Legacy Codes.

Do They Have Any Value?

David Watkin
Chemical Crystallography
OXFORD

Siena, 2005

Software Archaeology
Legacy Codes

This talk will look at:

- The evolution of legacy codes
- Their scientific value.
- Problems with maintaining them.

Legacy Code

Legacy Code is a generic term for software written at 'some time' in the past and falling into one of two broad categories:

Obsolescent.

Code currently in active use, but clearly with a limited future.

Obsolete.

Code no longer in active use.

Current Code

Although there are still substantial bodies of legacy FORTRAN code still in use in the domains of both powder crystallography and macromolecular crystallography, these two fields are also active in developing modern programs.

Small molecule crystallographers seem to worry less about using ageing software.

Obsolescent Legacy Code

In small molecule crystallography, this includes **almost all** the programs currently in use.

These codes will have limited future life spans for several reasons:

1. They were written by a single author, or small group of authors.
2. They were written in languages (usually FORTRAN) which are themselves becoming obsolete.

Obsolescent Legacy Code

Examples of obsolescent code include:

SHELX* - based almost entirely upon the effort of one author.

SIR200* - Designed and maintained by a non-dispersed group, and thus vulnerable to the fate of that group.

DIRDIF – The principal programmers are now retired.

CRYSTALS – The internal data structure is at the limit of adaptability.

Obsolete Legacy Code

Obsolete legacy codes include:

- Ancestors of current obsolescent codes (*e.g.* SHELX-76).
- Code that the author is unable or unwilling to support.
- Codes addressing problems which have gone away (*e.g.* DIFABS).

Re-inventing the Wheel

Should one try to keep old codes working, or

should one use old codes as background to new, better, codes?

Most wheels are round

But not all wheels are equal

www.bedrock.deadsquid.com

www.concordesst.com

Uses of Legacy Code

Very likely, the absolute number of crystallographers able to modify and compile programs has declined since the 1970's

The number of potential users of legacy code is probably also declining.

Uses of Legacy Code

For the few people who are able to understand and compile legacy code, potential uses are:

- Ability to perform computations not in current codes.
- Possibility of validating new codes against old codes.
- A source of information at a more detailed level than that found in published papers.

Uses of Legacy Code - Novel Computations

Ability to perform a computations not in current codes.

Example:

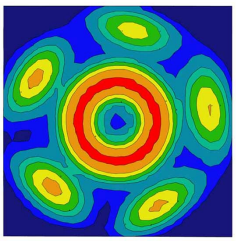
Some formulas for the X-ray scattering from atoms in various spatial probability distributions.

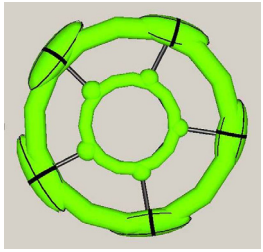
Murray Vernon King, and William N. Lipscomb

Acta Cryst. (1950). 3, 318

Uses of Legacy Code - Novel Computations

Example: The replacement of a disordered group of atoms by electron density distributed over a geometrical shape.





Uses of Legacy Code - Novel Computations

Ability to perform a computations not in current codes.

Example: The replacement of a disordered group of atoms by electron density distributed over a geometrical shape.

- King, M.V. and Lipscomb, W.N., (1950) Acta Cryst. 3, 155-158
- Bennett, M.J., Hutcheon, W. L. and Foxman B. M. (1975) Acta Cryst. A**31**, 488-494
- Chernyshev,V.V., Fetisov,G.V., Laktionov,A.V., Markov,V.T, Nesterenko,A.P. & Zhukov,S.G.(1992) J. Appl. Cryst. 25, 451 – 454
- Zlokazov,V.B. & Chernyshev,V.V. (1992) J. Appl. Cryst. 25, 447 - 451
- Chernyshev,V.V., Zhukov,S.G., Yatsenko,A.V., Aslanov,L.A. & Schenk,H. (1994)Acta Cryst. A50, 601 – 605

Uses of Legacy Code - Novel Computations

Example: The replacement of a disordered group of atoms by electron density distributed over a geometrical shape.

