Staying off the Hot Seat with Cool Mobile Systems

MobiSys '2005

Dr. Alfred Z. Spector
Chief Technology Officer, IBM Software
IBM Corporation

aspector@us.ibm.com
June 7, 2005

IBM

Outline

- Mobile System Trends
- Implications
- Current State of Robustness
- Some Challenging Research Areas
  - Security research opportunities
    - Trusted computing base
    - Uses of trusted computing base; e.g., provenance
  - Complexity research opportunities
- Conclusions

Abstract

Mobile systems are benefiting from continuous innovation: ever reduced physical size, increased connectivity, and more interaction modalities. In parallel, we have envisioned and enabled ever more sophisticated scenarios in which these devices interact with humans and their physical environment. When deployed, these scenarios will often require complex software operating in large scale, on open shared networks, and involving people and machinery. Thus, failures (whether unintended or due to malicious attack) could make traditional I/T security and robustness failures seem relatively minor in comparison. The associated pain will also spread from logical I/T domains to physical domains.

With this motivation, I argue that the greatest challenge in building large scale mobile and pervasive systems will lie in providing robustness and security, with the concomitant need to manage complexity to users and administrators. I will discuss key elements of a research agenda here. As one component, I'll discuss the importance of trustworthy hardware modules that are used by trustworthy software modules. I will propose application of some specific ideas as the application of currently available technologies like the Trusted Platform Module (TPM), and some newer work in secure hypervisors and the attestation of data provenance.

IBM Research Division

aspector@us.ibm.com
June 7, 2005

Mobile System Trends

- Technology Push
  - Modalities growing
  - Form factor improving
  - Cost declining
  - Connectivity exploding
  - …
- Scenario Pull
  - Medical informatics
  - Societal Security
  - Integration of people and machines
  - Inputs for continual optimization
  - …
- Most scenarios envision amalgams of components where principals, devices types and instances, device/server software, and communication networks are increasingly fluid.
- It is impossible to fully anticipate/enumerate all system interactions at time of system construction
- Embedded modularity (e.g., hierarchy) not likely

Ex.: Intelligent Notification in Health

Notify me when…

- Results of blood work for Smith completed.
- Suspicious biometric data for Jones available.
- A patient of mine enters the ER.

**"From: Lab
Subj: Blood Work (Smith)"**

Email

Notification Service

Short text message, voice notification

Instant message

"Your patient (Brown) just entered the ER"

Right info, right person, right time, right device

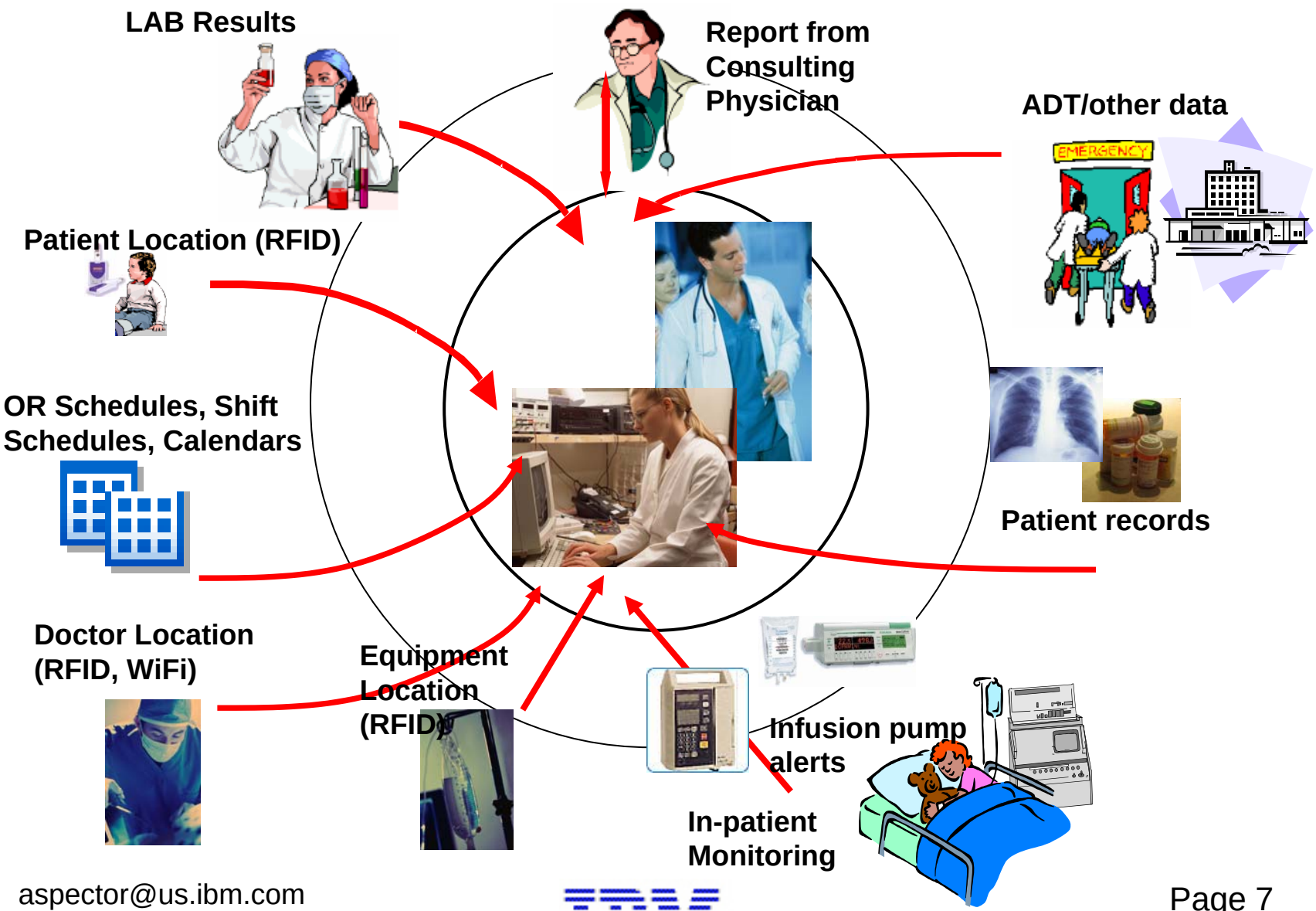
- **Increased productivity, responsiveness**
  - **Short shelf life of information**
  - **Real time information for real time decision-making**
  - **Proactive problem prevention**

