SOFT 437

Software Performance Analysis

Course Outline

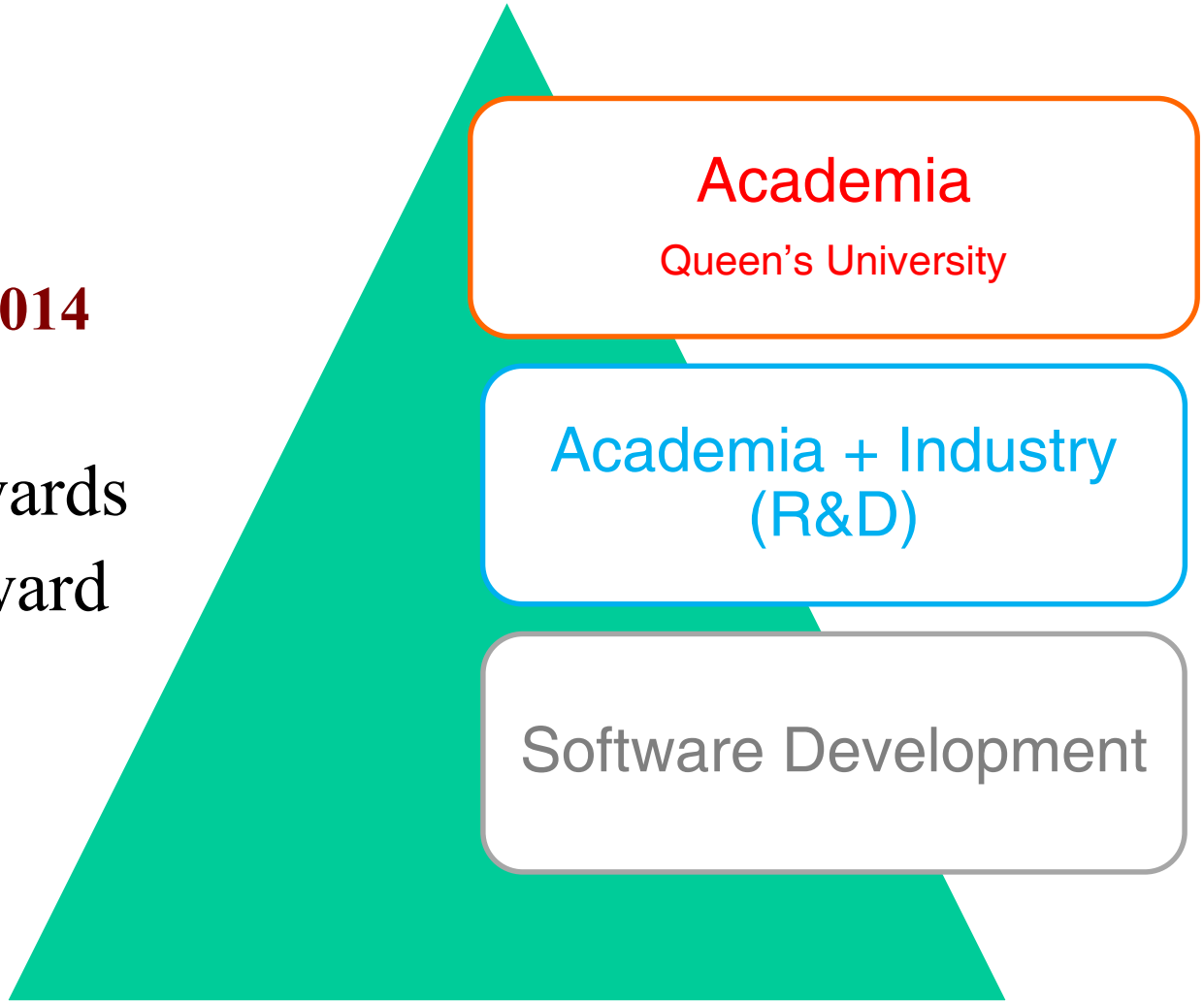
General Information

- Instructor: Dr. Khalid Elgazzar
  - Adjunct Assistant Professor – Queen's School of Computing
  - http://cs.queensu.ca/~elgazzar/
  - elgazzar@cs.queensu.ca
  - Office: Goodwin 531
  - Office hours: Anytime (advance appt is recommended)
- Course Web site/Resources
  - http://cs.queensu.ca/~elgazzar/soft437/
  - http://www.perfeng.com/
- Schedule – **Goodwin 247**
  - Monday: 11:30 AM – 12:20 PM
  - Tuesday: 13:30 - 14:20 PM
  - Thursday: 12:30 - 13:20 PM

