor performance. Diagram assumes no overhead in parallel processing. Years shown for design rules based on Intel planned and actual technology. Core count assumed to double for each rule generation.



Great Idea #6: Dependability via Redundancy

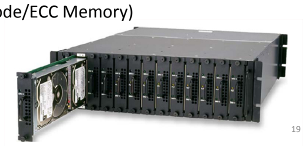
- Redundancy so that a failing piece doesn't make the whole system fail



Increasing transistor density reduces the cost of redundancy

Great Idea #6: Dependability via Redundancy

- Applies to everything from datacenters to storage to memory
  - Redundant datacenters so that can lose 1 datacenter but Internet service stays online
  - Redundant disks so that can lose 1 disk but not lose data (Redundant Arrays of Independent Disks/RAID)
  - Redundant memory bits of so that can lose 1 bit but no data (Error Correcting Code/ECC Memory)

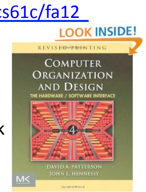


Agenda

- Great Ideas in Computer Architecture
- Administrivia
- From Phones to Datacenters

Course Information

- Course Web: http://inst.eecs.Berkeley.edu/~cs61c/fa12
- Instructors: Krste Asanovic, Randy Katz
- Teaching Assistants:
  - Alan Christopher (Head TA), Loc Do, James Ferguson, Anirudh Garg, William Ku, Brandon Luong, Ravi Punj, Sung Roa Yoon
- Textbooks: Average 15 pages of reading/week
  - Barroso & Holzle (B&H): *The Datacenter as a Computer (free download from web page)*
  - Patterson & Hennessey (P&H): *Computer Organization and Design*, Revised 4th Edition (not ≤3rd Edition, not Asian 4th edition)
  - Kernighan & Ritchie (K&R): *The C Programming Language*, 2nd Edition
- Piazza for class announcements, Q&A:
  - Just go to Piazza web page and add yourself to the class
  - Staff reads them all; please keep it class related and professional



Course Information

- The Good News:
  - Will accommodate as many people as possible
  - Should be able to add 2 to 3 new lab sections
    - Wednesday afternoon and evening
    - 64-96 waitlisters!
- The Bad News:
  - Still sorting through TA schedules
- Will be posted on Piazza (hopefully this weekend)

Reminders

- Labs start next week
  - Part of first lab is discussion relevant to first HW
  - Switching Sections: if you find another 61C student willing to swap discussion AND lab, talk to your TAs
  - Project Partners: only Project 3 and extra credit, OK if partners mix sections but have same TA
- First homework assignment due 2 September by 11:59:59 PM
  - Reading assignment on course page

Course Organization

- Grading
  - Participation and Altruism (5%)
  - Homework (5%)
  - Labs (20%)
  - Projects (40%)
    1. Data Parallelism (Map-Reduce on Amazon EC2)
    2. Computer Instruction Set Simulator (C)
    3. Performance Tuning of a Parallel Application involving Matrix Calculations using cache blocking, SIMD, MIMD (OpenMP, work with partner)
    4. Computer Processor Design (Logisim)
  - Extra Credit: Matrix Calculation Competition, anything goes
  - Midterm (10%): 8-10 PM Tuesday October 9
  - Final (20%): 1130-230PM Monday December 10 (conflicts with EECS 40 – we are bigger, so their problem!)

EECS Grading Policy

- http://www.eecs.berkeley.edu/Policies/ugrad.grading.shtml
  "A typical GPA for courses in the lower division is 2.7. This GPA would result, for example, from 17% A's, 50% B's, 20% C's, 10% D's, and 3% F's. A class whose GPA falls outside the range 2.5 - 2.9 should be considered atypical."
- Spring 2011: GPA 2.85
  24% A's, 49% B's, 18% C's,
  6% D's, 3% F's
- Job/Intern Interviews: They grill you with technical questions, so it's what you say, not your GPA
  (61c gives you good stuff to say)