"URGENT: Alarming biometric data (Jones)"

aspector@us.ibm.com
June 7, 2005

IBM

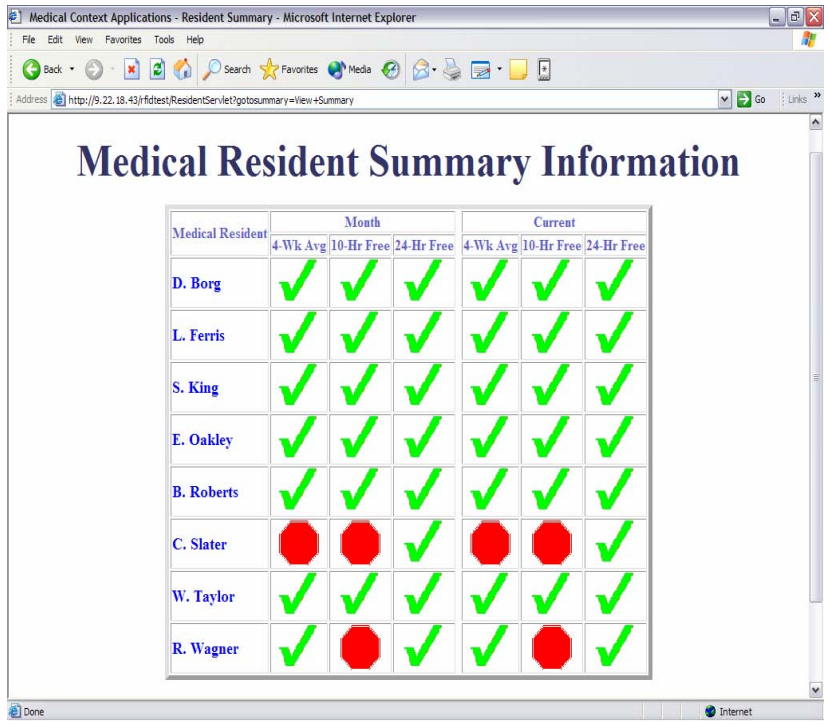
Breadth of Devices and Connectivity



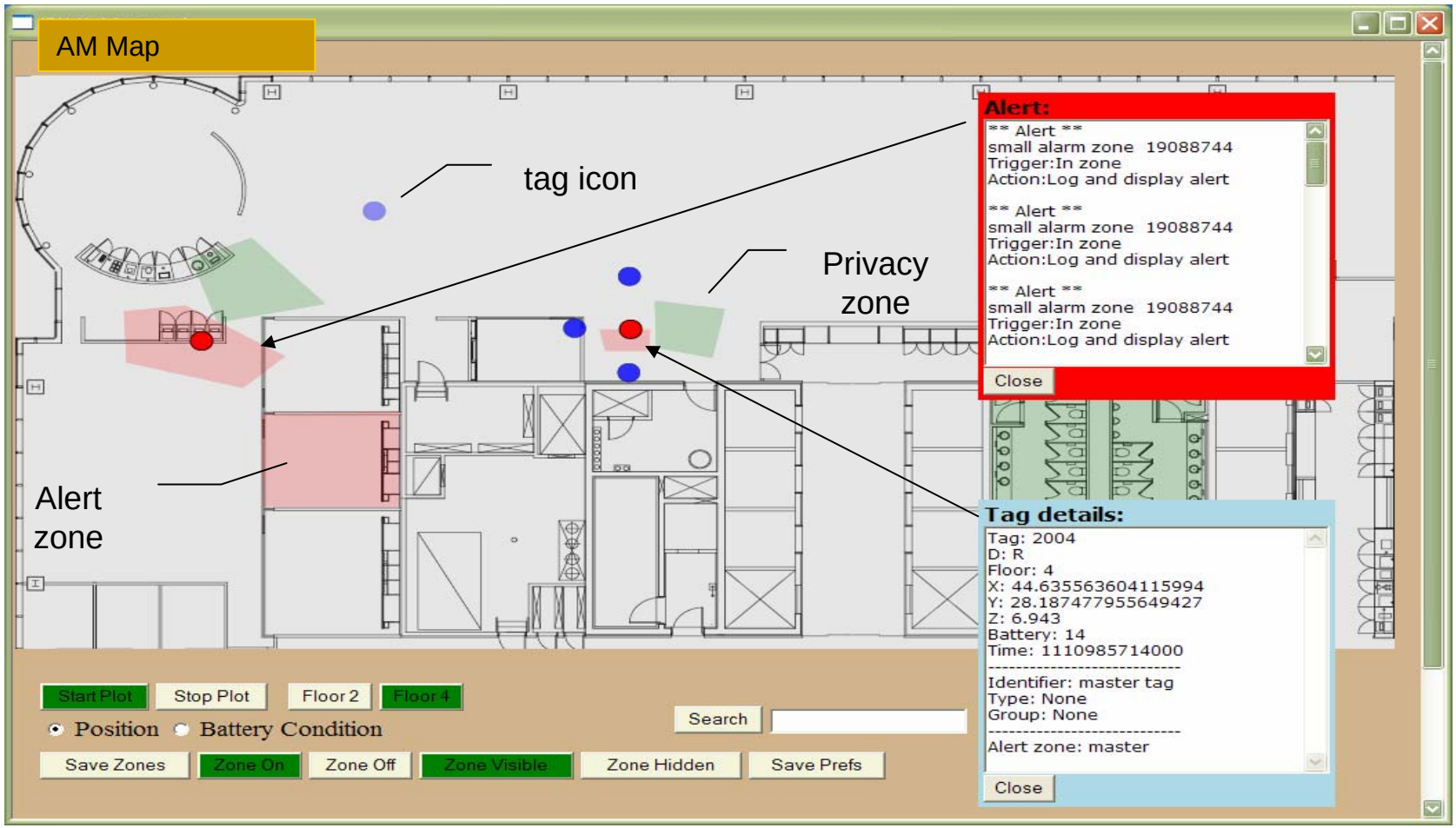
aspector@us.ibm.com
June 7, 2005

Medical Resident Monitoring

- On-duty time of medical residents limited by state and federal legislation (Hours per week, free periods, down time)
  - Noncompliance can lead to loss of medical school accreditation, and accurate reporting by residents is a problem.
- Solution: Tag residents with active RFID tags; place readers at exits & entrances; monitor and notify both resident and supervisor
- Context Sources: location, resident assignment schedules/calendars



Advanced Asset Monitoring



Ex: Highlighting *Continual Optimization*

1. Almost everything can almost always sensed
2. We can effect change at geometrically declining costs
3. With fast processors, and good optimization algorithms, the opportunity for continual optimization is great.

   (e.g., think real time societal scale feedback control...)

Observations:

- Continual optimization could fundamentally change how we might operate organizations and impact our lives
- Very interesting interplay of human & machine decision-makers
- But, “garbage in” or system failure could induce significant problems if systems are designed improperly

The greatest challenges are systemic in nature

Implications

- Mobile system scenarios dramatically increase need for Robustness:
  - Ease of use
  - Ease of evolution
  - Quality of service
  - Reliability
  - Security
  - Fitness to purpose
- Consider:
  - Medical monitoring vs. loss of availability in stock trading
  - Hacking societal systems vs. losing “sensitive” data

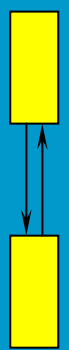
Current State of Robustness:

The Conundrum of Distributed Systems

Distributed Softare Systems Today

Score high on most metrics

- Amount of code
- # of dependencies
- # of programmatic interfaces
- # of layers
- Administrative interface size & configuration options
- Non-uniformity
- Non-orthogonality
- Defects
- Documentation
- # of programmers involved



- **My brand new cell phone is going down the same path ☹**
- **Sendmail is (or was) the longest O'Reilly Book**

RosettaNet Purchase Orders

- There are 551 XML fields in the PurchaseOrderRequest
