Experience Teaching a Semester-Long Inferno Course

Phillip Stanley–Marbell
Carnegie Mellon University

Motivation & Background

- Course designed and offered in spring semester, 2004
  - Part of CMU's student-taught curriculum
  - 3 course credits versus 12 course credits for a regular course
- Goal was to try to blend theory with practice in Inferno
  - Intended to be a learning experience for both myself and the participants
  - Models of computation / process calculi (CSP, CCS, etc.)
  - Verification of concurrent programs (Promela / SPIN)
- This talk (and the paper in proceedings)
  - Details the (*eventual*) structure and content of the course
  - Documents questions posed by students during the course
  - Constructive feedback is welcome

↩

Outline

- Motivation and Background
- Course Outline
- Common Questions — Recurring Themes
- Retrospective
- Summary

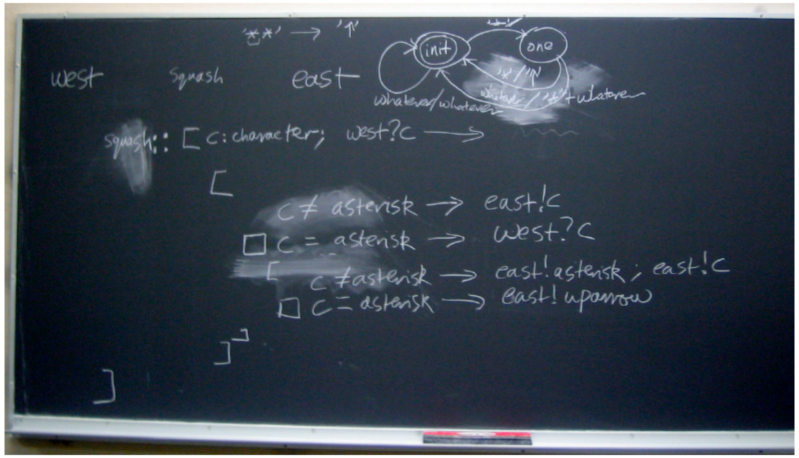
Course Outline

- 12 weeks, 2 meetings per week
- Initial class size of 8 students
- Topics:

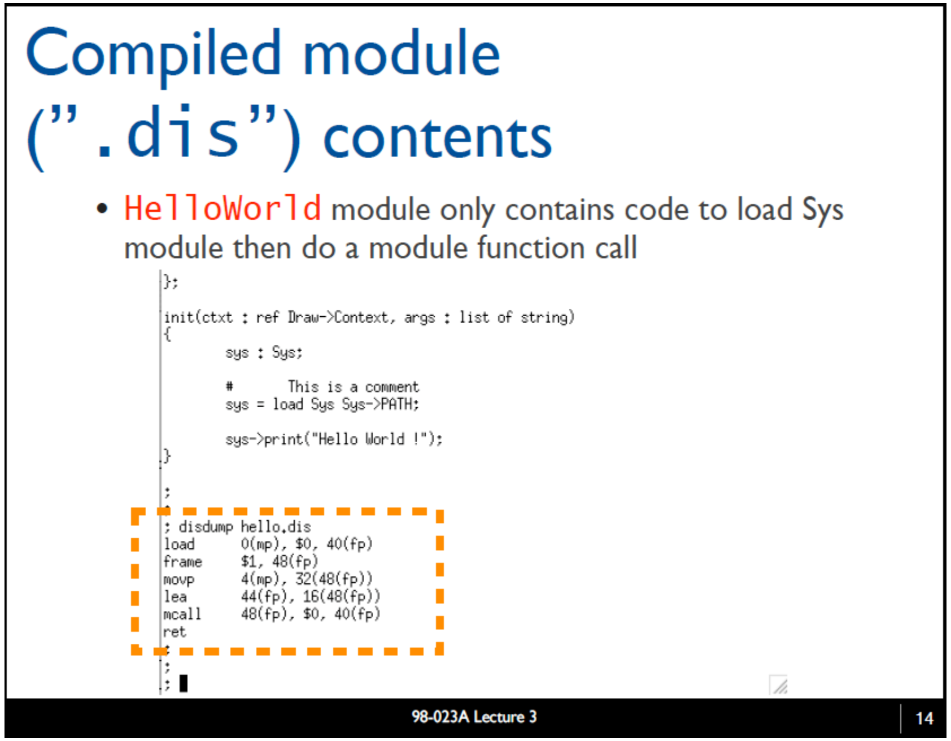
Week 1:	Introduction to Inferno; abstractions and names
Week 2:	Overview of the Limbo programming language
Week 3:	Data types in Limbo; the Dis Virtual Machine
Week 4:	Inferno kernel overview
Week 5:	Inferno kernel device drivers
Week 6:	Break
Week 7:	C applications as resource servers; built-in modules and device drivers
Week 8:	Case study
Week 9:	Platform independent interfaces; Limbo GUIs
Week 10:	Programming with threads; CSP
Week 11:	Debugging concurrent programs; Promela and SPIN
Week 12:	Factotum, Secstore and Inferno's security architecture

