Madagascar Fundamentals

Sergey Fomel

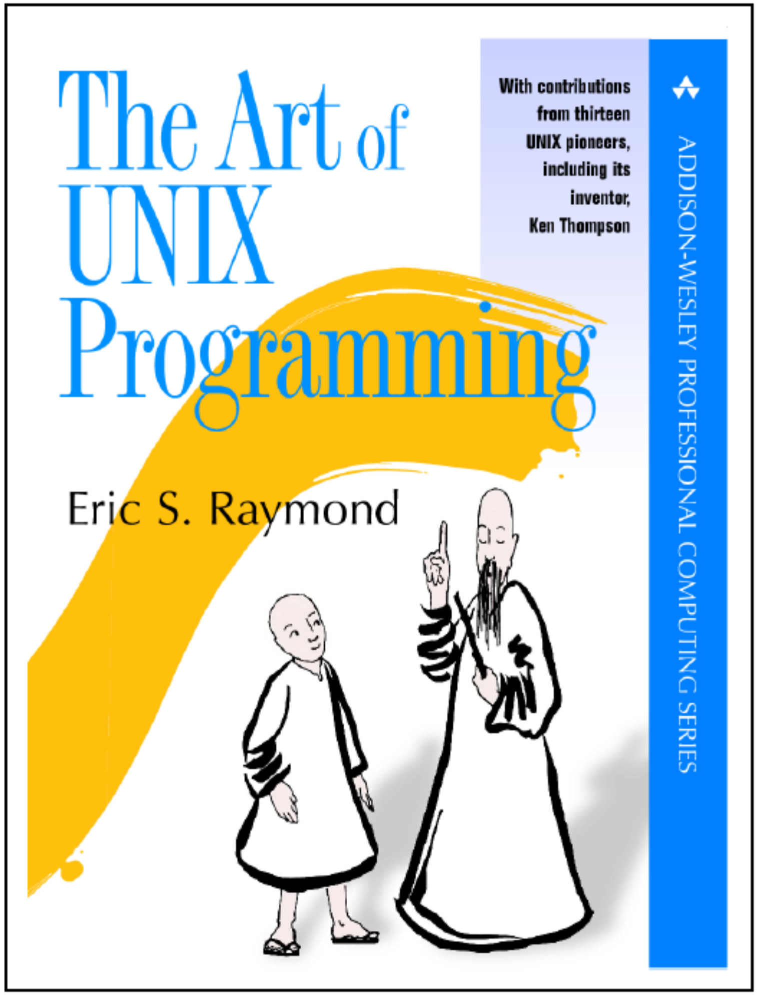
ASEG Workshop

August 15, 2013

Outline

- RSF file format
- Command-line usage
- Introduction to SCons/Python

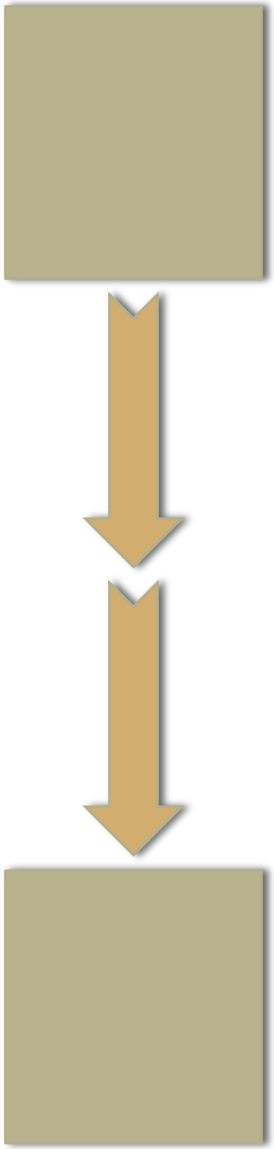
Madagascar Design Principle



The Art of UNIX Programming
With contributions from thirteen UNIX pioneers, including its inventor, Ken Thompson
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES
Eric S. Raymond

- Data arrays are file objects on disk
- “Write programs that do one thing and do it well. Write programs to work together. Write programs to handle text streams, because that is a universal interface.”