- There are 700 XML fields in the PurchaseOrderConfirmation

Excerpted First lines of purchase order confirmation:

```
fromRole.PartnerRoleDescription
    |– ContactInformation
    |    |– contactName.FreeFormText
    |    |– EmailAddress
    |– facsimileNumber.CommunicationsNumber
    |    |– telephoneNumber.CommunicationsNumber
    |– GlobalPartnerRoleClassificationCode
    |– PartnerDescription
    |    |– BusinessDescription
    |    |    |– GlobalBusinessIdentifier
    |    |    |– GlobalSupplyChainCode
    |    |– GlobalPartnerClassificationCode
GlobalDocumentFunctionCode
PurchaseOrder
    |– AccountDescription
    |    |– accountName.FreeFormText
    |    |– AccountNumber
    |    |– billTo.PartnerDescription
```

```
|    |    |– BusinessDescription
|    |    |    |– businessName.FreeFormText
|    |    |    |– GlobalBusinessIdentifier
|    |    |    |– PartnerBusinessIdentification
|    |    |    |    |– ProprietaryBusinessIdentifier
|    |    |    |    |– ProprietaryDomainIdentifier
|    |    |    |    |– ProprietaryIdentifierAuthority
|    |    |– ContactInformation
|    |    |    |– contactName.FreeFormText
|    |    |    |– EmailAddress
|    |    |    |– facsimileNumber.CommunicationsNumber
|    |    |    |– PhysicalLocation
|    |    |    |    |– GlobalLocationIdentifier
|    |    |    |    |– PartnerLocationIdentification
|    |    |    |    |    |– ProprietaryDomainIdentifier
|    |    |    |    |    |– ProprietaryIdentifierAuthority
...
```

Note: RosettaNet is a consortium of major companies working to create and implement industry-wide, open e-business process standards, that will form a common e-business language, globally aligning processes between supply chain partners. (From RosettaNet Home Page.)

Spam, Phishing, et al.

- Fighting spam will cost global businesses $50 billion in lost productivity & security expenses this year
- Workers at some of the country's biggest corporations report that they spend nearly 15 minutes every day sifting through an average of 29 unsolicited e-mail messages, dramatically higher than the seven minutes they spent sorting through spam in 2003[1].
- No single technique appears to be able to solve these problems, but it is clearly the case that poor engineering has engendered many problems.