By 1997 the 1975 code and the 1990's codes were unavailable for use or inspection.

Non of the published descriptions of the procedure made any mention of the latent problems in refining the parameters for these shapes.

The wheel was re-invented.

Uses of Legacy Code - Validation

It is extremely difficult to prove that a program will work correctly for any valid data input.

It is difficult to demonstrate that a program will work correctly over a wide range of unusual or marginal data inputs.

A wide user-community over a long period of time tends to uncover unstable coding.

Uses of Legacy Code – Design

Careful examination of codes that have evolved over a long period may help in the effective design of new programs. Potential issues are:

1. Practical user requirements.
2. Data representations.
3. Singularities and instabilities.
4. Exceptions and error recovery.
5. Algorithmic efficiency.

Uses of Legacy Code - Design

Practical user requirements.

A programmer from a non-crystallographic background will require a very detailed specification of what the code must do.

A programmer with a crystallographic background is likely to have experience restricted to certain fields.

Old codes, or their manuals, may reveal wider requirements.

Uses of Legacy Code - Design

Wider user requirements.

In small-molecule crystallography it is now fashionable to input space group information in the form of the short or long symbols.

What if the user wants a non-standard setting – *e.g.* A -1?

What if the user wants something which cannot be represented by a conventional symbol?

Uses of Legacy Code - Design

Wider user requirements.

Why should the user wants a non-standard setting – *e.g.* A-1?

A common reason is to simplify the visualisation of related structures – *e.g.* host lattices with different guests.

Space Group operators can be generated from a look-up table, or a set of rules.

Non-Standard Settings

The layered aluminium phosphates form extended lattices able to accommodate organic guest molecules.

Pyridine complex, *P* -1:

a=6.99 b=7.22 c=12.11 α=105.1 β=104.9 γ=90.3

Imidazole complex, *C* 2/c:

a=21.9 b=7.18 c=6.99 α=90 β=104.2 γ=90

Non-Standard Settings

Using a *C* centred triclinic cell reveals the structural similarities:

Pyridine complex, *C* -1:

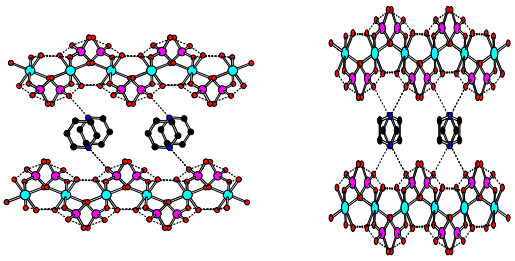
a=23.4 b=7.22 c=6.99 α=90.4 β=105.7 γ=88.1

Imidazole complex, *C* 2/c:

a=21.9 b=7.18 c=6.99 α=90 β=104.2 γ=90

Non-Standard Settings

Using a *C* centred triclinic cell reveals the structural similarities:



Uses of Legacy Code - Design

Wider user requirements.

The user wants something which cannot be represented by conventional symbols.

In this case, if the program will not accept symmetry matrices or operators, the need cannot be fulfilled.

Uses of Legacy Code - Design

Wider user requirements.

Why should the user wants something which cannot be represented by conventional symbols?

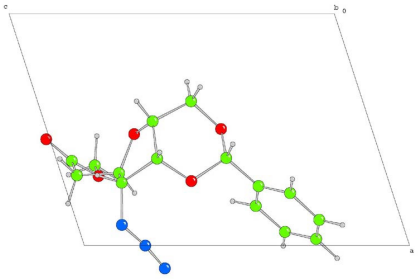
A common reason is to simplify the visualisation of related structures – *e.g.* structures before and after a phase transition.

Uses of Legacy Code - Design

Wider user requirements.

Visualisation of related structures.

At 293K this sugar azide has one molecule in the asymmetric unit

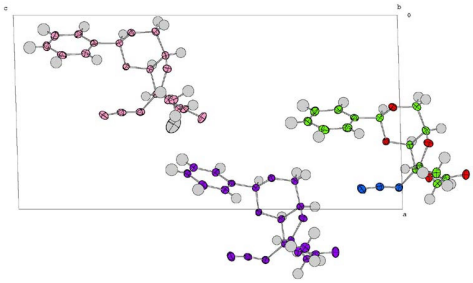


Uses of Legacy Code - Design

Wider user requirements.