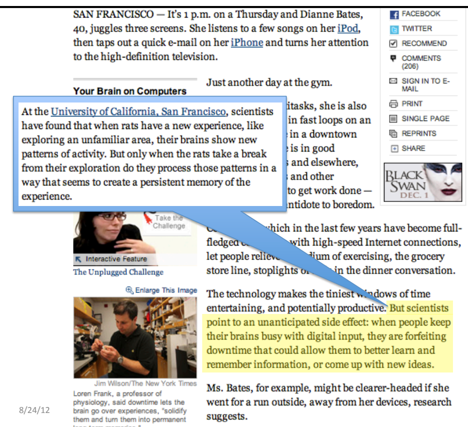
	Fall	Spring
2011	2.72	2.85
2010	2.81	2.81
2009	2.71	2.81
2008	2.95	2.74
2007	2.67	2.76

Late Policy

- Assignments due Sundays at 11:59:59 PM
- Late homeworks not accepted (100% penalty)
- Late projects get 20% penalty, accepted up to Tuesdays at 11:59:59 PM
  - No credit if more than 48 hours late
  - No "slip days" in 61C
    - Used by Dan Garcia and a few faculty to cope with 100s of students who often procrastinate without having to hear the excuses, but not widespread in EECS courses
    - More late assignments if everyone has no-cost options; better to learn now how to cope with real deadlines

Policy on Assignments and Independent Work

- With the exception of laboratories and assignments that explicitly permit you to work in groups, all homeworks and projects are to be YOUR work and your work ALONE.
- You are encouraged to discuss your assignments with other students, and extra credit will be assigned to students who help others, particularly by answering questions on Piazza, but we expect that what you hand is yours.
- It is NOT acceptable to copy solutions from other students.
- It is NOT acceptable to copy (or start your) solutions from the Web.
- We have tools and methods, developed over many years, for detecting this. You WILL be caught, and the penalties WILL be severe.
- At the minimum a ZERO for the assignment, possibly an F in the course, and a letter to your university record documenting the incidence of cheating.
- (People are caught every semester!)


SAN FRANCISCO — It's 1 p.m. on a Thursday and Dianne Bates, 40, juggles three screens. She listens to a few songs on her iPod, then taps out a quick e-mail on her iPhone and turns her attention to the high-definition television.

Just another day at the gym.

Your Brain on Computers

At the University of California, San Francisco, scientists have found that when rats have a new experience, like exploring an unfamiliar area, their brains show new patterns of activity. But only when the rats take a break from their exploration do they process those patterns in a way that seems to create a persistent memory of the experience.

...itasks, she is also ... in fast loops on an ... in a downtown ... is in good ... and elsewhere, ... and other ... to get work done — ...ntidote to boredom.

...which in the last few years have become full-fledged ... with high-speed Internet connections, let people relieve ...dium of exercising, the grocery store line, stoplights or ... in the dinner conversation.

The technology makes the tiniest windows of time entertaining, and potentially productive. But scientists point to an unanticipated side effect: when people keep their brains busy with digital input, they are forfeiting downtime that could allow them to better learn and remember information, or come up with new ideas.

Ms. Bates, for example, might be clearer-headed if she went for a run outside, away from her devices, research suggests.

Interactive Feature

The Unplugged Challenge

Enlarge This Image

Jim Wilson/The New York Times

Loren Frank, a professor of physiology, said downtime lets the brain go over experiences, "solidify them and turn them into permanent

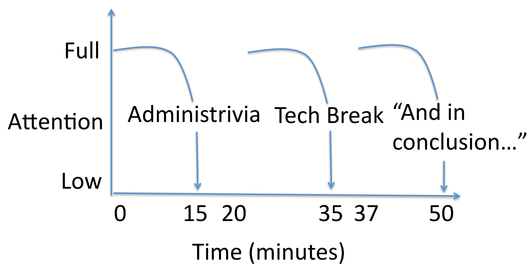
FACEBOOK
TWITTER
RECOMMEND
COMMENTS (206)
SIGN IN TO E-MAIL
PRINT
SINGLE PAGE
REPRINTS
SHARE

What are the Six Great Ideas in Computer Architecture?