[1] <http://www.washingtonpost.com/wp-dyn/articles/A21657-2004Jun7.html>

Really Reliable Systems

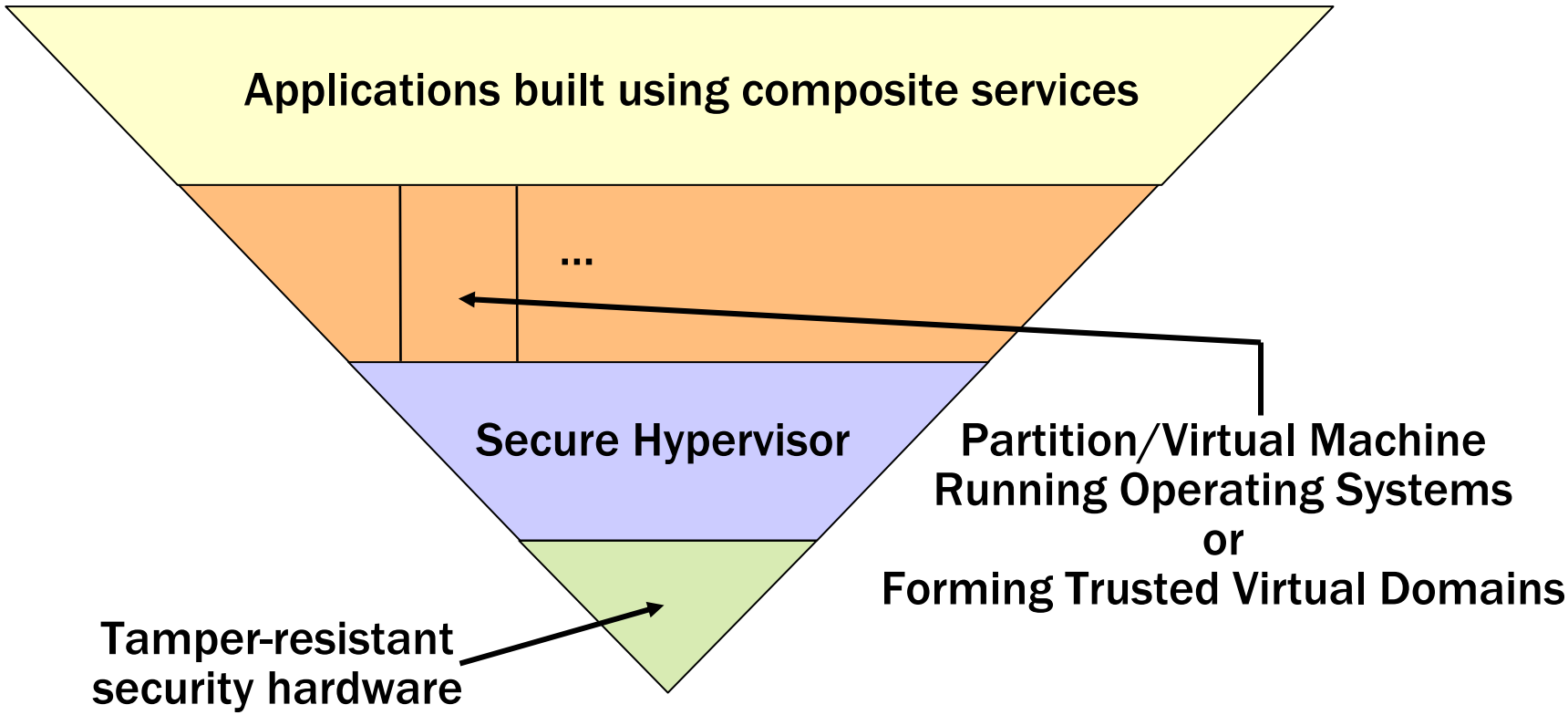
- Typically embedded, and rather closed
- Extremely expensive to build
- Very hard to modify => Rigid
- Very difficult to replace
- Not the right model

How have we done to-date?

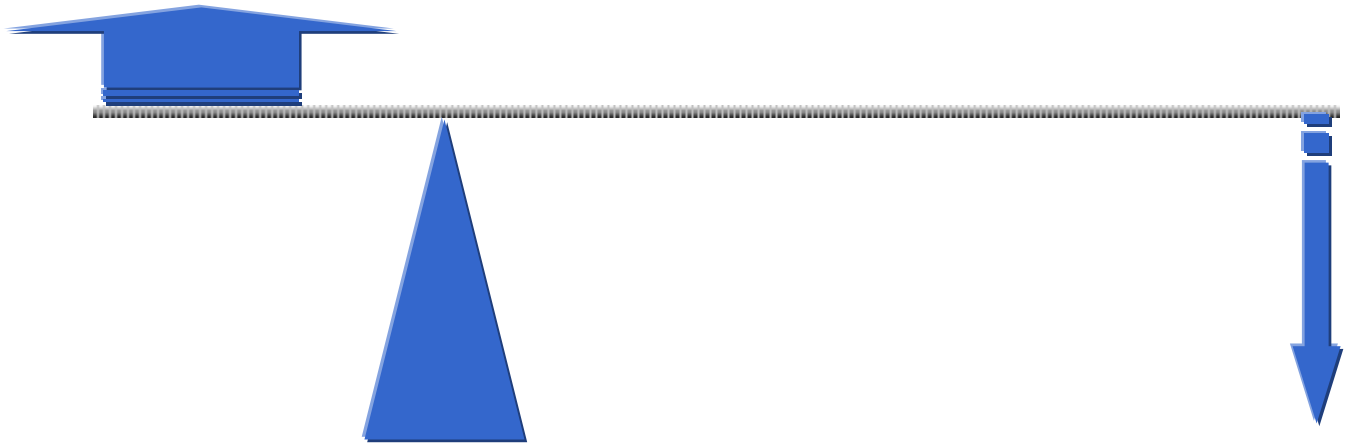
- We have build great systems that generate great value
- But we have clearly not solved, and in some cases ignored, hard problems
  - Configurability vs. protection
- Some practices are mdeicore
- Perhaps, we counted on a “closed community,” obeying a social compact, or applications limited downside risk
- In any case, we must now confront robustness issues particularly in pervasive systems.

