Lisp and Mac OS X

October 2003, ILC 2003
Rainer Joswig, joswig@lisp.de

- Mac OS X is a very popular desktop OS
- Mac OS X and Xserve are very cost/space effective servers
- Powerbook G4 is my personal Lisp workstation of choice
- PowerMac G4 (G5) with Cinema Display makes developing Lisp code fun
- any eMac/iMac/iBook is enough for Lisp development and Lisp applications
- a multitude of Lisp systems is available for Mac OS X
Mac OS X

- Mac OS 9 is history. Rich environment on an unstable OS. Great UI. I don’t have it anymore installed.
- Mac OS X is based on Mach, Unix, Carbon, Cocoa, Java, FreeBSD, OpenGL. Stable. Fast. Great UI.
- Darwin is an open source version of the core Mac OS X. Runs on PowerPC and x86.
- Server version of Mac OS X is available
- compatible to a wide variety of standards.
Mac OS X 10.2 “Jaguar”

• Quartz Extreme (OpenGL accelerated compositing of Windows)
• many speed improvements
• first really “usable” version of Mac OS X
Mac OS X 10.3 “Panther”

- lot’s of UI improvements
- GCC 3.3
- XCODE IDE
- new Finder
- early support for PowerMac G5
- appears 24th October 2003

http://www.apple.com/macosx/
Mac OS X system architecture
Carbon

- Port of the Mac OS 9 libraries to Mac OS X. Simplifies porting of Mac OS 9 software to Mac OS X.
- some functionality has been removed or changed
- some extensions
- Macintosh Common Lisp is a Carbon application

Cocoa

- the new Mac OS X libraries
- based on Objective C
- GC: Reference Counting
- Message Passing
- Cocoa-based applications can use Carbon
- OpenMCL and LispWorks can use Cocoa

new in Mac OS X

• Unix (FreeBSD 4.4) with Mach Kernel
• command line interface, shell
• multiple users (with live switching)
• long pathnames
• sockets
• OpenGL/Quartz/Aqua User Interface
• memory protection
• preemptive multi-tasking, threads
• multi-processor support
• Apache, sendmail, GCC 3.3, ...
• journalling filesystem
still in Mac OS X

- Unicode
- AppleEvents
- OpenGL
- AppleScript
- HFS+
PowerPC generations

- G3, power efficient, great for Laptops, runs Lisp very nicely
- G4, new versions are also usable for Laptops, runs Lisp very nicely, AltiVec extension (only some support by Lisp implementations)
- G5, not yet usable for Laptops, 64bit processor, currently many Lisp implementations seem to have problems (code runs slow)

http://www.apple.com/g5processor/
Common Lisp

- Lisp started in 1958, introduced many concepts (many still to enter mainstream)
- Evolution: ANSI Common Lisp
- based on decades of experience
- some very large applications (> 1 million LOCs) and some extremely large applications (like Cyc)
- advanced Object System (CLOS) -> OODL (Object-Oriented Dynamic Language)
extremely powerful Object-oriented dynamic language (OODL)

- CLOS dynamic objects allow changes at any time (read, compile, load, run)
- first class metaobjects (class, function, method, metaclass, ...)
- add/delete/redefine class, objects can change their classes
- change inheritance of classes
- classes are instances of metaclasses
- add/remove/redefine slots
- add/delete/redefine methods and functions
- method combination, multi-methods, multiple-inheritance
- methods for objects
- introspection, meta object protocol (MOP)
- compiler and debugger available at runtime, load code at runtime
Common Lisp history on the Mac

• available on the Mac for many years
  – Pearl Lisp, Coral Lisp, Coral Common Lisp, Macintosh Allegro Common Lisp, Macintosh Common Lisp, OpenMCL
    • once owned by Apple for a few years
    • used for a lot of applications (Apple Dylan, Sk8, OpenMusic, Symbolic Composer, Igor Engraver, ...) and teaching CS
  – PowerLisp (obsolete)
  – Procyon Common Lisp (obsolete)
  – Exper Common Lisp (obsolete)

http://openmcl.clozure.com/
http://www.digitool.com/
http://www.cormanlisp.com/PowerLisp.html
MCL 5.0

