The following is derived from our discussions at Evaluate 2012.

The following is what we propose to submit by way of feedback to the CS2013 draft (see: Commenting on Strawman Draft)

The following feedback is the result of the EVALUATE 2012 workshop. The workshop was the third in a series of workshops aimed at improving experimental evaluation in computer science (http://evaluate.inf.usi.ch). The meeting was held in Beijing on June 15, 2012 with the theme of *Improving Experimental Evaluation through Education*. One of the outcomes was a set of proposed changes to the strawman draft of CS2013. We believe that our modest proposal will address our concerns by covering basic principles and practices of evaluation.

The changes we propose include:

1. Re-working of the Core-Tier 1 knowledge unit **SF/Performance**. and renaming it **SF/ Evaluation Principles**.
2. Development of a new Core-Tier 2 knowledge unit **SF/Evaluation Processes**.
3. Re-inclusion of experimental evaluation in the description of '**Characteristics of Graduates**'.

Our proposal is an extension and deepening of the '**SF/Performance**' knowledge unit (KU) in the strawman draft. We have grouped the reworked material into two KUs, one Tier 1, and one Tier 2. This conserves the total number of Tier 1 hours relative to the strawman draft (three hours), whilst adding three hours of Tier 2 material. Our partitioning of the material and designation as Tier 1 and Tier 2 is tentative. Our broad rationale is that the Tier 1 material is more fundamental, whilst the Tier 2 material is more detailed and applied. The Tier 1 material includes explicit motivation for the importance of sound experimental evaluation in computer science.

We propose the following Core-Tier 1 KU to be added to **Systems Fundamentals** (**SF**) knowledge area. Because this KU replaces the existing draft '**SF/Performance**' KU, it does not impact the number of Tier 1 hours devoted to evaluation. Our proposal just refines the focus.

This unit *replaces* the '**SF/Performance**' Core-Tier 1 KU in the current draft (p160). The rationale for the changes include the following:

- The term 'performance' can be problematic because it is often interpreted as being limited to temporal measures (elapsed time, throughput). Today, measures such as power and energy are also important.
- We believe that it is important to introduce the principles of evaluation, from which specifics (such as concrete metrics and benchmarks) can be derived. For this reason we introduce the scientific method and the broad framework of experimental evaluation.
- We believe that it is important to explicitly motivate the importance of sound quantitative evaluation. This is influenced by our view that the discipline of computer science suffers from the absence of a well established culture of sound quantitative evaluation.

- The EVALUATE community has developed a framework for understanding the pitfalls of quantitative evaluation. Insights from that process have influenced the development of the new KU below. The framework is currently available as a technical report. Amer Diwan presented many of these ideas in his PLDI Keynote in June 2012, which was enthusiastically received by the PLDI community.

SF/Quantitative Evaluation Principles
[3 Core-Tier 1 hours]
[Cross-reference PD/Parallel Performance]

Topics:

- Importance of quantitative evaluation techniques for all computer science graduates
- The Scientific Method, as applied to experimental evaluation in computer science
  - Formulation of experimentally testable hypotheses
  - Choosing and understanding metrics
  - Choosing and understanding workloads
  - Choosing and understanding measurement contexts
- Basic experimentation techniques
  - Experimental design: dependent, independent, and control variables
  - Data collection
- Understanding experimental data
  - Data analysis and interpretation
  - Data presentation

Learning Outcomes:

1. Construct an example that demonstrates the consequences of poor experimental methodology.
2. Given a measurement describe how it may produce an observer effect.
3. Given a concrete hypothesis about a computer system, design an experiment to test it.
4. Given a hypothesis and some experimental data, interpret and present the results.
5. Conduct an experiment to explore the effect of a parameter's value.

We also propose the following *new* Tier-2 knowledge unit for **Systems Fundamentals** (**SF**). This unit extends the Tier-1 '**SF/Evaluation Principles**' knowledge unit listed above, with more depth and a stronger focus on the mechanics of evaluation. This unit includes all of the elements of the draft SF/Performance KU that were not included in our revised Tier 1 KU above.