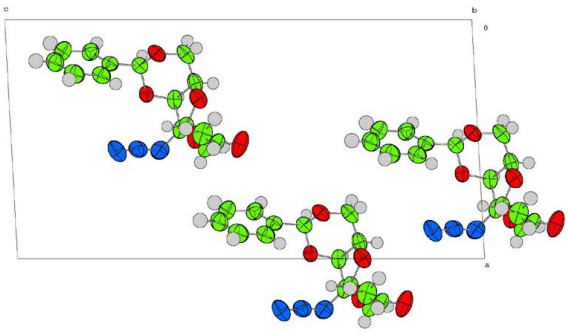
Visualisation of related structures.

At 100K the sugar azide has three molecules in the asymmetric unit



Uses of Legacy Code - Design

Tripling the unit cell for the Z'=1 cell and introducing more SG operators makes comparison of the structures simpler.



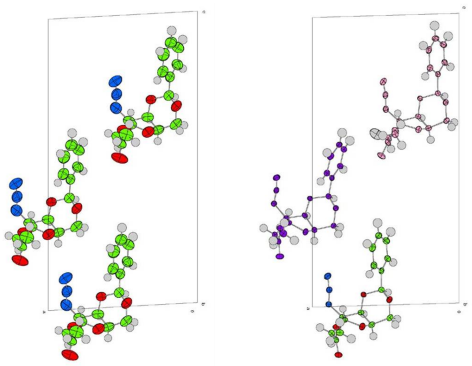
Un-named Space Group

The true SG for both the 100K and 290K cells is *P* 2_1. For the tripled cell, the operators are:

SYMMETRY x, y, z
SYMMETRY 1/3+x, y, 1/3+z
SYMMETRY 2/3+x, y, 2/3+z
SYMMETRY -x, 1/2+y, -z
SYMMETRY 2/3-x, 1/2+y, 2/3-z
SYMMETRY 1/3-x, 1/2+y, 1/3-z

Uses of Legacy Code - Design

Tripling the unit cell and introducing more SG operators makes comparison of the structures simpler.



Uses of Legacy Code
Data Representations

At the design stage, attention too closely focussed on solving a particular problem may lead to restrictive data representations, and hence limited novel applications.

However, good initial design both reduces the need for future changes, and also makes the task easier if changes become inevitable.

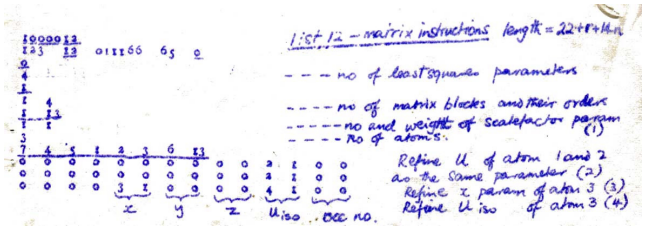
Modern languages make it easier to change data structures as a program evolves.

Uses of Legacy Code
Data Representations
Evolution of CRYSTALS

'EDITION 1'	(Plain text data base definition) (Cruickshank, Freeman, Rollet, Truter, Sime, Smith and Wells, 1964)
'NOVTAPE'	(In AUTOCODE) (Hodder, Rollet, Prout and Stonebridge, Oxford, 1964),
'FAXWF'	(In ALGOL) (Ford and Rollett, Oxford, 1967),
'CRYSTALS'	(In FORTRAN) (Carruthers and Spagna, Rome, 1970)
'CRYSTALS'	(In FORTRAN, major re-write) (Carruthers, Prout, Rollet and Spagna, Oxford, 1975)
'CRYSTALS'	Issue 2 (In FORTRAN, major re-write) (Carruthers, Prout, Rollet and Watkin, Oxford 1979)
'CRYSTALS'	Issue 7 (in FORTRAN, VAX SMG user interface) (Betteridge, Prout and Watkin Oxford, 1983)
'CRYSTALS'	Issue 11 (in FORTRAN with C++ GUI) (Watkin, Prout, Carruthers, Betteridge, Cooper, 1997)