Security Opportunity

Build & Leverage a Trusted Computing Base



Gain Leverage on Problem



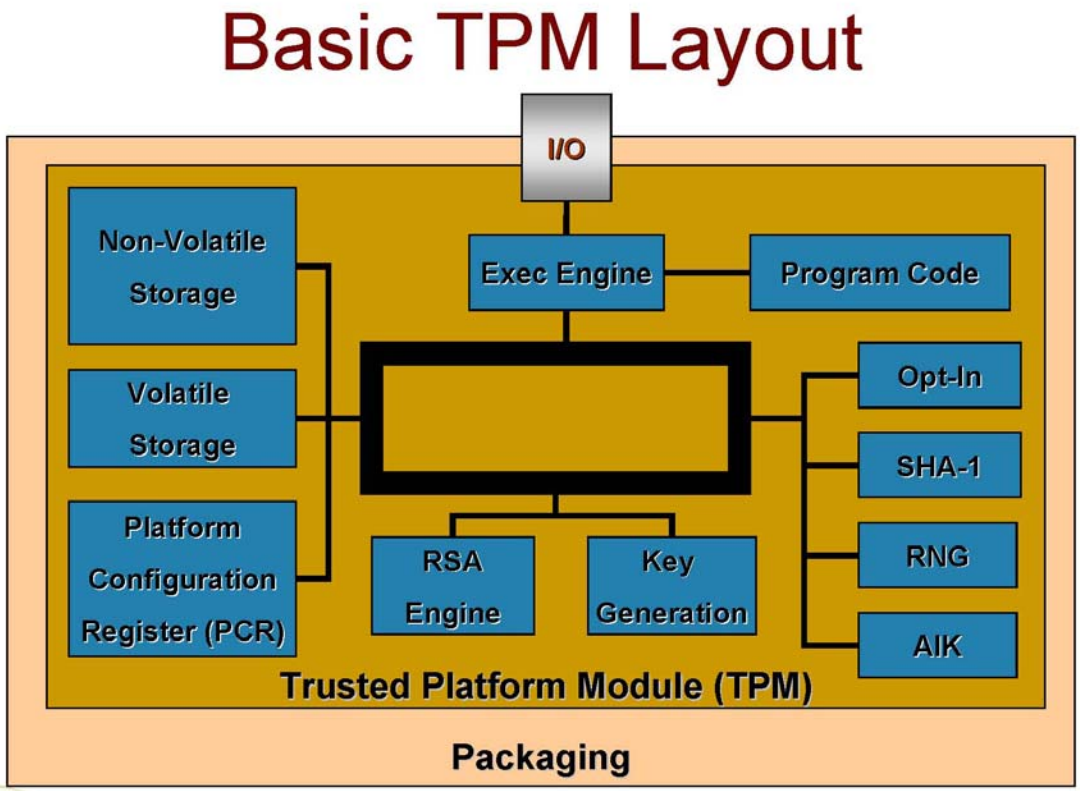
- We need a fulcrum from which to gain leverage
- The fulcrum cannot require the wholesale modification of the entire stack for many reasons
- We need some place where small amounts of hardware and software can yield great benefits

IBM 4758 PCI Cryptographic Coprocessor

- Performs high speed cryptography
- Provides secure key storage
- **Tamper-resistant, sensing** and **responding:**
  detecting physical attacks (probe, voltage, temperature, radiation)

- **Programmable**
- Secure configuration and field upgrades
- US and Canadian Government Certified:
  FIPS 140-1 overall level 4 certified
- Support for Windows, Linux, AIX, Solaris, OS/390, OS/400