- from Digitool, http://www.digitool.com/
- „carbonized“ port of the popular Macintosh Common Lisp
- final version since mid 2003
- compiles to native PowerPC code and can generate applications
- Integrated Development Environment with Interface Builder and Editor (FRED)
- supports cooperative threads, OpenTransport, AppleEvents, ...
- CLIM 2.0 as option
MCL 5.0 (2)

- lot’s of libraries (Quicktime, OpenGL, …)
- still also runs under Mac OS 9
- still: old event loop, short pathnames, no Cocoa support, …
- http://www.digitool.com/
MCL 5.0
Allegro Common Lisp 6.2

• from Franz Inc., http://www.franz.com/
• port of the widely available Unix version
• compiles to native PowerPC code
• Does not have a Mac GUI or IDE
  – IDE is Emacs + ELI + Composer (X11)
• evaluation version for Mac OS X is available
Allegro Common Lisp 6.2
OpenMCL 0.13.6 / 0.14 alpha

- from Clozure, based on MCL from Digitool, http://www.clozure.com/
- initially developed for NASA by Gary Byers for server and embedded use (→ small footprint)
- solid free and open source core of Macintosh Common Lisp
- currently under active development by Gary Byers
- also runs on PPC Linux, older ports for SPARC/Solaris and QNX
- compiles to native PowerPC code
- usually used with Emacs/ilisp
OpenMCL 0.13.6 / 0.14 alpha

- supports threads, sockets (ACL API), CLX, ...
- (simple) Cocoa-IDE is under development
- can be used for double-clickable applications under Mac OS X (example Alpaca)
- OpenMCL 0.14 alpha
  - native threads with support for multiple processors
  - MOP
- a CLX version also runs McCLIM
- work in progress
- EGC does not work yet
OpenMCL

This shows a bit of OpenMCL. OpenMCL is run via a carbonized Emacs with lisp. From within an Emacs OpenMCL listener we have started the Cocoa IDE. At the same time the Interface Builder application of MacOS X 10.2.2 edits the NIB description of the OpenMCL editor window. OpenMCL itself uses those descriptions and can talk to Cocoa objects in an Objective-C-like syntax.
Mikel Evin’s Alpaca

Using Digest Authentication as a SASL Mechanism

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the “Internet Official Protocol Standards” (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2000). All Rights Reserved.

Abstract

This specification defines how HTTP Digest Authentication [Digest] can be used as a SASL [RFC 2222] mechanism for any protocol that has a DAV profile. It is intended both as an improvement over GTM/MDS [RFC 2195] and as a convenient way to support a single authentication mechanism for web, mail, LDAP, and other protocols.

Table of Contents

1
LispWorks 4.3.6

- from Xanalys, http://www.lispworks.com/
- very feature-complete Lisp implementation, also available for Unix/Linux and Windows
- compiles to native code
- LispWorks for Mac OS X Personal Edition available now
- comes in three flavors:
  - command line
  - CAPI via Cocoa
  - X11/Motif version, also supports CLIM 2.0
- Editor provided in source
- Extensive IDE
LispWorks 4.3.6

- lacks support for Mac libraries (Quicktime, MIDI, CoreAudio AppleEvents, ...)
- no CLIM for Cocoa-LispWorks
- CAPI is native on the Mac and cross-platform (Unix, Linux, Windows, Mac)
LispWorks 4.3.6
LispWorks 4.3.6

Mouse-L spins the object, Mouse-R moves the light, Shift Mouse-L moves your view.

(defun gl-vertexes (contents)
  (mapcar "#(lambda (c) (apply 'gl-double-vector c) contents))

(defstruct xyz (x 0.0 0.0000000 0.0000000)
  (y 0.0 0.0000000 0.0000000)
  (z 0.0 0.0000000 0.0000000))
LispWorks 4.3.6
CLISP 2.31

- portable open source Common Lisp under active development
- written mostly in C
- single-threaded, byte-code compiler
- can be used for scripting
- getting a working/compilable version can be difficult
- http://clisp.cons.org/
ThinLisp

- originally from Gensym, now Open Source
- small CL subset used for delivery and real-time programming
- compiles to C
- uses other Lisps as a host
- http://www.thinlisp.org/
- no closures, no GC
Embeddable Common Lisp (ECLS)

- Current version 0.8
- compiles to C
- ported to Mac OS X, has some rough edges
- http://ecls.sourceforge.net/
- embeddable
ECLS embedded in Quake
Steel Bank Common Lisp (SBCL)

- Current version 0.8.4.x
- forked from CMUCL, simplified build
- can be compiled with some other Common Lisp implementations like OpenMCL
- sophisticated native code compiler
- type inference and type checking
- no threads on generational GC in the current port, but in the works
- http://sbcl.sourceforge.net/
• GCL, GNU Common Lisp
  – not yet fully ported, but people are working on it.