Uses of Legacy Code
Data Representations

Evolution of CRYSTALS - Matrix of Constraint
Autocode, 1965



Uses of Legacy Code
Data Representations

Evolution of CRYSTALS - Matrix of Constraint
FORTRAN, 1975

```
#LIST 12
FULL C(3,X,U[ISO])
EQUIVALENCE C(1,U[ISO]) C(2,U[ISO])
```

Uses of Legacy Code
Data Representations

Other Solutions - SHELXL

```
FVAR osf uiso
At1 1 10+x 10+y 10+z  11 21
At2 1 10+x 10+y 10+z  11 21
At3 2 x    10+y 10+z  11 uiso
```

Uses of Legacy Code
Data Representations

Other Solutions – XTAL CRYLSQ

```
NOREF (X,Y,Z)(At1,At2)
NOREF (Y,Z)(At3)
CONSTR U(At1) = 1.0*U(At2)
```

CONSTR $p_s(At_s) = Q + m_1p_1(At_1) + m_2p_2(At_2) +$

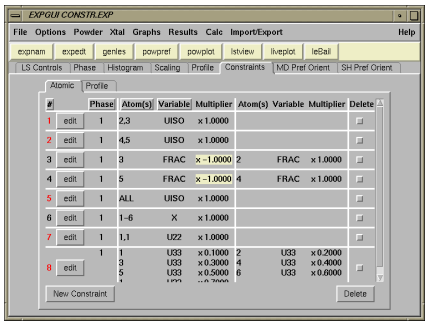
Uses of Legacy Code
Data Representations

Other Solutions – GSAS
GSAS has a coded command-line input mode

```
l a l
k          (Konstraint)
I          (Insert)
1 uiso 1 1          (phase, param, atom, mult)
1 usio 2 1
```

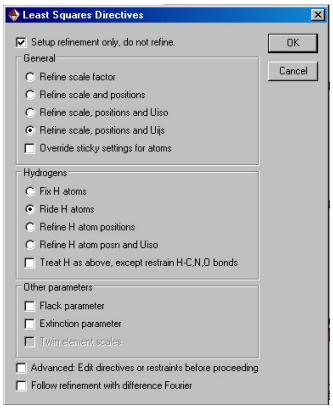
Uses of Legacy Code
Data Representations

Other Solutions - GSAS



A GUI overlay, EXPGUI, simplifies some input.

Uses of Legacy Code



Evolution of CRYSTALS Matrix of Constraint

C++, 1999

Uses of Legacy Code
Data Representations

The Z' 1 & 3 structures re-visited

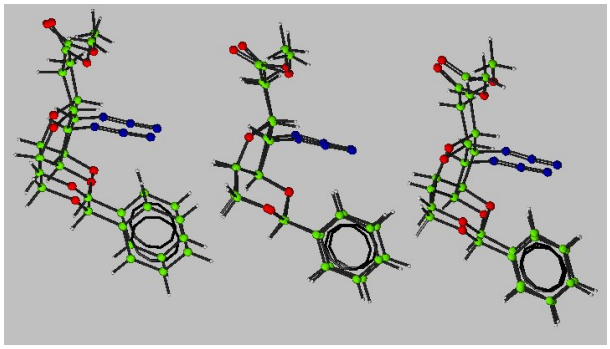
An alternative, less computationally efficient, method is to triple the contents of the asymmetric unit, and use the matrix of constraint to reduce the number of variables

```
FULL
LINK C(101,X'S)  UNTIL H(1293) AND
CONT C(201,X'S)  UNTIL H(2393) AND
CONT C(301,X'S)  UNTIL H(3493)

LINK C(101,U'S) UNTIL N(123)  AND
CONT C(201,U'S) UNTIL N(223)  AND
CONT C(301,U'S) UNTIL N(323)
END
```

Uses of Legacy Code - Design

With the 100 and 293K structures referred to a common cell and origin, the consequences of the phase change become evident



Integral & Bolt-on GUIs

Bolt-on GUIs are generally restricted to passing normal user-commands to the program, and parsing output files.

