Large Scheme: A Personal View

John Cowan
<cowan@ccil.org>

Scheme 2014

Recap of R7RS-small

- Based on R5RS, but with many R6RS changes
- Case-sensitive, like many implementations
- String and character escapes
- Datum and block comments
- Datum labels
- `#true` and `#false`

Recap of R7RS-small

- R6RS-style libraries; R5RS refactored
- R6RS exception handling (but not conditions)
- `letrec*`
- `define-values`, `let-values`, `letrec-values`
- `define-record-type` from SRFI 9
- Dynamically bound parameters like SRFI 39

Recap of R7RS-small

- Numeric extensions, including optional IEEE floats
- Revised integer division routines
- Unicode semantics (but a subset is allowed)
- String comparison no longer lexicographic
- String and vector procedures matching list procedures (with start and end arguments)

Recap of R7RS-small

- Bytevectors
- Binary and textual ports
- String and bytevector ports
- Environment variables, command line, and exit status
- Time of day and run time
- Various other points

Basic WG2 process

- Proposals are put on the wiki
- When ready, the SRFI process is used to develop and evaluate them
  - Posted on the SRFI site
  - Discussed on the SRFI-specific mailing list
  - A sample implementation is required
    - Preferably a portable one
- The WG votes on adding them to R7RS-large

Existing implementations

- Chibi (small, embedded)
- Chicken (R5RS/R7RS; fast compiler to C)
- Foment (compiler and interpreter)
- Gauche (script interpreter)
- Kawa (JVM-based)
- Owl Lisp (pure functional subset)
- Picrin (lightweight interpreter)
- Sagittarius (R6RS/R7RS)

What follows is a personal view

- Not based on Working Group votes
  - Unless otherwise noted
- The WG has an indefinite membership
  - If you cast a vote on the mailing list, you're in
  - A majority of votes cast carries a motion
  - I expect people will drop in and out

Release structure

- There will be rolling releases
  - Waiting till it's done would be frustrating
- Each release will build on the last
  - Infrared Edition: Overview of Scheme (done)
  - Red Edition: Data structure libraries
  - …?
  - Ultraviolet Edition: Complete (but out of sight)

The Red Edition

- List library (SRFI 1; unanimous consent)
- String library (string slices and positions, plus parts of SRFI 13)
- Vector library (enhanced SRFI 43)
- Sorting vectors and lists (SRFI 32 revised)
- Comparators (SRFI 114)

The Red Edition

- Boxes (SRFI 111; already voted in)
- Sets and bags (SRFI 113), integer sets, character sets (SRFI 14)
- Mutable queues
- Immutable deques, sets, maps
- Immutable pairs and lists (SRFI 116)
- Enumerations and their sets and maps

The Red Edition

- Hash tables and bimaps
- Generators (Gauche) or streams (SRFI 41)
- Lazy sequences
- Immutable cyclic lists
- Run-time records (SRFI 99)

The Red Edition — Maybe

- Multi-dimensional general arrays
- Sparse vectors and maps
- Ternary search trees
- Ephemerons and weak hashtables (optional)

Typical procedures

- Constructors: make-foo, foo, foo-unfold
- Predicates: foo?, -contains?, -empty?
- Selectors: -ref, -take, -drop, -split-at
- Mutators: -adjoin!, s-set!, -delete!, -search!
- The whole foo: -length, -append, -concatenate, -reverse, -count, -copy, -zip, -unzip, foo->list, list->foo

Typical procedures

- Fold & map: -map, -for-each, -reduce
- Delete: -delete, -delete-dups
- Filter & partition: -filter, -remove, -partition
- Search: -find, -any, -every, -take-while, -drop-while
- Comparison: foo=?, foo<?, foo>?

Possible future editions

- Orange Edition: numerical libraries
- Yellow Edition: I/O
- Green Edition: syntax enhancements
- Blue Edition: ???
- ???

Stand-alone issues

- Require full numeric tower?
  - WG voted yes
    - Except for exact complex numbers
- Require full Unicode repertoire?
  - Maybe except NUL in strings
- Which R7RS-small libraries to require?
- Require multiple inexact-number precisions?
  s

Help!

- I can handle the overall process
- I can handle spec design if I have to
  - At worst, the whole thing will reflect my own prejudices
  - At best, it will be nicely consistent
- Writing the implementations is another story
  - Alex Shinn wrote Chibi Scheme during WG1
  - I'm not sure I can write all the WG2 code
  - Volunteers needed and welcome!

Where

<http://trac.sacrideo.us/wg/wiki/WG2Dockets>