• Poplog ?

• old Xlisp, originally from David Betz
  – XLispStat, Vista
# Feature comparison

<table>
<thead>
<tr>
<th></th>
<th>ACL</th>
<th>LispWorks</th>
<th>MCL</th>
<th>CLISP</th>
<th>OpenMCL</th>
<th>SBCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDE</td>
<td>commercial</td>
<td>commercial</td>
<td>commercial</td>
<td>modified GPL</td>
<td>free</td>
<td>free</td>
</tr>
<tr>
<td>UI support</td>
<td>Emacs, ELI, Composer</td>
<td>Native</td>
<td>Native</td>
<td>Emacs, ILisp</td>
<td>Emacs, ILisp</td>
<td>Emacs, ILisp</td>
</tr>
<tr>
<td>use in shell</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>compiler</td>
<td>native</td>
<td>native</td>
<td>native</td>
<td>byte-code</td>
<td>native</td>
<td>native</td>
</tr>
<tr>
<td>threads</td>
<td>Lisp</td>
<td>Lisp</td>
<td>cooperative</td>
<td>no</td>
<td>native</td>
<td>no</td>
</tr>
<tr>
<td>GC</td>
<td>generational</td>
<td>generational</td>
<td>EGC, generational</td>
<td>?</td>
<td>generational</td>
<td>mark &amp; sweep</td>
</tr>
<tr>
<td>delivery</td>
<td>?</td>
<td>images, applications</td>
<td>images, applications</td>
<td>?</td>
<td>images, applications</td>
<td>?</td>
</tr>
<tr>
<td>CLIM</td>
<td>yes</td>
<td>yes (X11)</td>
<td>yes</td>
<td></td>
<td>McCLIM</td>
<td></td>
</tr>
<tr>
<td>Price</td>
<td>$ ?</td>
<td>$999 PE, $2999 EE</td>
<td>$750</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Benchmarks

