Spring-11: PDC in CS1/2 and a mobile/cloud intermediate software design course

J. Kaylor • K. Läufer • C. Sekharan • G. K. Thiruvathukal
Dept. of Computer Science • Loyola University Chicago
www.cs.luc.edu • laufer@cs.luc.edu
Prior Outcomes

**Spring 2011:** three 9h PDC modules -> required second-year courses

**AY 2011-12:** four 9h advanced PDC course modules in programming and distributed computing -> electives

Institutional Profile

- Urban, private, Jesuit, liberal arts, ~16k students
- College of Arts and Sciences, ~8k students
- Department of Computer Science, ~250 students
- ~10 FTE: 9 CS, 1 bioinf (1/2 FTE), 1 math (1/4 FTE CS)
- ~140 undergrad majors in CS, SE, IT, Networks/Sec, Bioinf
- ~110 master's students in CS, SE, IT
- External funding: NSF S-STEM, NSF BPC lead institution, NSF research grants, industry grants and donations
Abstract/Current Focus

**AY 2012-13:** We are moving PDC topics further down into CS1 and CS2, fleshing out PDC coverage in our intermediate object-oriented development course (CS 313), and stepping up evaluation.

Position Statements

[for discussion]

- To teach PDC topics effectively, they should not be taught in isolation. Instead, they should be taught in the context of relevant software engineering best practices. (Go beyond functional and performance requirements.)
- Mobile applications backed by RESTful cloud services are rapidly emerging as the mainstream paradigm of computing [Christensen, OOPSLA '09]. PDC curricula should embrace it.
**CS1**

- Spring 2013: separate section aimed at majors in physics and other hard sciences
- includes some material on numerical methods at the K and C levels
- about 9 class hours were dedicated to sequential and parallel versions of these algorithms and the possible resulting speedup using data parallelism in C#
- *example: threads for speeding up trapezoidal rule integration*

**CS2**

- emphasis on PDC topics in CS2 starting in fall 2011
- 9-hour PDC module on task parallelism, speedup, and load balancing in algorithms involving arbitrary precision arithmetics
- presentation at C/A levels in the form of various examples
- *example: compute Fibonacci numbers based on repeated squaring of 2-by-2 matrices of BigIntegers in Java*
  - experiment with the speedup and load balancing btw. tasks
Intermediate Object-Oriented Development (CS 313)

• since fall 2011: C#, emphasis on PDC topics
• since fall 2012: Java + Android as context for concurrent/distributed comp. topics (18h C/A)
  • concurrency and coordination
  • design forces: safety, liveness, performance
  • external events and internal timers
  • background threads
  • progress reporting and cancelation
  • offloading computation to cloud

```java
// local task: i is number to check
final long half = i / 2;
for (long k = 2; k <= half; k += 1) {
    if (isCancelled()) break;
    publishProgress((int) ((k / half) * 100));
    if (i % k == 0) return false;
}
return true;
```
**Eval: Course Effectiveness**

Sample questions (Likert scale): Rate BEFORE/AFTER taking course
- your event-based programming expertise
- your thread-based programming expertise
- your feeling of preparedness for the job market
- ...

**Eval: Learning Assessment**

Suppose we have 2 philosophers. The 1st one, Kant, repeatedly behaves like so: think for 10 minutes
...

**Future Plans**

- **Computer Systems/CS3**: PDC module with architecture, programming, and cross-cutting
- **Algorithms/CS7**: PDC module: models of computation and complexity, basic algorithmic paradigms, specific problems and their algorithmic solutions
- **Evaluation**: longitudinal; work with Loyola’s Center for Sci & Math Education, TCPP, fellow adopters
- **Dissemination**: workshops for other adopters in the Midwest