1.
2.
3.
4.
5.
6.

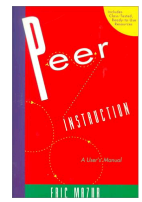
The Rules (and we really mean it!)

Architecture of a Lecture



Peer Instruction

- Increase real-time learning in lecture, test understanding of concepts vs. details
  mazur-www.harvard.edu/education/pi.phtml
- As complete a "segment" ask multiple choice question
  - <1 minute: decide yourself, vote
  - <2 minutes: discuss in pairs, then team vote; flash card to pick answer
    - Try to convince partner; learn by teaching
- Mark and save flash cards (get in discussion section)



Question: Which statements are TRUE about this class?

- ☐ The midterm is Wednesday October 10 during class (11-noon)
- ☐ The midterm is Tuesday October 9 in the evening (8-10 PM)
- ☐ We will accommodate EECS 40 students with a special late final examination sitting
- ☐ I can save money by buying Asian edition of Computer Organization and Design

CS61c in the News

Computer Science Becomes Cool Major Again

Susan Hall | CHARTING YOUR IT CAREER | 31 JUL, 2012

Share

Computer science has become the most popular major at Stanford, part of a swell of popularity of the discipline over the past three years. **U.S. News & World Report** points to the **myriad opportunities** in an otherwise tight job market, but then there's also the cool factor.

education program, told **Siliconvalley.com**:

"Today's students have grown up using many computing technologies, including Web search engines, social networks and smartphones."
The computer science major "affords them the opportunity to go from being consumers of computing technology to producers of it, and that's a tremendously powerful transition."

Stanford, of course, occupies a stellar reputation for training computer nerds as well as entrepreneurs. Meanwhile, **the University of Washington has been gaining distinction** in that arena as well with the rising demand for tech talent in the Seattle area, according to a **New York Times** story.

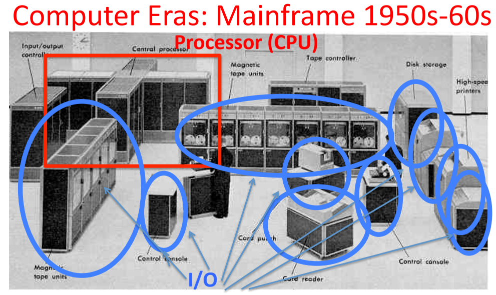
Professor Ed Lazowska, however, notes that despite the demand for engineers, the university turns away about three-quarters of the students who apply because it lacks the capacity to teach that many, a problem mentioned in a **2011 report** by the Computing Research Association, which tracks trends.

Unlike many majors, computer science students can launch high-paying careers with just an undergraduate degree. Enrollment in computer science started dropping after the dot-com bust, reaching its lowest point in 2005 before starting its rebound.

Agenda

- Great Ideas in Computer Architecture
- Administrivia
- PostPC Era: From Phones to Datacenters

Computer Eras: Mainframe 1950s-60s



"Big Iron": IBM, UNIVAC, … build $1M computers for businesses => timesharing OS (Multics)

Minicomputer Eras: 1970s

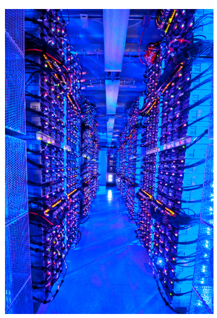
Using integrated circuits, Digital, HP… build $10k computers for labs, universities => UNIX OS

PC Era: Mid 1980s - Mid 2000s

Using microprocessors, Apple, IBM, … build $1k computers for individuals => Windows OS, Linux

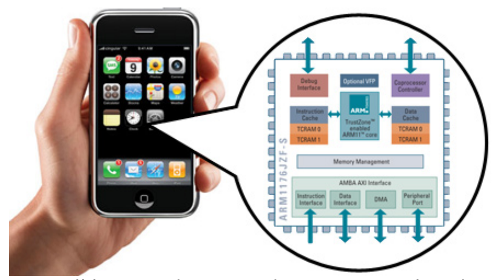
PostPC Era: Late 2000s - ??