From TrustedComputingGroup

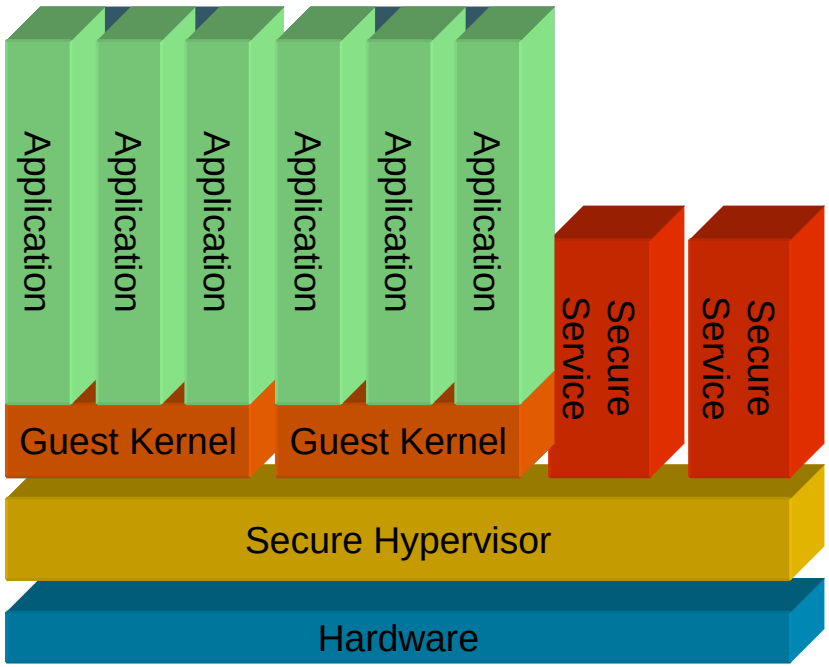


TRUSTED COMPUTING GROUP™

TCG Confidential

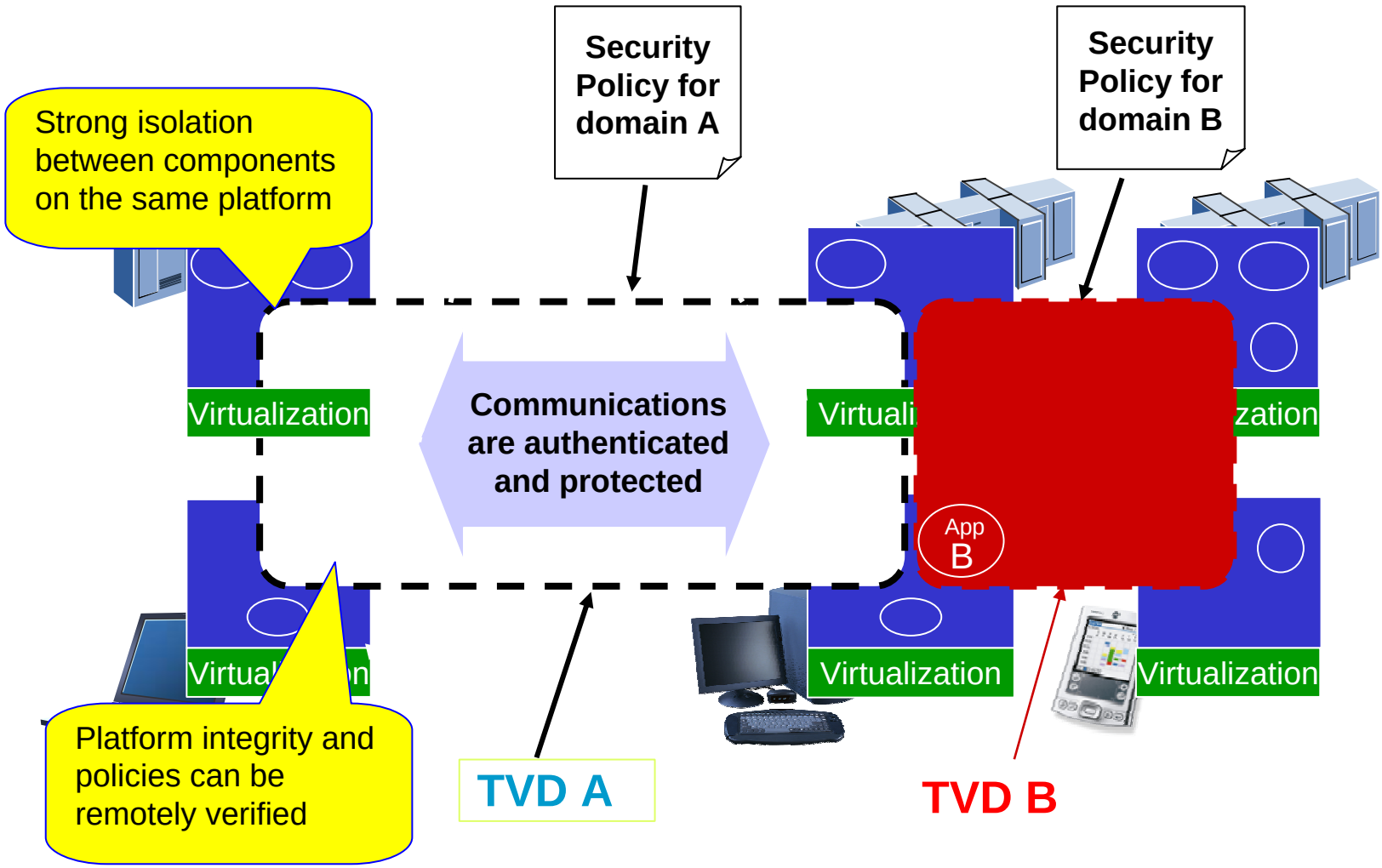
Slide #11

Secure Hypervisor Architecture Goals



- Strong isolation guarantees between virtual machines
- Mediated resource sharing and communications
- Platform and virtual machine content integrity guarantees
- Platform and virtual machine content attestation
- Resource control and metering
- Secure services – e.g., audit, monitor, I/O, …
- Research Implementations: Xen and PHYP

Trusted Virtual Domains



What to do with this?

- Isolation of Operating System Partitions
- Development of trustworthy capabilities
  - E.g., attestation
  - E.g., privacy services
  - E.g., authentication
  - E.g., provenance management

Provenance Management Defined

- **prov·e·nance** P Pronunciation Key (pr v -n ns, -näns) *n.*
- Place of origin; derivation.
- Proof of authenticity or of (historical) past ownership. Used of art works and antiques. ***From dictionary.com***

Transforming this notion to the information world: the origin (including all modifications) of each piece of data could be ascertained

Why Care?