Doug McIlroy

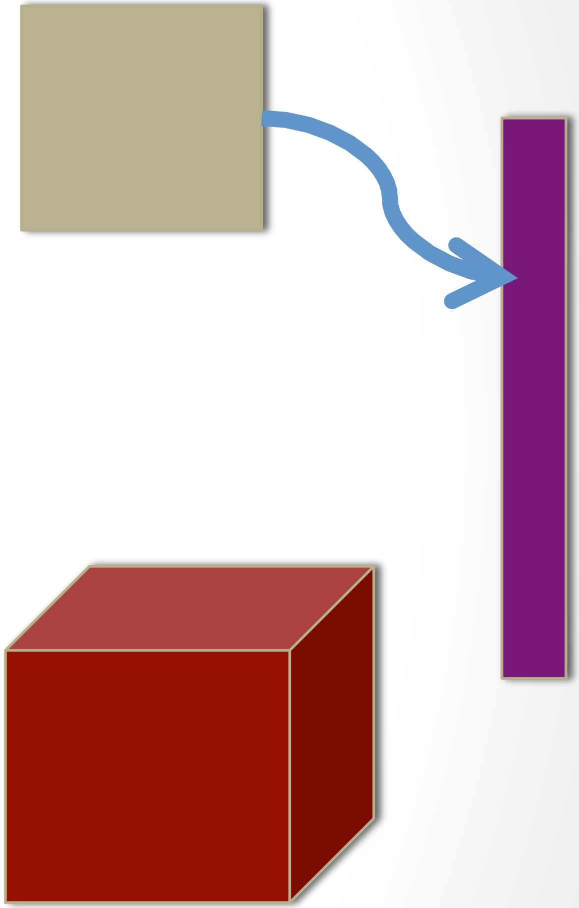


- “To design a perfect anti-Unix, make all file formats binary and opaque, and require heavyweight tools to read and edit them.”
- “If you feel an urge to design a complex binary file format, or a complex binary application protocol, it is generally wise to lie down until the feeling passes.”

Eric Raymond

RSF File Format

- Principles
- Header and data files
  - Datapath
  - Type
  - Form
  - Hypercube
- Compatibility with other formats
  - SEPlib
  - SEGY
  - SU

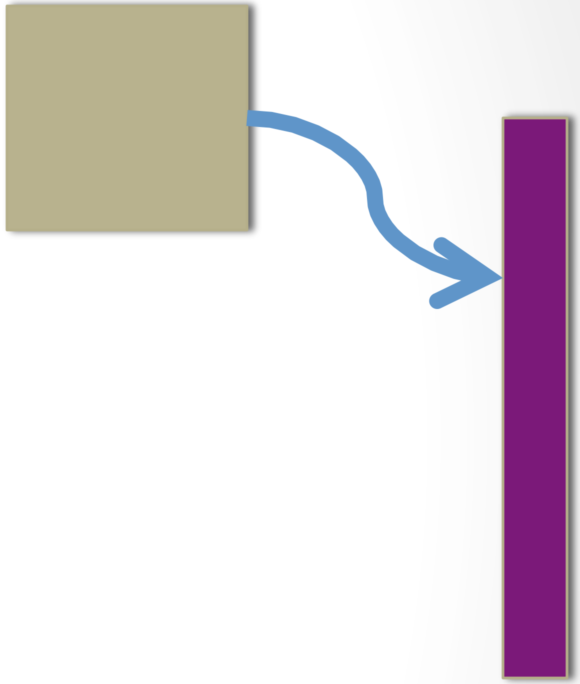


Datapath Rules

- datapath= parameter on the command line.
- DATAPATH environmental variable
- .datapath file in the current directory.
- .datapath file in the user home directory.
- Current directory.

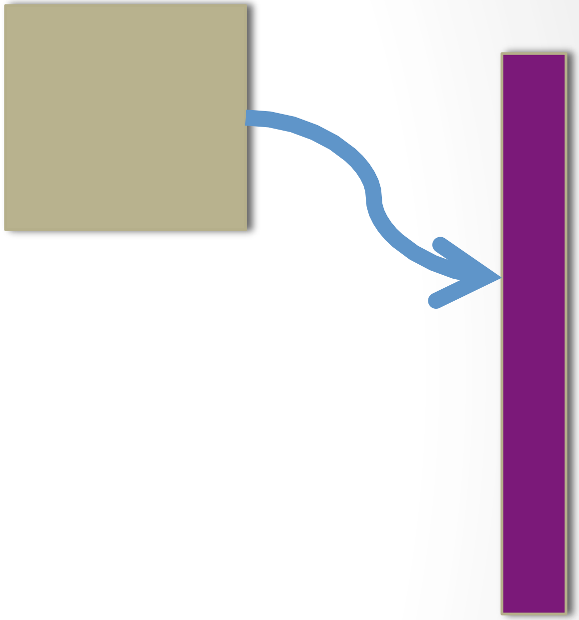
Data Type

- uchar,
- char
- int
- float
- complex
- short
- double



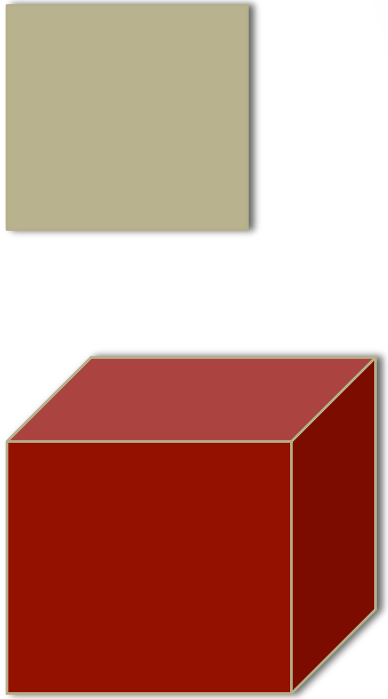
Data Form

- native
- XDR
- ASCII



Hypercube

- n1, n2, …
- d1, d2, …
- o1, o2, …
- label1, label2, …
- unit1, unit2, …
- label
- unit



Outline

- RSF file format
- Command-line usage
- **Introduction to SCons/Python**

What is SCons?

