Conquering Complexity

Building Systems with Billions of Parts

Participants (at the end):

Rod Brooks, Seth Copen Goldstein, Anant Jhingran,
Len Kleinrock, Richard Newton, Steve Reiss, Bob Sproull

June 25, 2002

Conquering Complexity

Building Systems with Billions of Parts

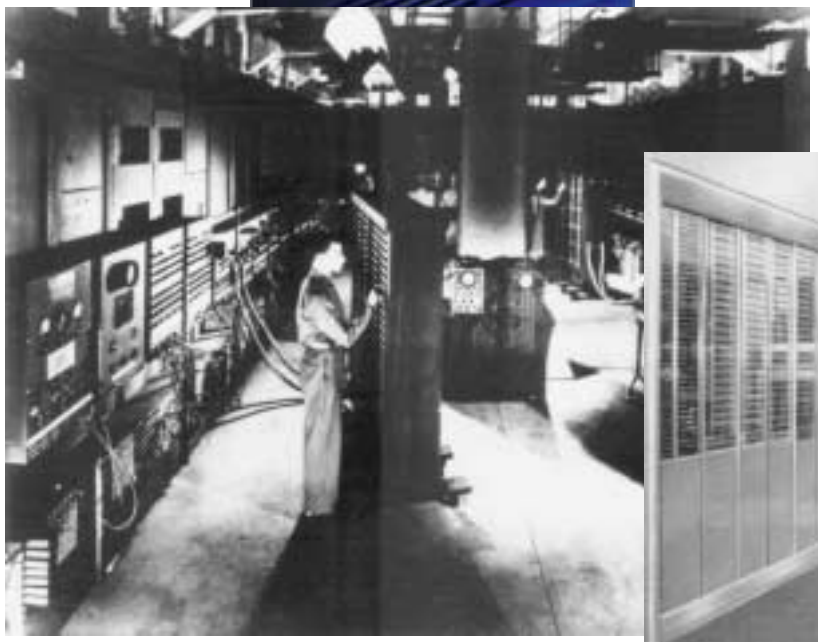
Participants (at the end):

Rod Brooks, Seth Copen Goldstein, Anant Jhingran,
Len Kleinrock, Richard Newton, Steve Reiss, Bob Sproull

June 25, 2002

What do these have in common?

musical greeting card?

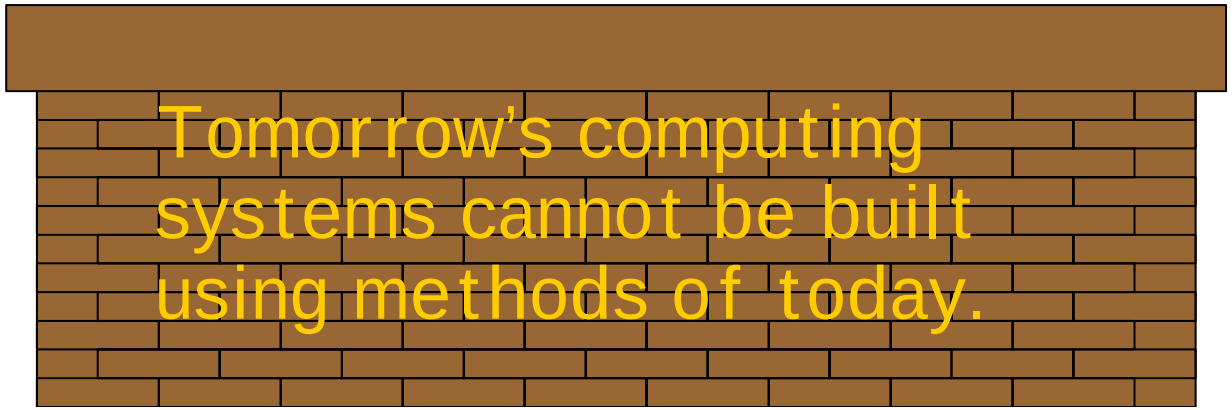


Mission Statement

To reformulate computing systems architectures at all levels (from circuits to global-scale distributed systems) that break through the complexity wall to deliver robust, scalable, long-lasting, systems.

Why?

- We need computing systems that are the agent of change in society rather that enemy of change
- Every artifact we build or grow in the future will likely have a computing component
- We have run into a complexity wall, that limits and inhibits growth in business and societal systems



Two Themes

1) Complex organized behavior out of many simple unreliable components

2) Make complex systems simple to the
   - User
   - Administrator
   - Designer

Common Challenges

- Federation of a large number of units
- Units that can change over time
- Wide and dynamic range of latencies and bandwidths among components (all the way to occasionally disconnected?)
- Scaling with ease

Common Attributes

- Self-configuration
  - The inductive step is free
  - Emergent behavior
- Self-Adaptation
  - Changes in environment (e.g., load, failures)
- Reusablity
  - Small changes in function without reengineering
  - Meta-programming
- Motherhood and apple pie (robust, secure, stable, …)

How we do it now

- Abstraction/Layering
  - Fixed APIs between layers
  - Fixed functionality at each layer
- Deterministic interaction between components
- Deterministic approach to failure
  - Explicit coding of failure into system
- Performance centric implementations
- Result: Rigid, brittle systems

Possible Approaches

- Collective intelligence
  E.g., Swarms
- Localize change
- Evolutionary models
  - Adaptation
  - Bio-mimetic approaches
- Modeling for system level effects
- More autonomy at every level
- Market mechanisms
- Simple Many and Self-Healing (SMASH)

Some Applications

- Build reliable computer systems with billions of components
- Sensor network that covers the earth
- Connect every person to the network
- Smart matter - reconfigurable artifacts
- Networked matter (instrumented earth)
- Simulated Reality: Simulation engine
- Understanding biological systems
- Intelligent Transportation Systems
- Critical to Ubicomp & Trustworthy

Computational Paint

- Click to add text

Organized behavior from many simple devices

Intelligent Transportation Systems

- Click to add text

Robust, reliable, maintainable, scalable behavior from complex devices

Success Is

- Complexity is not the weakest link
- Metrics
  - Deployed systems per engineer
  - Maintenance costs
  - Administration costs
  - System longevity
- At least linear improvement with increased size

Conclusions

- Complexity limits our ability to meet important needs
- Tweaking won't solve the problem
- We need a top-to-bottom re-examination of the way we architect and build computing systems

We must conquer complexity