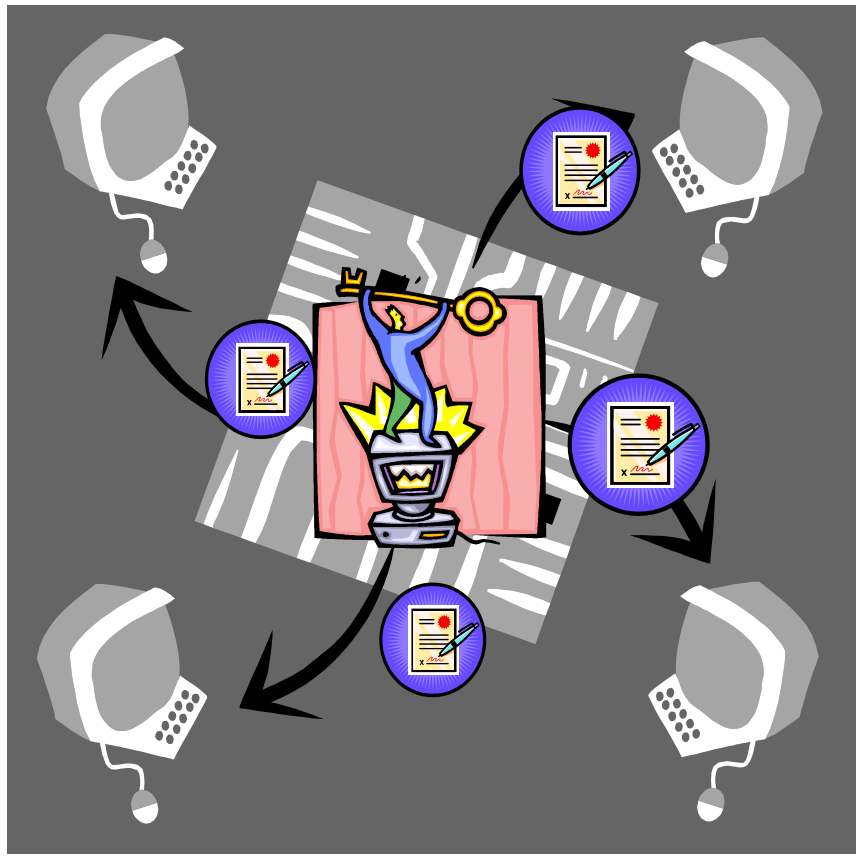
- The source origin of things such as viruses, worms and spam are unclear
- Solution: Information Provenance (InfoP)
- When a virus attack breaks out, the information provenance is accessed, the generator of the virus can be found and then laws and law enforcement will come into play

How Do We Achieve InfoP?

- **Main idea:**
  - Sign everything
  - Associate in perpetuity signatures with creators/updaters in perpetuity*
  - Don't trust unsigned or improperly certified data
- **Enabling components**
  - Trusted Components
    - Trusted Platform
    - Trusted Virtual Domain
    - Certification Authority (CA)
    - Storage repository
  - Laws

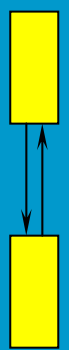
Concept: Everything Is Signed

- All data is signed by creator and all later modifiers
- Signatures are stored irrevocably
- Upon transmission, the data, signature list is sent, and certificate of signing key
- Policies on the computer determine what is acceptable provenance



Utilizing the InfoP

- A virus is found with U's signature
- U is contacted by law enforcement
- U can access its repository and look for a signed version of the virus with a certificate
  - If U finds such data then it passes it on to the law enforcement, and if certificate is acceptable to them then they proceed from there
  - Otherwise, user U is dealt with by the law



Challenges

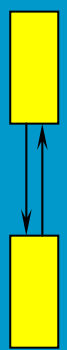
- Privacy
- Storage:
  - How is such mass data stored and searched
- Signatures and CA's
  - What will constitute a valid signature
  - What will constitute a trusted CA
- How are cross country laws enforced

Slide Inserted at Req. of Richard Paine

(Richard made a comment on this during the Q+A's)

The Secure Mobile Architecture (SMA) is an integration architecture developed in The Open Group by Boeing, Lockheed, IBM, HP, Netmotion Wireless, and a number of universities. The URL of the published document is at the following address:

http://www.opengroup.org/bookstore/catalog/select.tpl?text=secure+mobile+architecture



Science of Design Increasing Robustness

From the "Conundrum of Systems,"
http://www.csail.mit.edu/events/DLStalks/dlsspector03.html
http://www.research.ibm.com/people/a/aspector/presentations/AZSDertouzos.pdf

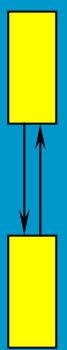
The Conundrum of Systems

- Our field is 50+ years old
- We have many great engineering techniques:
  - Generalization, Encapsulation, Re-use
  - Components Integration Technologies
- We have a large base of systems, tools, techniques, and components
- Despite all this, systems aren't what we want them to be
- But, there is more effort going into robustness now than previously

3 Categories of Complexity

- Classic Complexity
  - Time
  - Space
- Implementation Complexity
  - Logical
  - Structural
  - Comprehensibility
- Usage Complexity

Task	Pre-Use	Novice	Middle	Expert	Except-ion
Install					
Configure					
Administer					
Use					



There are steps to take

- Meaning
- Measuring
- Methodology
- System Architecture
- Science and Technology
- Acknowledgment, Legal & Cultural Change

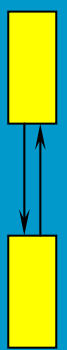
Meaning

- Computer scientists instantly know about time and space bounds
- It is just as important to arrive at clear **definitions** of all forms of complexity
- There has been work in this area, but we are likely to arrive at something like:
  - Classic Complexity
  - Implementation Complexity
  - Usage Complexity
- I note this topic is a very small part of the CS curriculum today

The Unmeasured Life Is Not Worth Leading

- If we can reach some definitions, we should try to create metrics
  - Minimization or Maximization adds focus and fun
  - Where metrics have existed in the field
    - Latency/Throughput
    - Word accuracy
    - Recall & precision
    - Translation quality

  More progress has been made
- The are risks to measuring things (you get what you measure)
  - I think metrics could be the strongest weapon against complexity