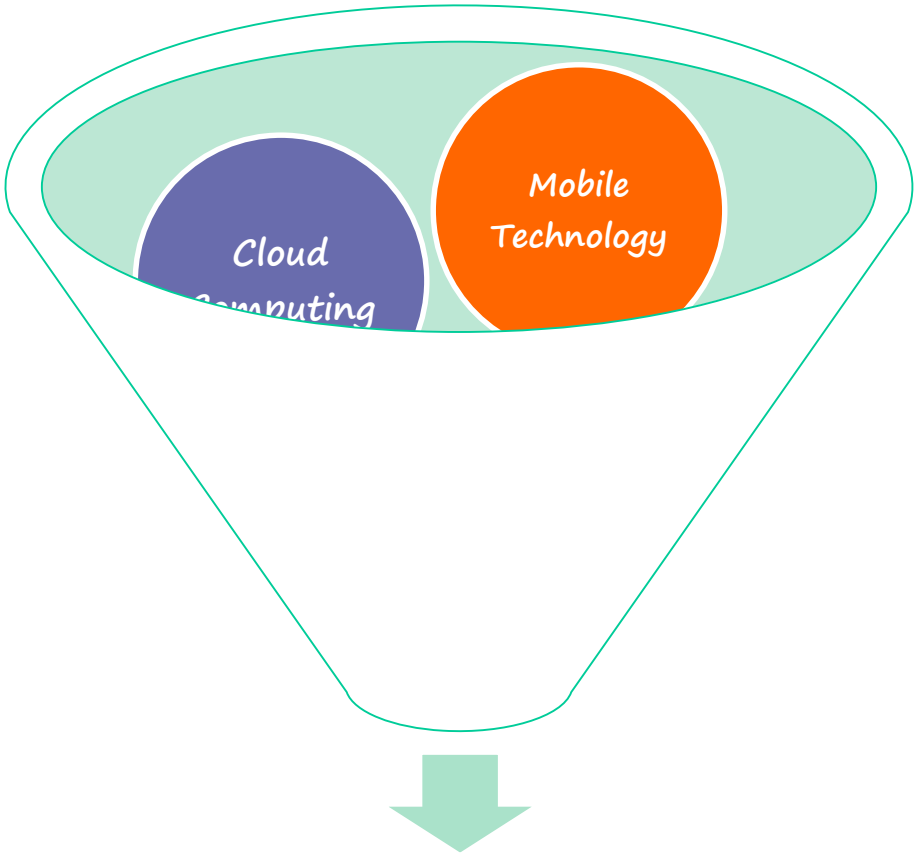
Khalid Elgazzar

Research Award 2014

- 2 Best Paper awards
- Mitacs/IBM Award
- Founder of 3 Workshops



Research Interests



Mobile and cloud Services

Web and Text Resources

- **Lecture notes available at:**
  http://cs.queensu.ca/~elgazzar/soft437/
- **Text book**
  - *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*, by Connie Smith and Lloyd Williams, First Edition, 0-201-72229-1
- **Reference books**
  - *Capacity Planning for Web Services: Metrics, Models, and Methods, by Daniel A. Menasce and Virgilio A. F. Almeida, Prentice Hall PTR*
  - *Scaling for E-Business: Technologies, Models, Performance, and Capacity Planning, by Daniel A. Menasce and Virgilio A. F. Almeida, Prentice Hall PTR*

The Big Picture

- What factors impact the performance of a software system?
  - Response Time
  - Scalability
  - Cost
- Model and measure software systems to predict and find performance problems ahead of time

It is much easier to correct performance problems early in the design phase than it is when all code is written!