Course Structure

- Presentation slides & chalkboard discussions
- Pictures of chalkboard merged with slides after each lecture
  - Course content maintained in the university-wide “Blackboard” web interface (http://www.cmu.edu/blackboard)


'**' → '↑'
init
one
west
squash
east
whatever/whatever
'*'/'↑'
whatever / '*' + whatever
squash:: [c: character; west?c →
[
c ≠ asterisk → east!c
□ c = asterisk → west?c
[
c ≠ asterisk → east!asterisk ; east!c
□ c = asterisk → east! uparrow
]
]
]

+



Compiled module (".dis") contents

- HelloWorld module only contains code to load Sys module then do a module function call

```
};

init(ctxt : ref Draw->Context, args : list of string)
{
        sys : Sys;

        #       This is a comment
        sys = load Sys Sys->PATH;

        sys->print("Hello World !");
}

;
;
; disdump hello.dis
load     0(mp), $0, 40(fp)
frame    $1, 48(fp)
movp     4(mp), 32(48(fp))
lea      44(fp), 16(48(fp))
mcall    48(fp), $0, 40(fp)
ret
;
;
;
```

98-023A Lecture 3 | 14

Outline

- Motivation and Background
- Course Outline
- Common Questions — Recurring Themes
- Retrospective
- Summary

Participant's Questions

- Many are questions you've certainly heard before
  - However, they keep coming up repeatedly over the years
  - The answers are often straightforward
  - It's interesting however, how people get stuck on / turned off by these issues
  - It may be worthwhile to ponder why
- It would be interesting to hear your opinions
- I'll go over 10 of them here; more in the paper
  - The questions are in chronological (course topic) order
  - You can interpret them differently if you wish!

10 Questions

1. How many people use Inferno ?

2. What kinds of applications is it good for ?

3. How do resources as files in Inferno differ from `/dev` & `/proc` in Unix?

4. Can you do pattern matching on Limbo tuples?

5. Is string indexing into the string’s UTF-8 bytes, or into the `int` characters

↩

10 Questions

6. Can you assign to a function member within an ADT?
   (Syntax seems to suggest you can?)

7. Can the emulator be compiled with GCC?

8. ... with Cygwin under windows?

9. Why can’t the native kernel be compiled with GCC?

10. When to use a built-in module versus device driver interface to C code?

Outline

- Motivation and Background
- Course Outline
- Common Questions — Recurring Themes
- Retrospective
- Summary

↩

What could be done differently?

- Content
  - Content strayed significantly from initial objectives; turned out to be more fun
  - Use construction of distributed applications as main theme ?
  - Focus more on concurrency and verification of concurrent applications?
- Structure
  - Course was ¼ of a normal course load; current content was already too much
  - If offered as a regular course, cram 12 weeks into 1 month and focus on projects ?
- Suggestions ?

Outline

- Motivation and Background
- Course Outline
- Common Questions — Recurring Themes
- Retrospective
- Summary

Summary

- A 12 week Inferno course
  - Held in the spring semester of 2004, as part of CMU’s student-taught curriculum
  - Inferno concepts, Limbo language, emulator & native kernel, Promela/SPIN, CSP
- All course material (.pdf) will be on the web this week
  - `http://www.ece.cmu.edu/~pstanley/98-023`
  - Might be nice to convert into a wiki? so people can comment/correct/extend

Thanks

Backup Slides ↩