The opportunity for real interaction is restricted.

In CRYSTALS, because we both maintain the underlying FORTRAN and designed the GUI, we can give the GUI access to anything available in the FORTRAN.

Uses of Legacy Code - Design

Singularities and instabilities.

Careful analysis of the mathematics before coding begins should reveal latent singularities.

e.g. standard uncertainty in a torsion angle.

$$\frac{\partial\tau}{\partial v^s} = K\left(\frac{\partial A}{\partial v^s} - \frac{A}{2B}\frac{\partial B}{\partial v^s} - \frac{A}{2C}\frac{\partial C}{\partial v^s}\right)$$

$$\frac{\partial\tau}{\partial w^s} = K\left(\frac{\partial A}{\partial w^s} - \frac{A}{2C}\frac{\partial C}{\partial w^s}\right)$$

with $K = -1/[(BC)^{1/2}\sin\tau]$.

Acta Cryst. (1974). A30, 848

On the standard deviation of dihedral angle. By URI SHMUELI

Uses of Legacy Code - Design

Singularities and instabilities.

Instabilities and their cure may need *ad hoc* solutions.

e.g. Solution of Simultaneous Equations.

The NAG subroutine library contains 37 different routines for this purpose.

Experience with old codes may indicate which methods are most appropriate for different crystallographic tasks.

Uses of Legacy Code - Design

Singularities and instabilities.

Instabilities and their cure may need *ad hoc* solutions.

e.g. Least Squares Parameter Refinement.

Over-shifting of ill-defined parameters can be controlled by the use of:

1. Marquardt-type augmentation of the normal matrix.
2. A matrix of partial shift (damping) factors.
3. Boundary conditions on parameter values.

Uses of Legacy Code - Design

Algorithmic Efficiency.

Abstraction of a procedure into structured layers helps in the design, building and de-bugging of code.

However, over-generalisation may have serious impacts on performance.

Equally, over-optimisation can hinder future development of the code.

Choosing the Right Wheel

Sometimes the programmer must make choices at the design stage, but often a better strategy is to offer a range of alternatives to the user

www.bedrock.deadsquid.com www.concordesst.com

User Choice

Example:

Fixing the origin in polar space groups.

e.g. y in *P* 1 2_1 1

1. Do not refine the y coordinate of one atom
2. Augment the normal matrix by using Lagrange multipliers
3. Use eigenvalue filtering of the normal matrix
4. Use a matrix of constraint
5. Use supplementary equations of restraint

User Choice

Example: Fixing the origin in polar space groups, e.g. y in **P**1 2_1 1

1. Do not refine the y coordinate of one atom.

   This technique is available in any program which permits the user to decide which parameters to include in the refinement.

   The method, once popular, is now largely obsolete.

   Inversion of the normal matrix by Cholesky decomposition can automatically apply the technique if the user or program fails to do anything better.

User Choice

Example: Fixing the origin in polar space groups, e.g. y in *P* 1 2_1 1

2. Augment the normal matrix by using Lagrange multipliers.

   This is the classical method for applying constraints to least squares, but is uncommon in widely distributed crystallographic programs.

User Choice

Example: Fixing the origin in polar space groups, e.g. y in *P* 1 2_1 1

3. Use eigenvalue filtering on the normal matrix.

   Eigenvalue filtering removes singularities from the normal matrix by removing degenerate parameter combinations.

   It is an expensive way to fix floating origins, but the method may have other uses.

User Choice

Example: Fixing the origin in polar space groups, e.g. y in **P**1 2_1 1

4. Use a matrix of constraint.

   Constraints are fundamental to refinement (for example, in dealing with atoms on special positions).

   A generalised implementation gives the program user a powerful tool.