- ran on a PowerBook G4, 800Mhz, 1 GB RAM
- no surprise: native code compilers are the fastest
- overall fastest is LispWorks, followed by ACL and SBCL
- ACL has only a few problems with some benchmarks
- SBCL needs some speed improvements for CLOS
- CLISP has very fast bignum routines
- OpenMCL’s compiler generates compact code, but isn’t very sophisticated otherwise
<table>
<thead>
<tr>
<th>Benchmark</th>
<th>LispWorks 4.3</th>
<th>Allegro CL 6.2</th>
<th>SBCL 0.8.3.38</th>
<th>OpenMCL 0.14a</th>
<th>CLISP 2.31</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOYER</td>
<td>5.13</td>
<td>3.86</td>
<td>1.69</td>
<td>3.75</td>
<td>21.31</td>
</tr>
<tr>
<td>BROWSE</td>
<td>0.72</td>
<td>0.77</td>
<td>0.86</td>
<td>1.41</td>
<td>4.25</td>
</tr>
<tr>
<td>DDERIVIV</td>
<td>0.9</td>
<td>0.96</td>
<td>0.89</td>
<td>1.76</td>
<td>7.19</td>
</tr>
<tr>
<td>DERIV</td>
<td>1.57</td>
<td>1.64</td>
<td>1.64</td>
<td>3.14</td>
<td>12.98</td>
</tr>
<tr>
<td>DESTRUCTIVE</td>
<td>0.78</td>
<td>1.1</td>
<td>1.27</td>
<td>1.53</td>
<td>10.95</td>
</tr>
<tr>
<td>DIV2-TEST-1</td>
<td>1.52</td>
<td>0.72</td>
<td>1.01</td>
<td>2.35</td>
<td>10.48</td>
</tr>
<tr>
<td>DIV2-TEST-2</td>
<td>0.77</td>
<td>0.36</td>
<td>0.53</td>
<td>1.2</td>
<td>5.23</td>
</tr>
<tr>
<td>FFT</td>
<td>0.18</td>
<td>0.12</td>
<td>0.17</td>
<td>7.73</td>
<td>16.38</td>
</tr>
<tr>
<td>FRPOLY/FIXNUM</td>
<td>1.12</td>
<td>1.04</td>
<td>0.77</td>
<td>1.25</td>
<td>9.96</td>
</tr>
<tr>
<td>FRPOLY/BIGNUM</td>
<td>2.61</td>
<td>1.86</td>
<td>1.68</td>
<td>2.82</td>
<td>8.69</td>
</tr>
<tr>
<td>FRPOLY/FLOAT</td>
<td>1.15</td>
<td>1.08</td>
<td>0.78</td>
<td>1.79</td>
<td>6.68</td>
</tr>
<tr>
<td>PUZZLE</td>
<td>0.6</td>
<td>5.73</td>
<td>0.33</td>
<td>5.71</td>
<td>17.11</td>
</tr>
<tr>
<td>CTAK</td>
<td>2.12</td>
<td>2.1</td>
<td>1.06</td>
<td>2.17</td>
<td>17.88</td>
</tr>
<tr>
<td>TAK</td>
<td>1.23</td>
<td>0.94</td>
<td>0.78</td>
<td>1.54</td>
<td>22.93</td>
</tr>
<tr>
<td>RTAK</td>
<td>1.22</td>
<td>0.92</td>
<td>0.78</td>
<td>1.52</td>
<td>22.93</td>
</tr>
<tr>
<td>TAKL</td>
<td>1.33</td>
<td>1.58</td>
<td>1.2</td>
<td>4.52</td>
<td>31.58</td>
</tr>
<tr>
<td>STAK</td>
<td>1.54</td>
<td>10.98</td>
<td>1.25</td>
<td>3.95</td>
<td>17.29</td>
</tr>
</tbody>
</table>
## Benchmarks (II)

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>LispWorks 4.3</th>
<th>Allegro CL 6.2</th>
<th>SBCL 0.8.3.38</th>
<th>OpenMCL 0.14a</th>
<th>CLISP 2.31</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPRINT</td>
<td>0.71</td>
<td>1.88</td>
<td>4.91</td>
<td>4.35</td>
<td>6.63</td>
</tr>
<tr>
<td>TRAVERSE</td>
<td>4.71</td>
<td>2.86</td>
<td>4.41</td>
<td>3.92</td>
<td>55.23</td>
</tr>
<tr>
<td>TRIANGLE</td>
<td>2.38</td>
<td>4.33</td>
<td>2.07</td>
<td>3.55</td>
<td>81.87</td>
</tr>
<tr>
<td>RICHARDS</td>
<td>1.41</td>
<td>3.59</td>
<td>1.04</td>
<td>1.46</td>
<td>24.11</td>
</tr>
<tr>
<td>FACTORIAL</td>
<td>0.7</td>
<td>0.74</td>
<td>0.48</td>
<td>1.46</td>
<td>2.3</td>
</tr>
<tr>
<td>FIB</td>
<td>0.18</td>
<td>0.16</td>
<td>0.35</td>
<td>0.23</td>
<td>5.55</td>
</tr>
<tr>
<td>BIGNUM/ELEM-100-1000</td>
<td>1.28</td>
<td>1.23</td>
<td>1.33</td>
<td>2.32</td>
<td>0.17</td>
</tr>
<tr>
<td>BIGNUM/ELEM-1000-100</td>
<td>5.58</td>
<td>4.66</td>
<td>5.0</td>
<td>10.6</td>
<td>0.51</td>
</tr>
<tr>
<td>BIGNUM/ELEM-10000-1</td>
<td>10.36</td>
<td>11.58</td>
<td>5.14</td>
<td>16.34</td>
<td>0.47</td>
</tr>
<tr>
<td>BIGNUM/PARI-100-10</td>
<td>0.2</td>
<td>0.11</td>
<td>1.52</td>
<td>2.71</td>
<td>0.1</td>
</tr>
<tr>
<td>BIGNUM/PARI-200-5</td>
<td>0.95</td>
<td>0.42</td>
<td>25.19</td>
<td>33.24</td>
<td>0.4</td>
</tr>
<tr>
<td>HASH-STRINGS</td>
<td>0.08</td>
<td>2.32</td>
<td>0.09</td>
<td>0.24</td>
<td>0.52</td>
</tr>
<tr>
<td>HASH-INTEGERS</td>
<td>0.54</td>
<td>2.9</td>
<td>0.76</td>
<td>5.9</td>
<td>1.67</td>
</tr>
<tr>
<td>BOEHM-GC</td>
<td>6.77</td>
<td>18.12</td>
<td>8.08</td>
<td>13.1</td>
<td>75.45</td>
</tr>
</tbody>
</table>
## Benchmarks (III)