Course Contents at a Glance (cont.)

- Introduction and Overview
  - Ch1 - Introduction
  - Ch2 – SPE Quick Overview
  - Ch3 – SPE and UML
- Software Performance Engineering (SPE) Models
  - Ch4 - Software Execution Models
  - Ch5 – Distributed Systems
- Software Architecture and Styles
  - Ch6 – System Execution Model

Course Contents at a Glance

- Data Collection
  - Ch7 – SPE Data Collection
  - Ch8 – Measurements and Instrumentation
- Software Contention and Capacity Planning
  - Resource Contention and Queuing
- Performance Oriented Design
  - Ch9 – Design Principles
  - Ch10 - Software Design Patterns
  - Ch11 - Software Anti-patterns

Marking Scheme

- Course Assignments
  - Apply what you learn into practice
  - Four small projects
    - Skills for software performance modeling, analysis and design
    - Techniques for architecture design
    - Programming skill
    - Presentation
  - You may work in a group of 2, or individually.
- Marking Scheme
  - 4 Assignments - 40%
  - 3 Quizzes - 15% on Thursdays in Week 4, 8, 11
  - Final Exam - 45%

Get to know each other

- What is your name?
- What is your background on software developments?
- Why do you select SOFT 437?
- What do you expect from SOFT 437?

SOFT 437

Software Performance Analysis

Chapter 1: Introduction

What is Performance?

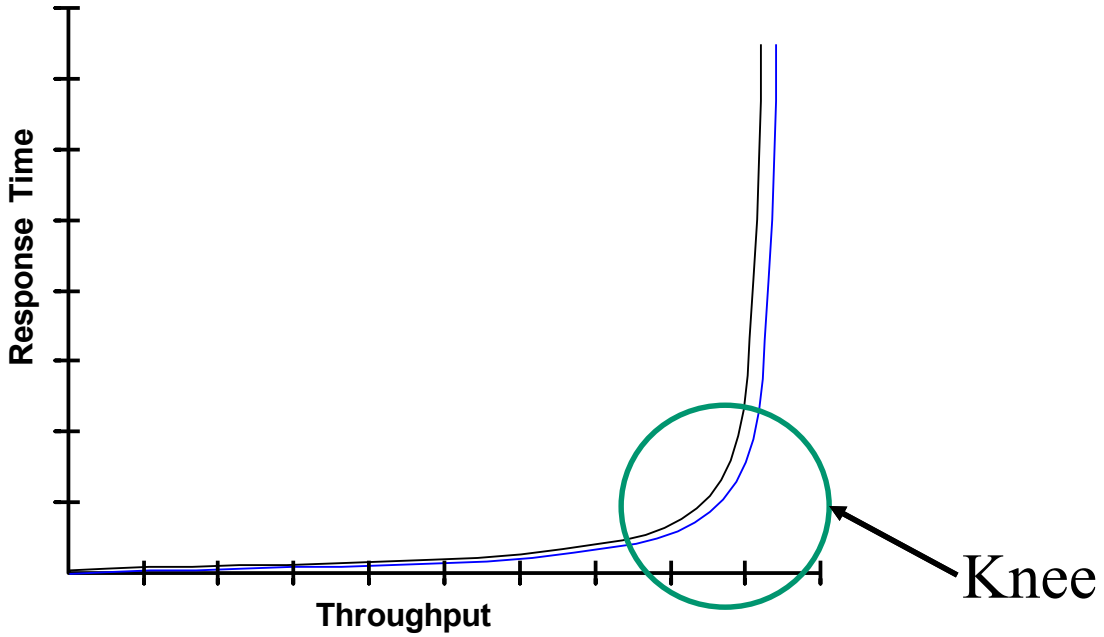
- Performance refers to the ***response time*** or ***throughput*** as seen by **the users**
- ***Response time*** is the time required to respond to a request
  - For example, time required for an ATM withdraw transaction
- ***Throughput*** is the number of requests that can be processed in some specified time interval
  - An ATM network may be required to process 100,000 transactions per hour

Responsiveness

- ***Responsiveness*** is the ability of a system to meet its objectives for response time or throughput
- Responsiveness is a measure of
  - How fast the system responds to an event (example: time for an ATM withdraw transaction)
  - The number of events that a system can process in a given period (example: the number of transactions during an hour)
- Responsiveness has both an objective and a subjective (user perceived) component

Scalability

- ***Scalability*** is the ability of a system to continue to meet its response time or throughput objectives as the demand for its functionality increases.