Matrix of Constraint

The physical ('real') parameters are related to a smaller set of least squares parameters together with some additional, unconditional, knowledge.

$$[physical\ \ parameters] = [knowledge].[LS\ \ parameters]$$

Matrix of constraint

Atom on special position $(x,-x,z)$ requires only 2 LS parameters

$$\begin{bmatrix} x \\ -x \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} u \\ v \end{bmatrix}$$

Matrix of constraint

Atom on special position $(x,2x,z)$ requires only 2 LS parameters.

$$\begin{bmatrix} x \\ 2x \\ z \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 2 & 0 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} u \\ v \end{bmatrix}$$

Matrix of Constraint

'Riding' a hydrogen atom on a carbon atom is done via the matrix of constraint.

$$\begin{bmatrix} C\delta x \\ C\delta y \\ C\delta z \\ H\delta x \\ H\delta y \\ H\delta z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \delta x \\ \delta y \\ \delta z \end{bmatrix}$$

Matrix of Constraint

A floating origin can be fixed via the matrix of constraint.

$$\begin{bmatrix} C\delta y_1 \\ C\delta y_2 \\ C\delta y_3 \\ C\delta y_4 \\ C\delta y_5 \\ C\delta y_6 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix} \cdot \begin{bmatrix} \delta p_1 \\ \delta p_2 \\ \delta p_3 \\ \delta p_4 \\ \delta p_5 \end{bmatrix}$$

User Choice

Example: Fixing the origin in polar space groups, e.g. y in ***P*** 1 2_1 1

5. Use supplementary Equations of Restraint

   'Restraints' have become a popular method for adding knowledge into the normal matrix.

   In this case the knowledge is that the scattering power weighted centre of gravity of the structure should not change during refinement.

$$0.0 = \sum_{i=1}^{n} w_i \delta y_i$$

User Choice

1. User-choice of parameters for refinement.
2. Augment the normal matrix by using Lagrange multipliers.
3. Eigenvalue filtering on the normal matrix.
4. Matrix of constraint.
5. Supplementary equations of restraint.

A problem-independent implementation of these techniques will address the floating origin issue, but also provide tools for solving other problems.

Funding Legacy Code

Software exists to help work out the cost of software projects

Funding New Code

The commercial cost (including overheads, pension, insurance, salary etc) of one programmer is of the order of $200,000 *p.a*.

If a program contains a decent amount of informative comment and is supported by both programming and user documentation, a rule of thumb costing over the whole project is:

$20 per executable line

The Investment in Legacy Code

The FORTRAN source of PLATON is 79,908 lines.

The commercial cost of PLATON is about $1.6 Million

The FORTRAN source of the 2005 release of CRYSTALS (excluding C++ GUI and SCRIPTS) is 155,000 lines.

The commercial cost of CRYSTALS is about $3.1 Million

The FORTRAN source of SHELXL is 17,134 lines

The cost of SHELXL is $1/3 Million??

Legacy Code Development Costs

The cost per line of extending old code may be much higher if:

1. The code is poorly documented.
2. The code has un-structured data management.
3. The code has an inflexible data structure.
4. Procedures are monolithic.
5. The code was optimised for speed rather than flexibility.

Modifying Existing Programs -Effort

The cost of extending old code must include the cost of understanding it.

Much of the standard legacy software is 'economically' commented.

PLATON-01	2%
SHELXL	10%
ORFLS	11%
CRYSTALS SFLS	24%
ORION-74	33%

Documentation

Documentation is crucial for the maintenance or recycling of old codes.

'It is expected that the Fortran listing and the glossary of symbols which are provided will serve as a complete description of the program.'
ORFLS, August, 1962

```
C       START LOOP TO STORE MATRIX AND VECTOR.
C       SEE GLOSSARY FOR STORAGE SCHEME
          DO  3010  J=1,NV
3010      DV(J)=DV(J)*SQRTW
          JK=NM
          DO 05001 J=1,NV
04301           IF(DV(J))04501,04401,04501
C             BY-PASS IF DERIVATIVE IS ZERO
04401             JK=JK-NV+J-1
                GO TO 05001
04501             DO 04801 K=J,NV
                  AM(JK)=AM(JK)+DV(J)*DV(K)
                  JK=JK-1
```

Sadly, the Le Bail Museum does not hold a copy of the Glossary.