- Build system (**S**oftware **Cons**truction)
- Written in **Python**
  - Configuration files are Python scripts
- Built-in support for different languages
- Dependency analysis
- Parallel builds
- Cross-platform

Evolution of Build Systems

- **Make (1977)**
  - "Sendmail and **make** are two well known programs that are pretty widely regarded as originally being *debugged into existence*. That's why their command languages are so poorly thought out and difficult to learn. It's not just you - everyone finds them troublesome."

    Peter van der Linden
- **GNU Make (1988)**
- **SCons (2000)**

What is Python?

- **Dynamic programming language**
- **Clear, readable syntax**
  - “friendly and easy to learn”
- **Full modularity**
  - “batteries included”
- **Integrates with other languages**
  - “plays well with others”
- **Free and open-source**

Who uses Python?

- "**Python** has been an important part of Google since the beginning, and remains so as the system grows and evolves. Today dozens of Google engineers use Python, and we're looking for more people with skills in this language." *Peter Norvig*

- “**Python** is fun, free, runs on a broad range of platforms and has a large library of sophisticated modules, including numerical. It meets all our criteria for a first language.” *John Scales & Hans Ecke*

Python in 5 Easy Steps

1. Variables and strings
2. Lists and dictionaries
3. For loop
4. If/else, indentation
5. Functions and modules

1. Variables and Strings

```
>>> a=‘Melbourne’
>>> a[0]
‘M’
>>> a[:3]
‘Mel’
>>> b = a + “ rocks”
>>> print b
‘Melbourne rocks’
>>> a+2
Traceback (most recent call last):
  File “<stdin>”, line 1, module <module>
TypeError: cannot concatenate ‘str’ and ‘int’ objects
>>> a+str(2013)
‘Melbourne2013’
```

2. Lists and Dictionaries

```
>>> a = [‘Melbourne’, ‘ASEG’]
>>> a[0]
‘Melbourne’
>>> len(a)
9
>>> a.append(20)
>>> a
[‘Melbourne’,‘ASEG’,20]
>>> b = (‘Melbourne’,‘ASEG’)
>>> b.append(20)
Traceback (most recent call last):
  File “<stdin>”, line 1, in <module>
AttributeError: ‘tuple’ has no attribute ‘append’
>>> c = {‘city’: ‘Melbourne’, ‘year’: 2013}
```

3. For loop

```
>>> a = (‘Melbourne’, ‘Perth’)
>>> for city in a:
…       print city, len(city)
Melbourne 9
Perth 5
>>> for k in range(2):
…    print k, a[k]
0 Melbourne
1 Perth
>>> c = {‘city’: ‘Perth’, ‘year’: 2015}
for key in c.keys():
…    print c[key]
‘Perth’
2015
```

4. If/else, indentation

```
>>> for k in range(4):
>>>    if k < 2:
…         print k
…      else:
…         print ‘no’
0
1
no
no
>>> try:
…      a = ‘Perth’ + 2015
except:
…      print ‘error’
error
```

5. Functions and modules

```
>>> def add_5(a):
…       ‘Add 5 to input’
…       return 5+a
>>> a = add_5(3)
>>> a
8
>>> def add_b(a,b=5):
        ‘Add b to a’
…       return b+a
>>> add_b(a)
13
>>> import math
>>> math.sqrt(add_b(a,8))
4
```

SCons for Program Compilation

```
env=Environment()
env.Program(“program”, “file.c”)
```

```
scons –h
scons -n
scons -c
scons –Q
scons -j
```

Madagascar Processing: rsf.proj

- **Fetch(‘filename’, ’dirname’)**
  - A rule for downloading files from a server
- **Flow(‘target’, ‘source’, ‘command’)**
  - A rule for making target from source
- **Plot(‘target’, ’source’, ‘command’)**
  - Like Flow but generates a figure file
- **Result(‘target’, ‘source’, ‘command’)**
  - Like Plot but generates a final result

Parallel Processing with pscons

- Figures out the number of nodes/CPUs and runs **scons –j**
- Use **split=** and **reduce=** to split data for simple data-parallel processing

Summary

- **RSF** file format is simple, represents a multidimensional array as text + binary
- Madagascar programs are building blocks, can run on the command line and pipe Unix-style
- **SCons** is a powerful and convenient build system adopted by Madagascar for data processing and reproducible documents
  - Configuration files are **Python** scripts