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>LispWorks 4.3</th>
<th>Allegro CL 6.2</th>
<th>SBCL 0.8.3.38</th>
<th>OpenMCL 0.14a</th>
<th>CLISP 2.31</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLOS/DEFCLASS</td>
<td>0.8</td>
<td>0.66</td>
<td>8.75</td>
<td>0.49</td>
<td>1.97</td>
</tr>
<tr>
<td>CLOS/DEFMETHOD</td>
<td>0.52</td>
<td>0.63</td>
<td>19.17</td>
<td>0.69</td>
<td>0.42</td>
</tr>
<tr>
<td>CLOS/INSTANTIATE</td>
<td>2.43</td>
<td>3.35</td>
<td>16.1</td>
<td>9.59</td>
<td>9.69</td>
</tr>
<tr>
<td>CLOS/METHODCALLS</td>
<td>8.16</td>
<td>7.19</td>
<td>8.07</td>
<td>13.98</td>
<td>26.55</td>
</tr>
<tr>
<td>CLOS/METHOD+AFTER</td>
<td>4.16</td>
<td>3.59</td>
<td>16.82</td>
<td>6.12</td>
<td>15.18</td>
</tr>
<tr>
<td>CLOS/COMPLEX-METHODS</td>
<td>0.86</td>
<td>1.21</td>
<td>3.54</td>
<td>1.77</td>
<td>no value</td>
</tr>
<tr>
<td>1D-ARRAYS</td>
<td>1.6</td>
<td>4.15</td>
<td>2.85</td>
<td>1.93</td>
<td>20.59</td>
</tr>
<tr>
<td>2D-ARRAYS</td>
<td>4.58</td>
<td>7.97</td>
<td>6.55</td>
<td>no value</td>
<td>16.67</td>
</tr>
<tr>
<td>BITVECTORS</td>
<td>0.25</td>
<td>0.41</td>
<td>0.9</td>
<td>4.19</td>
<td>2.32</td>
</tr>
<tr>
<td>FILL-STRINGS</td>
<td>1.17</td>
<td>2.95</td>
<td>2.91</td>
<td>1.07</td>
<td>0.8</td>
</tr>
<tr>
<td>FILL-STRINGS/ADJUSTABLE</td>
<td>0.78</td>
<td>1.18</td>
<td>1.36</td>
<td>0.55</td>
<td>1.6</td>
</tr>
</tbody>
</table>
Emacs, Xemacs

- various versions are available
- Emacs Lisp, single-threaded, byte-code compiler
- Mac OS X 10.2.x comes with GNU Emacs 21.1.1 (terminal version)
- very good Carbon/Cocoa-versions are have appeared, http://mindlube.com/products/emacs/index.html
  - even GNUS works
- GNU Emacs 21.4 will support Mac OS X directly
- currently XEmacs only works with X11 on Mac OS X
ilisp

• free Interface for „inferior“ Lisps for Emacs
• remote Lisp runs in a listener buffer
• Source code buffers with many features
  – highlighting
  – indenting
  – M-., arglist, evaluation, compilation, …
• problem supporting multiple-threaded Lisps
• http://sourceforge.net/projects/ilisp/
Glossary

- GUI: graphical user interface
- IDE: Integrated development environment
- LOC: Lines of Code
- Lisp: Lisp is simply perfect
- CLOS: Common Lisp Object System
- CLIM: Common Lisp Interface Manager
- CAPI: Windowing Toolkit for LispWorks
- OODL: Object-oriented dynamic language
- MOP: Meta-Object Protocol