Funding Legacy Code

Funding formulae are difficult to apply to development.

Factors influencing re-implementation costs include:

- Complexity of user interface.
  - SHELX76 – *file in – file out*. 1 person-day *p.o.s.*
  - Main Frame CRYSTALS - *with multiple I/O files and binary data base*. 1 person-week *p.o.s.*
  - CRYSTALS 2005 – *with full GUI*. Unknown person-week *p.o.s.*

p.o.s = per operating system

Supporting Legacy Code

Is it worth the cost and effort?

Sometimes.

People tend to like what they know.

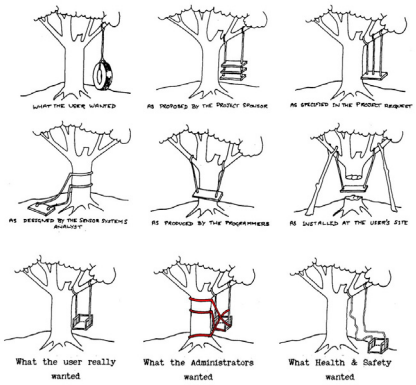
Often, people don't know what they don't know.

Learning from the Past

Minimise the cost of developing new code.

1. Evaluate consumer's needs
2. Evaluating existing products
3. Product specification
4. Detailed design
5. Coding
6. Validation
7. Maintenance
8. Development

Learning from the Past



Consumer's needs

Learning from the Past

Existing products – Background Research

Table 1.1 Precursors to CAMERON, 1990

NAME	Line	Ball and stick	Space filling	Polyhedra	Ellipsoids
SNOOPI	YES	YES	NO	NO	YES
PLUTO	YES	YES	NO	NO	NO
CHEM-X	YES	YES	YES	YES	NO
ORTEP	YES	NO	NO	YES	YES
SHAKAL	YES	YES	YES	NO	NO
DTMM	YES	YES	YES	NO	NO
STRUPLOT	YES	NO	NO	YES	NO
MOLDRAW	YES	YES	NO	NO	NO
COSMIC	YES	YES	NO	NO	NO
MOLVIEW	YES	YES	YES	NO	NO
MACMOLECULE	YES	YES	YES	YES	NO

Re-use of Good Code

```
      NB=8.0*ALOG10(0.05*FLOAT(MAX0(NREF,100))+0.5)
C MAXIMUM OF 30 POINTS ON WILSON PLOT
      IF(NB.GT.30) NB=30
      WRITE(NCWU,440) NREF,RHOMAX,NB
440   FORMAT(23H NUMBER OF REFLEXIONS =,I6,8X,
     1 32HMAXIMUM (SIN(THETA)/LAMBDA)**2 =,F7.4,8X,
     2 33HNUMBER OF POINTS ON WILSON PLOT =,I3)

C OBTAIN SUMS FOR WILSON PLOT AND FIT LEAST SQUARES STRAIGHT LINE
      CALL WILSUM(PTS,ISTATP,IL28FL)

C PLOT WILSON CURVE AND LEAST SQUARES STRAIGHT LINE
      CALL GRAPH80(0,IPLOTW)
 450 BT=2.0*BT

C CALCULATE SCALE FACTORS FOR APPROPRIATE REFLEXION GROUPS
```

NORMAL80 & SIR92

```
      nb=8.0*alog10(0.05*float(max0(nref,100))+0.5)
c     maximum of 30 points on wilson plot
      if(nb.gt.30) nb=30
      if (iprin.gt.0.and.jump.lt.0) write(lo,442) nb
442   format(34h number of points on wilson plot =,i3)
      if(jump.ge.0) go to 450

c     obtain sums for wilson plot and fit least squares straight line
      call sir_sum(pts,ier)

      if (ier.lt.0) return
c     plot wilson and debye curves and least squares straight line
      call plotw
 450 bt=2.0*bt

c     calculate scale factors for appropriate reflexion groups
```

Finding Legacy Code

Software Museum.