SF/Quantitative Evaluation Processes
[3 Core-Tier 2 hours]

[Cross-reference AL/Basic Analysis]

Topics:

- Analytical tools to guide quantitative evaluation
  - Use of simple (e.g., back-of-the-envelope) calculations to identify spurious results
  - Amdahl's Law
  - Order of magnitude analysis (e.g., using Big O notation)
  - Analysis of slow and fast paths of a system
  - Constant factors
- Common metrics: why they are useful, when they should be used, and how to compute them
  - Time: e.g., Response time, Throughput
  - Energy: e.g., Joules, Performance Per unit Energy (PPE)
  - Events: e.g., Retired instructions, Cache misses, Page faults
- The pitfalls of evaluating layered systems: how layers can obfuscate and confound evaluation
  - Understanding layered workloads: for example, understand why high level languages are harder to correctly evaluate because optimization and virtualization can substantially affect the behavior of a workload.
  - Understanding layered platforms: understand why platforms are hard to evaluate correctly without a solid understanding the behavior of each of the layers, including features such as schedulers, caches, and branch predictors.
- Threats to validity of experimental evaluation
  - Correctness comes before measurement
  - Micro-benchmarking pitfalls

Learning Outcomes:

1. Use limit studies or simple calculations (e.g., back-of-the-envelope-calculations) to produce order-of-magnitude estimates for a given metric in a given context.
2. Explain the circumstances in which a given metric is useful.
3. Explain how a given layer of the system stack (software or hardware) can affect a program's performance.
4. Give an example of a metric that would be inappropriate if you do not understand some detail of an underlying layer.
5. Given an experimental design, explain the threats to validity of its results.

CS2013 omits some characteristics of graduates with respect to evaluation that were in 2008 and previous documents (e.g., under the heading of *Practical capabilities and skills relating to computer science* in Section 4.2 of CS2008). This should be remedied and updated to reflect current needs.

We propose to add the following to the list of characteristics that appear in Chapter 3, '**Characteristics of Graduates**'. The text should be added to page 20, before line 38.

Evaluation

A fundamental skill in computer science is knowing how to design experiments for evaluating computer systems. Therefore graduates need to have a basic understanding of how to design
such experiments, conduct the experiments to collect data, and analyze and interpret the data.

In addition to the above changes, all cross-references to *SF/Performance* should be updated:

- p115, line 235, cross reference to *SF/Performance/Figures of performance merit* should be changed to *SF/Evaluation Principles/Importance of quantitative evaluation techniques*.
- p115, line 237, cross references to *SF/Performance/Figures of performance merit* should be changed to *SF/Evaluation Principles/Basic experimentation techniques*.
- p120, line 53, cross reference to *SF/Performance* should be changed to *SF/Evaluation Principles* and *SF/Evaluation Processes*.
- p124, line 163, cross reference to *SF/Performance* should be changed to *SF/ Evaluation Principles*, and a cross reference to *SF/Evaluation Processes* should be added.
- p157, line 16, table entry *SF/Performance* should be changed to *SF/Evaluation Principles*
- p157, line 16, a new row should be added: *SF/Evaluation Processes*, with 3 Core-Tier 2 hours.

Contacts:
Steve Blackburn (Australian National University)
Amer Diwan (Google)
Matthias Hauswirth (University of Lugano)
Peter F. Sweeney (IBM T. J. Watson Research Center)

Co-Authors:
Jose Nelson Amaral (University of Alberta)
Walter Binder (University of Lugano)
Chen Ding (University of Rochester)
Christophe Dubach (University of Edinburgh)
Yossi Gil (Technion)
Michael Hind (IBM T. J. Watson Research Center)
Doug Lea (State University of New York at Oswego)
Nate Nystrom (University of Lugano)
David J. Pearce (Victoria University of Wellington)
Alex Potanin (Victoria University of Wellington)
Shahnawaz Talpur (Beijing Institute of Technology)