Why Performance?

If life were like this, we'd not need SPE

Performance in real-time systems

- Siri captures voice snippets on your iPhone, GPS data... (A lot on information)
- Ships the data to an **Apple data center**
- Uses a mixture of cutting edge AI/NLP with a vast database of utterances to make sense of what you said
- Remembers information to improve responses

High performance is not always cool!

"Do you know why I pulled you over today?"

"I'm sorry, sir. I don't know how fast I was going.

"John, your speed was 82 mph, and your top speed today was 91 mph.

Resume "text Sally" ?

Examples of Performance Failures

- NASA Space Expedition – delayed 8 months due to poor performance of FOS software
- [Real-time Systems] Quebec senior house fire 2013 - Fire sprinkler did not activate on time
- On-line Christmas shopping
  - futureshop.ca, bestbuy.ca
  - etoy.com

*Ms. [Lauren Cooks] Levitan said eToys' **performance** looked good until the week before Christmas when it suddenly deteriorated. "If you had a disappointing experience, as a consumer, you're probably not coming back to eToys," she said. "And we won't know for sure if they've fixed their **problems** until next Christmas."*

Examples of Performance Failures

- News media Web sites after 911
  - ***4 seconds*** is the acceptance/tolerable rate for loading a web page
  - It took on average ***13 seconds*** for home pages of news Web sites to load on Tuesday morning
    - CNN.com, New York Time
  - The sites cut photos, graphics, text and ads from the first page, leaving just one story on static Web sites

How should you mange performance?

Fix-it-later

Reactive

- Let's build it first and see what it can do
- We'll tune it later
- We cannot do anything until we have something to measure
- We have fast HW
- Don't worry, you are in safe hands
- Problems? We don't have problems

Proactive

- The project has a performance engineer (PE)
- Everyone in the project knows the name of the PE
- There is a procedure in-place on how to identify performance issues
- Team members are trained in performance processes