Le Bail.

http://sdpd.univ-lemans.fr/museum/

Crystallography Source Code Museum - FORTRAN

1960s	1970s	1980s	1990s	2000s
tracer-65	fordap-70	eiffi-80	treor-90	espoir-00
patnat-67	reduce-70	rmultan-80	dicvol-91	ortep-00
cryls-68	ortep-71	normal-80	powder-91	xtal-00
datap-68	sadiana-71	search-80	rmca-92	diffax-01
lsqpl-69	wilson-72	dicvol-82	pluto-92	
orfls-69	vanadu-73	xrs-82	sir-92#	
weight-69	agnost-74	mprof-83	sirpow-92#	
	icon-74	ito-84	stidy-92	
	linex-74	orffe-84	tmacle-92	
	camel-75	struplo-84	wyz-92	
	latcon-75	treor-84	absorb-93	
	dls-76	block-85	parst-93	
	shelx-76	eiffi-85	shabba-93	
	camel-77	qetspec-85	strump-93	
	lazy-77	geom-85	arihe-94	
	louv-77	lsq-85	dbws-94	
	xfls-77	orfep-85	cascade-95	
	celref-78	rbls-85	difabs-95	
	fordap-79	search-85	gtsym-85	
	shelx-79	sorb-85	ccsl-96	
	volcal-79	vtanal-85	lhpm-96	
		hole-86	mprof-96	
		powd12-86	ortep-96	
		shadow-86	struvir-96	
		shelxs-86	caos-97	
		visser-86	cif2sx-97	
		appleman-87	difrac-97	
		ito-87	dirdif-97	
		patsee-87	fullprof-97	
		pawley-87	glassvir-97	
		wppf-87	laue-97	
		dragon-88	tessel-97	
		mcmaq-88	xhydex-97	
		pro-fit-88	lucnral-98	
		xfit-88	lapos-98	
		dbws-99	lhpm-98	

Finding Legacy Code

SinCris.

Y Epelboin.

This resource contains the URL of over 600 crystallography software sites.

Software database for crystallography

A B C D E F G H I
J K L M N O P Q R
S T U V W X Y Z Misc

- New contribution to be submitted to the editor:
- Message to the Editor

Any question should be directed to the author or to the Editor, Yves.Epelboin@lmcp.jussieu.fr .

13.01.2004 - SinCris Editor - Copyright © International Union of Crystallography

Finding Legacy Code

SinCris.

Some of the sites pointed to have closed down

ABSCYL	ABSEN	ABSORB	ACN
ADM	ALIGN	ALTWYCK	AMI
ANSIG	ANTHEPROT	ARITVE	ARP
ATOMS	AUTO_XPL	AXES	AZA
Acquisition of Images	Alscript	Altwyk	Ama
AmiraMol	Amps	Archive for MacOS	Arg
Assp	AutoDep 2.0	AutoDock	BAB
BALSAC	BEAM-ish	BGMN	BIO
BLANC	BLAST	BOB	BRA
BRASS	BRL	BREADTH	BUN
BUSTER	bca-spreadsheets	BenchFFT	Bett
Biological software EBI	Bond Valence Wizard	Biological software JHU	Biop
CACTVS	CALCRYS	CAMEL JOCKEY	CAC
CCL	CCP14	CCP14 Solutions	CCI
CCSL	CCTBX	CELLSIZE	CEL
CGI programing	CHARMM	CHIME	CHC
CIF	CIF2	CIFLIB	CLU
CMPR	CNS	CNSsolve	CNS
COMPANG	COMPDIS	CONSCRIPT	COS
CORINA	CRISP	CRUSH	CRY
CRYSCOMP-CRYSDRAW	CRYSCON	CRYSFIRE	CRY
CRYSTALVIEW	CSD	CSD2RES	CSI
CUFOUR	CVIS	Ca.R.ine Crystallography	Cel
Cerius2	Chekcell	Chem-Ray	Che

Legacy Code - Conclusions

If it is well-liked and much-used, someone will maintain it.

Command-line I/O is most easily maintained.

If it has complicated I/O or a proprietary GUI, it will probably die.

Good code can supplement published work.

Code without manuals is almost valueless.

Code with good commenting is valuable.

Don't re-invent the wheel. Build better ones.