Getting Started in Programming Languages Implementation Research

Cristina Cifuentes
Sun Labs
Overview

• About Cristina
• Projects
• Lessons Learned
• Overall Advice
About Cristina – Birth to PhD
## Projects

<table>
<thead>
<tr>
<th>Year</th>
<th>Project</th>
</tr>
</thead>
<tbody>
<tr>
<td>1991-94</td>
<td>Decompilation</td>
</tr>
<tr>
<td>1995-99</td>
<td>Retargetable static binary translation</td>
</tr>
<tr>
<td>2000-01</td>
<td>Dynamic binary translation</td>
</tr>
<tr>
<td>2002-03</td>
<td>Verilog compiler for massively parallel machine</td>
</tr>
<tr>
<td>2004-06</td>
<td>Compilation techniques for small Java virtual machine</td>
</tr>
<tr>
<td></td>
<td>Project management wireless sensor project</td>
</tr>
<tr>
<td>2007</td>
<td>Static program analysis for security vulnerability</td>
</tr>
</tbody>
</table>

© 2007 Sun Microsystems, Inc.
Walkabout – Dynamic Binary Translation

- 10 KLOC
- Reused SPARC, x86 instruction specs
- Instrumentation spec 247 LOC
- Several tools: disassemblers, emulators, instrumented interpreters, binary rewriters, binary translators
- 1.2 staff-years over 9 months, 5 diff people
- SPEC95 benchmarks (1MB) [WBT’02]
Phaser – Verilog Compiler for Massively Parallel Machine

- Phaser machine: 10 boards x 64 ASICs x 64 processors => > 40,000 procs
- Input: RTL designs
- 500 KLOC, C++
- Tools: compiler for Sparc and Phaser, emulators, visualizer
- Effort: 10-12 people, 4 years
- My contribution: partitioning of code [PACT'04]
Squawk – Small Java™ Virtual Machine Written in Itself

- 300 KLOC
- Designed for small devices
- GC and Interpreter automatically translated to C
- Effort: 2-3 people over 3 years
- My contribution: fast and compact compiler in Java
- Deployed on the Sun™ SPOT platform [VEE'06]
Sun™ SPOT – Java on Wireless Sensor Devices

- Squawk on the bare metal
- Distributed team California, Boston, UK, Australia
- Project manager/QA: build process, testing infrastructure, project schedules, task allocation, porting, ...
- Support role: hands-on lab (JavaOne'06), presentations (JavaOne'06, various universities and Java industry forums), community-based research
SPAsec – Static Program Analysis for Security Vulnerability

Too new for a project diagram!

• Analyze C source code for bugs
  > focus on security bugs
• Check systems code
  > scalability
• Techniques of interest: type analysis, abstract interpretation, model checking
• Effort:
  2 people + interns
• Reuse existing infrastructure
  > build prototype on top

© 2007 Sun Microsystems, Inc.
### Lessons Learned

<table>
<thead>
<tr>
<th>Project</th>
<th>Implementation</th>
<th># pp</th>
<th>LOC</th>
<th>input type</th>
<th>input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Walkabout</td>
<td>Reused UQBT and NJMC specs</td>
<td>2</td>
<td>40K</td>
<td>SPEC</td>
<td>&lt; KLOC</td>
</tr>
<tr>
<td>Phaser</td>
<td>Written from scratch in C++</td>
<td>15</td>
<td>1000K</td>
<td>Next gen SPARC</td>
<td>&lt; MLOC</td>
</tr>
<tr>
<td>Squawk</td>
<td>Written from scratch in Java</td>
<td>2.5</td>
<td>300K</td>
<td>Java ME</td>
<td>&lt; KLOC</td>
</tr>
<tr>
<td>Sun SPOT</td>
<td>Developed from scratch</td>
<td>10</td>
<td>500K</td>
<td>Java ME</td>
<td>&lt; KLOC</td>
</tr>
<tr>
<td>SPA</td>
<td>Reuse compiler front-end, build abstraction on top of IR</td>
<td>2</td>
<td></td>
<td>Solaris</td>
<td>&lt; MLOC</td>
</tr>
</tbody>
</table>
Lessons Learned

“Technology transfer is a contact sport.”

Bert Sutherland
Former Sun Labs Director
Lessons Learned

• Industry Research Focus
  > more applied research
    – have impact on the company
  > justify relevance to the company
  > interaction with product group essential
  > at Sun Labs, small teams
    – work with colleagues and interns
    – team size grows when/if needed for tech transfer
Lessons Learned

• Role in the project
  > Individual contributor
    – contribute to a subcomponent of the system
  > Principal investigator / chief investigator / project lead
    – determine project and direction
    – form group
    – create links with product division
    – extra admin overhead (reports, justifications, web pages, anything non-technical)
Lessons Learned

- Your group
  - the most important part for success of the project
  - like-minded people
  - need to know how to work in a group
  - agreement on choice of language and tools to use
    - be pragmatic
  - need to clearly define parts of the project
  - remove vs co-located
Lessons Learned

• Project life cycle
  > spend time on feasibility analysis and design
  > spend a large percentage of the project implementing ideas and techniques, as well as infrastructure
  > have a testing framework early in the process
  > once prototype built, rewrite parts that need rewriting or throw away prototype and rewrite
  > write the papers, file patents, update your web page
  > continue research or engineering (academia vs industry)
  > determine when project should stop
Lessons Learned

• Publications
  > Forums
    – workshops, conferences, journals
  > Impact
    – where you publish and whether you get cited
  > “Marketing”
    – web presence
    – facilitate access to publications, software artifacts
Overall Advice

• Follow your interests
• Be aware of trends
• Be recognized for something new/different
• Know when to stop
  > Keep on working while you're interested in the research
• Work in different areas
  > have depth and breadth
Overall Advice

“Be yourself and have fun”
Getting Started in Programming Language Implementation Research

Cristina Cifuentes
cristina.cifuentes@sun.com