Personal Mobile Devices (PMD): Relying on wireless networking, Apple, Nokia, … build $500 smartphone and tablet computers for individuals => Android OS



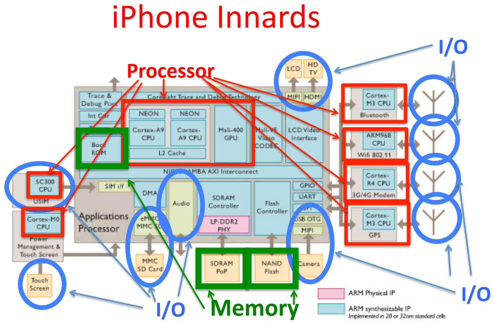
Cloud Computing: Using Local Area Networks, Amazon, Google, … build $200M Warehouse Scale Computers with 100,000 servers for Internet Services for PMDs => MapReduce/Hadoop

Advanced RISC Machine (ARM) instruction set inside the iPhone



You will how to design and program a related RISC computer: MIPS

iPhone Innards



You will about multiple processors, data level parallelism, caches in 61C

Why Not 80x86 vs. MIPS?

- Once learn one, easy to pick up others
- 80x86 instruction set is not beautiful
  - ≈ Full suitcase then add clothes on way to plane
  - Class time precious; why spend on minutiae?
- MIPS represents energy efficient processor of client (PostPC era) vs. fast processor of desktop (PC era)
- MIPS represents more popular instruction set: 2010: 6.1B ARM, 0.3B 80x86 (20X more)

Software as a Service: SaaS

- Traditional SW: binary code installed and runs wholly on client device
- SaaS delivers SW & data as service over Internet via thin program (e.g., browser) running on client device
  - Search, social networking, video
- Now also SaaS version of traditional SW
  - E.g., Microsoft Office 365, TurboTax Online

8/24/12 Fall 2012 -- Lecture #1 43

6 Reasons for SaaS

1. No install worries about HW capability, OS
2. No worries about data loss (at remote site)
3. Easy for groups to interact with same data
4. If data is large or changed frequently, simpler to keep 1 copy at central site
5. 1 copy of SW, controlled HW environment => no compatibility hassles for developers
6. 1 copy => simplifies upgrades for developers *and* no user upgrade requests

8/24/12 Fall 2012 -- Lecture #1 44

SaaS Infrastructure?

- SaaS demands on infrastructure

1. Communication: allow customers to interact with service
2. Scalability: fluctuations in demand during + new services to add users rapidly
3. Dependability: service and communication continuously available 24x7

8/24/12 Fall 2012 -- Lecture #1 46

Clusters

- Clusters: Commodity computers connected by commodity Ethernet switches

1. More scalable than conventional servers
2. Much cheaper than conventional servers
   - 20X for equivalent vs. largest servers
3. Few operators for 1000s servers
   - Careful selection of identical HW/SW
   - Virtual Machine Monitors simplify operation
4. Dependability via extensive redundancy

8/23/12 Fall 2012 -- Lecture #1 47

The Big Switch: Cloud Computing

"A hundred years ago, companies stopped generating their own power with steam engines and dynamos and plugged into the newly built electric grid. The cheap power pumped out by electric utilities didn't just change how businesses operate. It set off a chain reaction of economic and social transformations that brought the modern world into existence. Today, a similar revolution is under way. Hooked up to the Internet's global computing grid, massive information-processing plants have begun pumping data and software code into our homes and businesses. This time, it's computing that's turning into a utility."

8/23/12 Fall 2012 -- Lecture #1 48

"And In Conclusion, …"

- CS61c: Learn 6 great ideas in computer architecture to enable high performance programming via parallelism, not just learn C
  1. Layers of Representation/Interpretation
  2. Moore's Law
  3. Principle of Locality/Memory Hierarchy
  4. Parallelism
  5. Performance Measurement and Improvement
  6. Dependability via Redundancy
- Post PC Era: Parallel processing, smart phone to WSC, Software that executes across the net