Complete rewrites
When, why, and how?

James W. Pflugrath
Rigaku/MSC, Inc., The Woodlands, Texas, USA

Outline

• Previous software packages
• Current software packages
• When & Why
  • Whenever you get a benefit and the money to do it
  • Hardware requirements
  • Software requirements
• How
  • Modern software engineering practices in scientific programming
• Example
• Future software packages
• Bottom line
  • Whenever you get a benefit
  • Code maintenance

Previous software packages

• FRODO, Alwyn Jones, 1970’s
  • Evans & Sutherland PS300 version 1983

Current software packages

• JWP moves to Molecular Structure Corporation, 1994
  • d*TREK: device-independent diffraction image processing
    • DOE subcontract, 1994, E. Westbrook
    • Simple re-write or adaptation not possible
  • C++
    • Object-oriented programming language
    • No standard template library

When & why?

• 1994
  • Anytime! (2005)
  • Whenever a benefit or advantage arises from the re-write
    • This is always the case, you would not make a worse piece of software would you?
**When and why?**

- New programming tools
  - New languages and libraries
    - OpenGL, X Windows, OSF/Motif, Tcl/Tk
    - C++, Python
- New features
- New hardware
- New people
  - What skills do they have?
- Maintenance issues
- User issues
- Legal issues
- Who pays the bills?

**How?**

- Write a grant
- Start a company or go to work for a company
- Start a consortium
- Make your users pay
- In other words … sell it and get money

**Software engineering practices**

- Nuts & bolts
- Design beforehand
- User requirements
- Hardware requirements
- Data structures
- Algorithms
- Code management, version management
  - make, SourceSafe, cvs, bugzilla, backups
- Book: *Code Complete*
  - In the trenches: How-to
  - Variable naming, Hungarian notation

**Example: d*TREK**

- Design submitted to DOE in late 1994
  - Data objects
    - Devices
    - Source, Shutter, Goniometer, Detector, Crystal
    - Images, Reflns, Headers
    - Interprocess communication
  - Methods
    - Single objs: Goniometer move, Image write, etc.
    - Multiple objs: Find, Index, Predict, Refine, Integrate, Scale/Average
    - Reflnlist: merging, editing, sorting

**Devices and Objects**

- 1912 2005
Example: d*TREK

- User interface
  - Simple: command line arguments
  - Scripts
  - Graphical user interface helps build command lines
    - 1994: X-Windows/Motif

Scripting

- User defaults
- High throughput
- No need for GUI
- No button processing
- Customization
  - Beamline
  - Detector
  - Crystal

```
#!/bin/csh -f
set IMAGE_NAME = ../lyso12001.osc
set FIRST_IMAGE = 1
set LAST_IMAGE = 99
extractheader $IMAGE_NAME $FIRST_IMAGE $FIRST_IMAGE -seq $FIRST_IMAGE $FIRST_IMAGE -out dtfind.head
dtindex dtfind.head dtfind.ref
dtrefine dtindex.head dtfind.ref +All -go -go
dtrefine dtrefine.head -seq $FIRST_IMAGE +All -go

dtrefine dtrefine.head -seq $FIRST_IMAGE +All -go

dtintegrate dtrefine.head -seq $FIRST_IMAGE $LAST_IMAGE -profit -window 0 0 -batch 1 4
dtscaleaverage dtintegrate.head dtprofit.ref -sign -arcsinmodel -reject .0075 -batchscale -reqab spherical 4 4
dtscale.ref
```

GUI to control subprocesses
- Master scripter
- Flow chart
  - Manual
  - Auto strategy
    - one button screening
  - Auto processing
    - one button processing

Classes and objects of the d*TREK toolkit

- Class Cimage
  - Data and methods for a 2D diffraction image
  - Several constructors
  - nRead(), nWrite(), fGetPixel(), nSetPixel(), nGetRect(), etc.

- Class Cdetector
  - Cspatial, Cnonunf, CGoniometer
  - Cspatial::nPxToMM(), ::nMMToPX()

- Class Crefln
  - nGetHKL(), nGetH(), nGetK(), nGetL(), ...

...
Classes and objects of the d*TREK toolkit

- **Class Creflnlist**
  - Constructor Creflnlist()
  - ::nRead() (like Ralf's __init__)
  - ::nReduce() (needs Ccrystal object)
  - ::nSort()
  - ::nInsert(), nDelete(), nSelect()
  - ::nWrite()

**Hungarian notation**

- Used at Microsoft
- Invented by a Hungarian employee of MSFT
- Examples as used in d*TREK:
  ```c
  int nH;
  Int *pnH;
  double dH;
  int anHKL[3];
  Int a3x3dMatrix[3][3];
  Ccrystal *poCrystal;
  Cspacegroup *m_poSpacegroup;
  ```

**Future software**

- **Current problems**
  - Code maintenance in multi-platform environment
  - Lots of Windows users
  - CrystalClear – MFC-based (native Windows GUI)
  - Team of programmers know MFC
  - Lots of Linux users (X Windows is native to Linux)
  - One person knows OSF/Motif
  - Lots of Mac/OSX users
  - Installation problems
  - Users know less than before
- **Solution**
  - Java?
  - Python?
  - wxWidgets?

**Bottom line: When, why, and how?**

- There is no such thing as free software
  - At a minimum no one in this room works for free
  - "You get what you pay for." – Harry Powell August 2005
- Whenever there is a clear benefit
  - New hardware, operating systems
  - New users
  - New programmers
  - New methods
  - …
One last thing …

“Remember, software is just like paper: It’s the result of research.”
--- Wladek Minor, May 29, 2005

“Software is just like toilet paper: Users want to use the softest available, then throw it away.”
--- Jim Pflugrath, June 1, 2005

Acknowledgements

- Rigaku/MSC
  - Thad Niemeyer
  - Robert Bolotovsky
  - Cheng Yang
  - Kris Tesh
  - Tom Hendrixson
  - Joe Ferrara
- Ed Westbrook
- R. Jacobson
- US Dept of Energy
  - Contract 943072401
- Gerard Bricogne
  - EEC Workshops
- Clemens Vonrhein