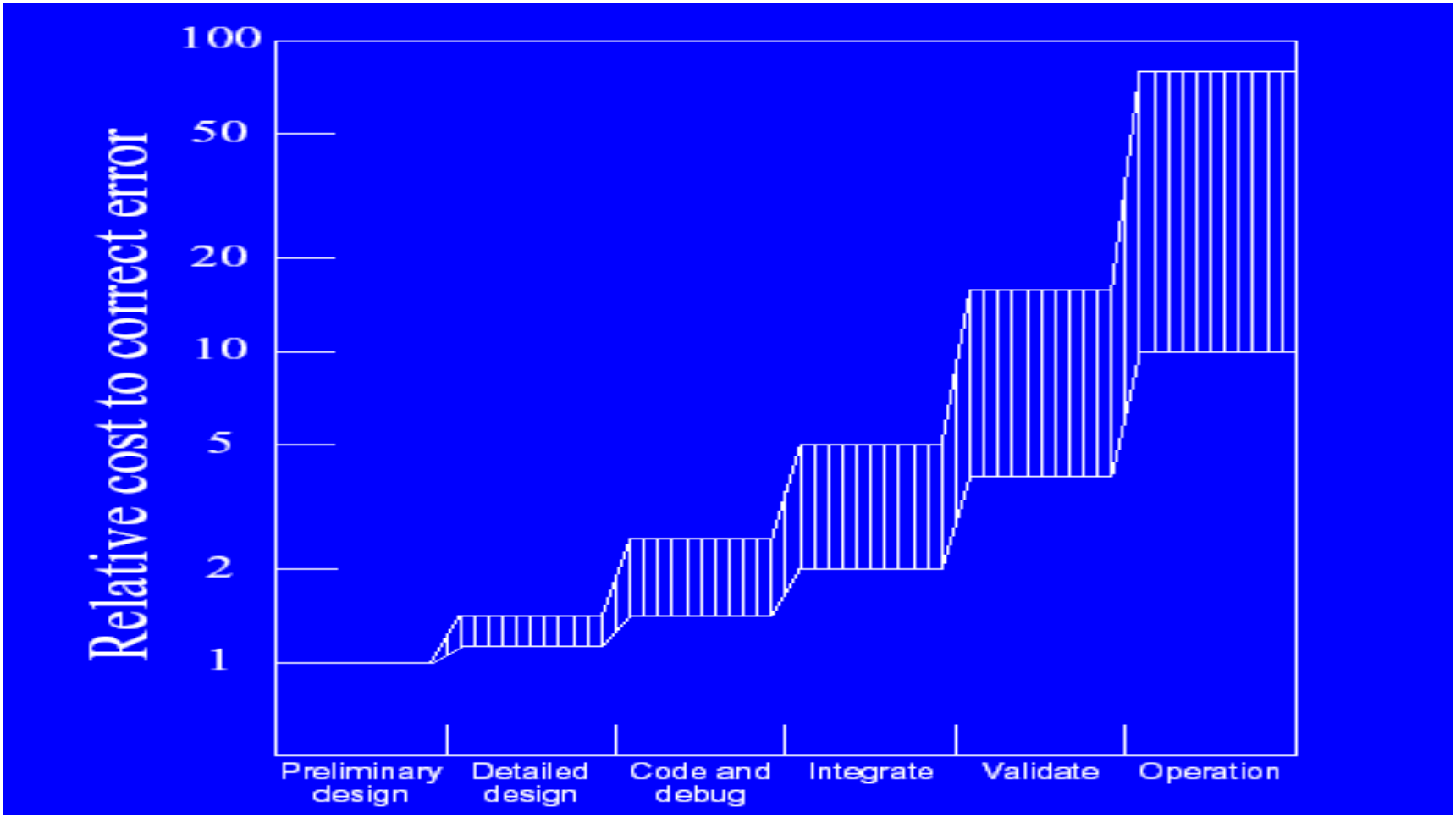
appeared first in the old days

Common Situations

- Performance problems often arise due to fundamental architecture or design factors rather than inefficient coding
- Many problems are not detected until integration testing
- It is more costly to modify code versus modifying designs

The most serious consequence of performance failure is the possibility of business failure!

Relative Cost to Correct Error



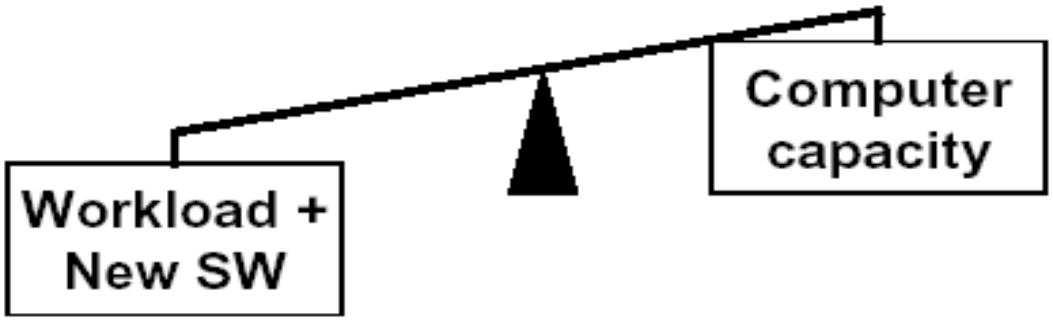
Taken from Course Notes by D. Berry in the School of Computer Science, University of Waterloo

ATTENTION!

The use of new software technology requires careful attention to performance until the performance aspects of the new technology are understood.

Resources VS Demands

Is SPE Necessary?



Resource Requirements Capacity

SPE detect problems early in developments and use qualitative methods to support cost-effective analysis