Software Systems & Engineering Research in ECE

25 June 2014
Sathish Gopalakrishnan
Real-Time in the Prime Time
(Sathish Gopalakrishnan’s research focus)
Real-Time in the Prime Time

Max ECUs Per Car

- 1978: Bosch’s ABS put in Mercedes S Class
- 1984: Motronic digital engine management
- 1987: CAN Bus Introduced

Point-to-point → Networked ECUs → Distributed Functions
Real-Time Systems Theory

Real-time systems often have resource constraints.
How do we provision for mixed-criticality systems?

We need methods for deriving high confidence results.
Some Challenges (2/3)

How do we architect systems to exploit concurrency?
Some Challenges (3/3)

How do we utilize the “approximability” of programs to improve energy efficiency and reliability?
Broad Theme
How do we build scalable, reliable software systems?
**ExM** -- System support for exa-scale science applications ($2M DoE funding; collaboration with UChicago and ANL)

**VMflock** -- Virtual machine migration across clouds (collaboration with Storage Systems at IBM Almaden)

**StoreGPU** -- Exploiting GPUs to accelerate distributed storage (NSERC & other funding)

**Totem** -- Graph processing on heterogeneous CPU and GPU platforms (NSERC funding, NVIDIA)

**Error resiliency of GPGPU applications** -- Characterization and fault tolerance mechanisms for GPU applications (collaboration with AMD, LANL)
Building Error Resilient Applications

Karthik Pattabiraman

Motivation: Hardware errors are increasing with shrinking transistor sizes

- Detecting errors in parallel programs
  - Use deviations in control flow among multiple threads to detect potential errors

- Enabling soft computing
  - What data items should we protect for “reasonable” accuracy?

- Low-Level Fault Injector
  - Mechanism to study reliability
Software Testing

Make web and mobile applications more dependable and maintainable by:

investigating new \textit{automated} ways of analyzing, understanding, and testing them.

Automated test generation
Test suite adequacy analysis
Exploration of GUI interactions

Cross-platform compatibility testing
Understanding program behavior
Automated fault localization and repair
Friction and Technical Debt in the Software Development Process

Technical Debt

Friction

Social Debt

Reduced velocity
More defects
Delays
...

Philippe Kruchten
Friction and Technical Debt in the Software Development Process

“Debt”: the results from poor, suboptimal decisions in the past, for example to meet a hard deadline

Technical debt: software hard to evolve or maintain
Social debt: team structure and process hard to evolve

Decision process in software engineering

- Impact of cognitive biases
- Management of technical debt:
  - taxonomy, assessment, measurement, prevention
- Role of software architecture
- Role of agile processes, in all of the above
Usable Security

- Heuristics for evaluating IT security management tools
- Security and privacy in online social networks
  - E.g.: Thwarting fake accounts by predicting their victims
- Privacy aspects of health-related information sharing
- ...

Kosta Beznosov
How do we manage complexity at scale?
Faculty Members

- Philippe Kruchten (since 2004)
- Kosta Beznosov (since 2003)
- Sid Fels (since 1998)
- Matei Ripeanu (since 2006)
- Sathish Gopalakrishnan (since 2007)
- Karthik Pattabiraman (since 2010)
- Ali Mesbah (since 2011)

Research Areas:

- Human-computer Interaction
- Usable Security
- Software Engineering Processes
- Distributed Systems
- Real-Time Systems
- Fault-Tolerant Computing
- Software Testing