CCP4i: Recent Developments, Future Prospects

Peter Briggs, CCP4

Introduction and outline

- New features in CCP4i 1.4.4
  - For users
  - For software developers
- Ongoing and short-term developments
  - CCP4i BIOXHIT database
  - Other developments
- The longer view of the future of CCP4i

Developments for users: new interfaces in 1.4.4

`Crank:`

- Suite of programs for automated structure solution based on the CCP4i infrastructure
- Developed by Steven Ness (formerly at Leiden University)
- Current version supports SAD, SIR and SIRAS
- Covers steps from scaled/merged data to density modification
- “Translucent box” – also intended as teaching aid

`Shelx_cde:`

- Interface to Sheldrick’s SHELX C/D/E programs
- Allows programs to be run as a pipeline or individually
- Performs file format conversions and extracts useful graphs from output
- Some issues with defaults, uninterfaced options and mode of operation

Developments for users: new interfaces in 1.4.4

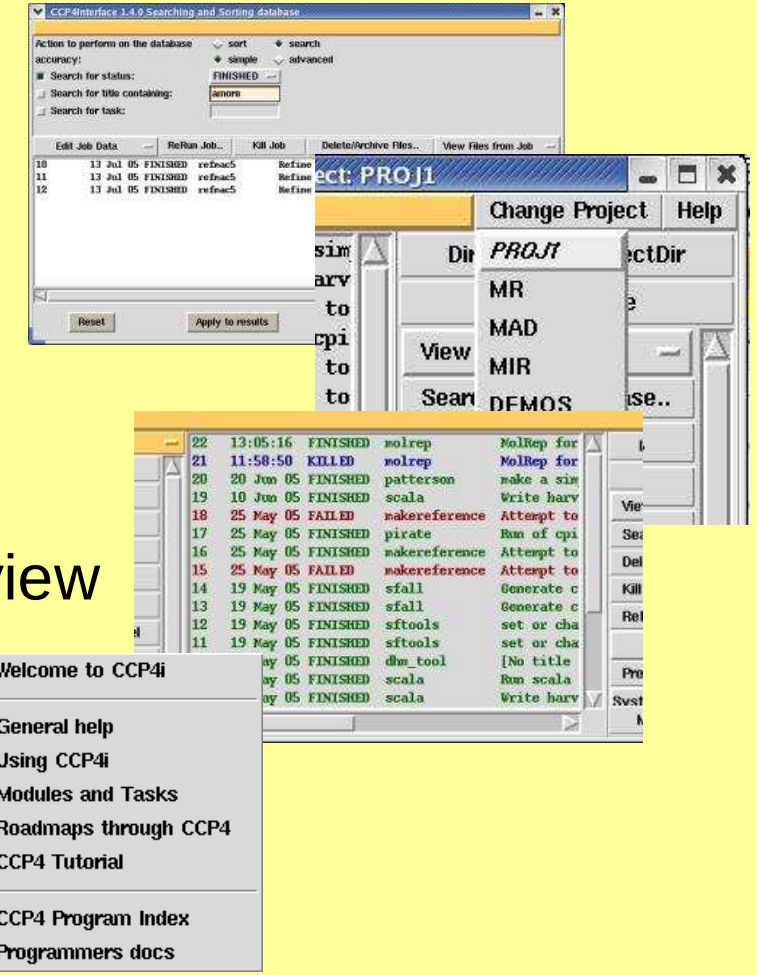
Interfaces for new programs:

- **`Phaser`**
- **`Pirate`** and **`Clipper utilities`**
- **`BP3`**
- **`Chainsaw`** *(“Create Search Model”)*
- **`CCP4mg`** launcher

Plus updates to **`areaimol`**, **`matthews_coef`** and **`pdbset/pdbcur …`**

Developments for users: major new core functionality

- Database search and sort
- Project shortcuts
- Customise job database view
- Help shortcuts



Developments for users: minor new core functionality

- Disable task buttons for tasks where underlying programs are missing ("greyed out tasks")
- Remote jobs can use ssh as well as (deprecated) rsh
- Harvest files appear explicitly as added as an output file
- Enable navigation within long logfiles split into "frames" (next/previous)
- Windows drives can also be browsed (e.g. C: or D:)
- Options to make O and/or QUANTA maps only appear if necessary programs are present
- Mapslicer retains user settings for many parameters between runs
- Notebook accessible from View Files from Job Menu

Developments for software developers

`FindExecutable:`

- returns full path of the specified program executable

`CreateLabinLine4:`

- updated version of CreateLabinLine that allows groups of four related MTZ labels to be displayed e.g. H-L coefficients or F(+)/sigF(+)/F(-)/sigF(-)

`GroupMtzCols, MtzColSameDataset, GetMtzColType, GetMtzGroupByType`:

- commands for handling MTZ column “groups” internally

`Export Tasks:`

- interface has been improved based on feedback from developers

Ongoing and short-term developments

- CCP4i/BIOXHIT database and related developments
- Development of interactive MTZ viewer and editor
- Extensions to MapSlicer
- Fixing various known issues with CCP4i, for example:
  - Improvements to the Data Harvesting Management
  - Better interface to the notebook facility
  - Handling acquisition of MTZ crystal-cells in some task interfaces
  - Known issues with Windows (e.g. spaces in pathnames)
  - Automatic configuration of helper applications e.g. web browser
  - …
- Task consolidation

