Hedgehog

A tiny Lisp for embedded applications

Kenneth Oksanen <cessu@iki.fi>
Lars Wirzenius <liw@iki.fi>

FOSDEM 2005

Never underestimate the power of a small tactical Lisp interpreter

A typical Oliotalo system

- Embedded boxes gather data, send to server
- Users interact with server
  - One server, many embedded boxes
- Communication via SMS, GPRS, Bluetooth, ...
  - Anything wireless
  - Low assumptions: not connection oriented, high failure rate, low bandwidth, expensive

A typical box for Oliotalo

- Not puny, not studly
  - ARM7 or better
  - Usually at least 256 kB memory
  - Flash or other persistent storage
- Most common model in 2002: operating system and application code statically linked together, flashed to box as an atomic unit
  - Big problems when application is updated

Language requirements

- 32-bit integers
- Reasonably fast
- Simple to port/implement, easy to maintain
- Easy to bind to operating system services and otherwise support everything the box can do
- App development efficient: high level language
  - Garbage collection!

Why Lisp?

- Simple syntax, simple semantics, very powerful
  - easy to implement, little code
  - gc, functional programming
  - Lasu had been reading Paul Graham...
- Our own implementation
  - easy to adapt to hardware, operating system
  - easier to achieve small size, speed
  - some NIH involved, admittedly

The timeline

- Fall of 2002: conception, prototype, first real use
  - concept is good, implementation is slow
- Late spring 2003: re-implementation
  - Cessu brought in, 600 times faster
- Summer 2003 - now: production use, tweaking
  - mostly adaptation to platform, language itself worked the first time
  - however, adaptation is hard, ugly work
- Winter 2005: free release 2.0

Current implementation

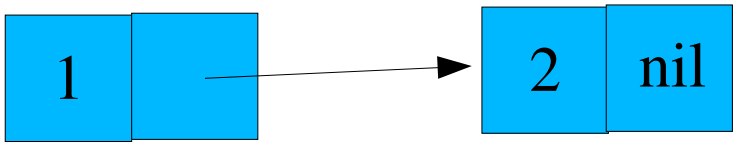
- Compiler on desktop
  - no linking, compiles full library every time (1 sec)
- Byte code file
  - very compact coding, specific to interpreter version (no codes for things not in a particular box)
- Byte code interpreter, desktop and box
  - simple stop&copy two semispace garbage collection
- Library
  - completely separate from interpreter

Example: hello, world

```
(pr "hello, world")
```

Language features

- Simple data types: 32-bit integers, strings, symbols
- Constructs: singly linked lists, tuples, AVL trees
  - lists constructed from cons cells (value pairs, 2-tuples)



- Lisp syntax: `(funcname arg1 arg2 arg3)`

Example: List length (bad)

```
(def (list-len list)
     (if (nil? list)
         0
         (+ 1 (list-len (cdr list)))))
```

Looping via recursion.
Tail recursion is removed, no speed/space penalty.

Example: List length (good)

```
(def (list-len list)
    (def (helper list length)
        (if (nil? list)
            length
            (tailcall (helper (cdr list)
                              (+ length 1)))))
    (helper list 0))
```

Look at those parentheses!

Example: Unit testing

```
(fail-unless-equal (list-len nil) 0)
(fail-unless-equal (list-len '(1)) 1)
(fail-unless-equal (list-len '(1 2)) 2)
(fail-unless-equal (list-len '(1 2 3)) 3)
```

Technically "function testing" since Hedgehog does not have modules as such.

The library

- Stuff that is useful to many applications
- String manipulation
- Simple math
- Lists, dictionaries/maps (via AVL trees)
- State machines, for simulating threads/tasks
  - No locking, less error prone than real threads
- More to be added as needed
  - Oliotalo has some private ones

Experiences

- Language design is fun
  - implementation mostly easy, too, the real work is supporting everything the platform provides
- Interpreter should be a thin layer
  - thin means less can go wrong
  - nice abstractions built with Lisp
  - repeatedly learned

Experiences

- Simulation environment on desktop is really nice
  - easier, faster to test and debug
  - programs run faster, too
- Lisp programming is easy, avoids typical C errors
  - has errors all its own: dynamic type checking...

Experiences

- Customers really love the flexibility to change things later
  - on the other hand, they *will* make use of this...
  - customers want early prototypes on-site, leading to many amusing anecdotes about debugging (axles, LED colors, wireless networks, ...)
  - easy to impress with fast turnaround

Future

- Maintain momentum
- Compiler optimizations
- Libraries
- Applications
- Ports
- Static typing?

Thank you

http://hedgehog.oliotalo.fi

hedgehog@hedgehog.oliotalo.fi