Methodology

- User-centered methods
- Ethnography
- Product Lines
- Increased use of metrics
- Component-based
- Sunset Clauses

Focus on The Right Function

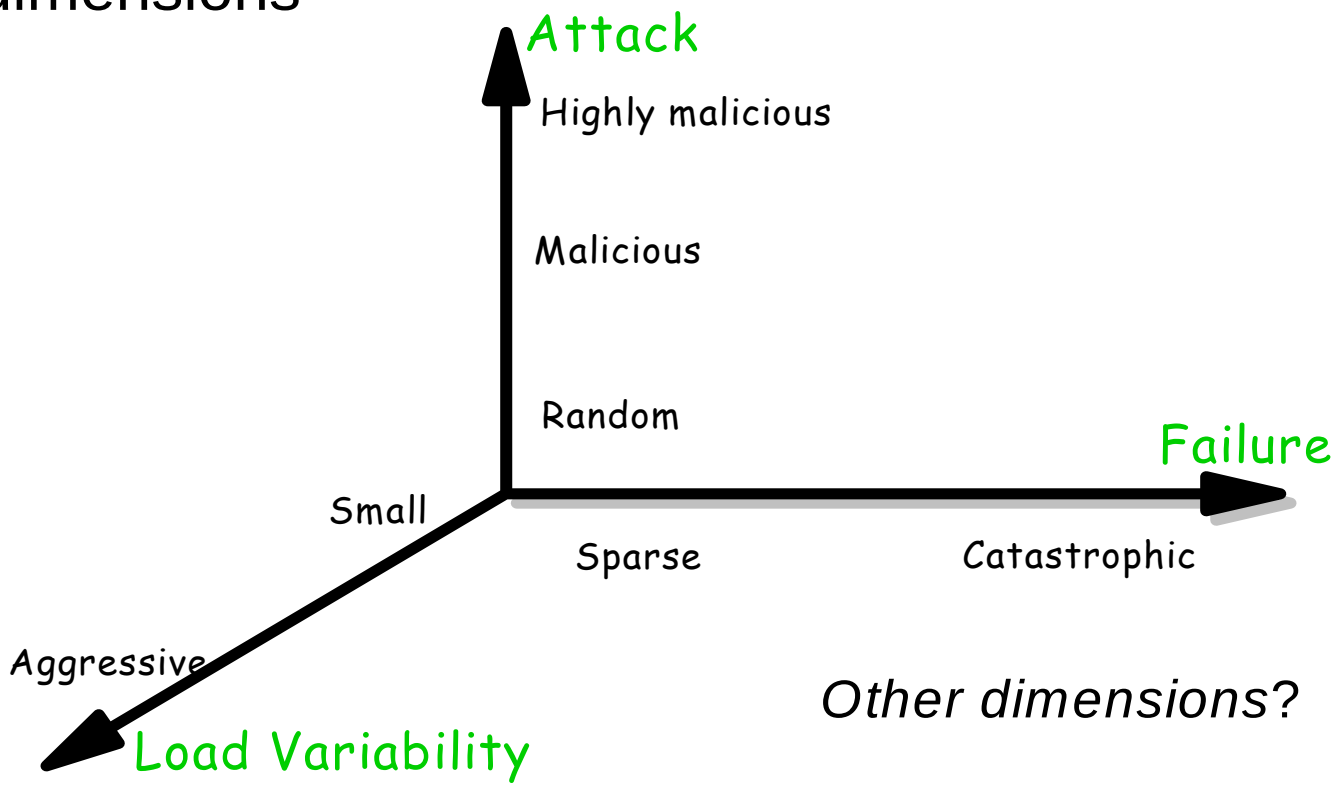
- What do our user communities really want?
  - Can we more directly provide exactly that and dispense with distracting and wasteful items
  - Can we focus on the breadth of the problem and provide a solution to it, perhaps with incrementally more function
  - Perhaps, either directed or automatic adaptation to usage community required

System Architecture

- We need to have higher standards
- Example:
  - In systems today, we have disks, partitions, volumes, logical volumes, file systems, and directories structures
    - Do all of these still need to be visible interfaces?
    - Why not have a configuration option to set MBTF to Low, Medium, or High?
  - With ACLs, there could be far more useful profiles established
- What about increased use of classic AI techniques?

Science and Technology

- Autonomic computing concept: Making systems robust in the presence of stimuli occurring in different dimensions



Autonomic Computing

- Subsystem design improved to eliminate manual control
- Core techniques:
  - Control theory
  - Increased use of rules systems; perhaps, with inference & common sense
  - Negotiation
- Standardization of event reporting to provide opportunities for data mining, statistical machine learning, and more feedback control
- Architecture

Acknowledgment, Legal, & Cultural Change

- As scientists, we should foster greater responsibility for consequences of designs
- We should increase university and research focus
  - Education curriculum
  - Research agenda
  - Opportunity to broaden university collaborations
- We need to debate role of legal system
  - As we ever-more depend on computers, how do customers/society evaluate risk?
- Systems builders need to return more to artistry

Conclusions

Summary

- Trust:
  - Trust in systems is a huge problem and likely to become greater with applications of pervasive device networks
  - There is some progress in traditional systems design; the Mobisys community needs to be leader her, not a follower
- Complexity:
  - Complexity grows despite all we have done in computer science, from Simon's *Sciences of the Artificial* to modern programming languages & software engineering techniques
  - There is valuable, rewarding, and concrete work for Computer Science in combating complexity:
- These areas of work will prove as valuable as direct functional innovation
- *If we get this right, Distributed Systems (including mobile systems) will become the World's Operating System*