CCP4i BIOXHIT Database Work

- Background and aims
- Architecture, Components & Technologies
- What it means for users
- Current status and plans
- Longer term plans

CCP4i BIOXHIT Database: Background

- Originally: a minor project to increase accessibility & functionality of the CCP4i job database

- As part of BIOXHIT: expanded remit to include extended tracking (i.e. relationships between jobs) and storage of crystallographic data
    - does not commit to providing a general data model for structure solution

- Useful for:
    - users of CCP4i
    - automated structure determination pipelines
    - non-CCP4i applications that want to use the database

- Resources:
    - 1 full-time developer (Wendy Yang) for 4 years
    - 1 part-time (Peter Briggs)

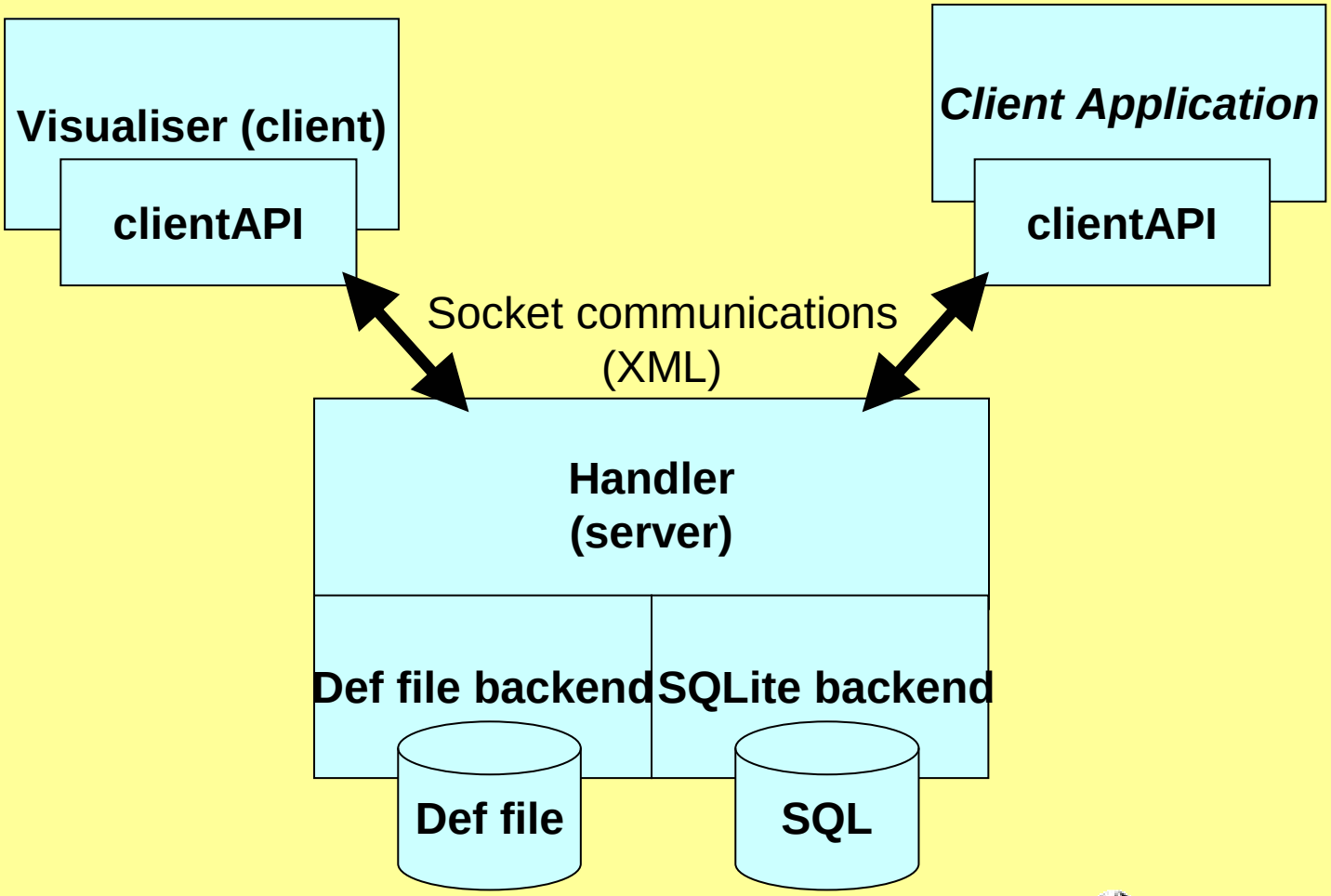
CCP4i BIOXHIT Database: Aims

- Separate the project database handling from main CCP4i process
  - Server process “Project Database Handler”
  - Allow access to db from non-CCP4(i) applications

- Expand scope of CCP4i database to store richer dataset
  - Tracking database stores project history
  - “Knowledge base” of crystallographic data for each project
  - Useful for automated pipelines
  - Aim to provide database schema and multiple backend implementations
  - CCP4i def files, SQL, …

- Develop visualisation tools to facilitate review and analysis of the data

CCP4i BIOXHIT Database: Architecture



CCP4i BIOXHIT Database: Components & Technologies

Database Handler

- implemented in Python
- communicates with client applications via a clientAPI library

clientAPI library

- gives clients access to database functionality
- hides details of starting/stopping handler & of socket/XML communications
- implement Python and Tcl versions (others later)

XML communications

- allows requests and responses to be (programming) language neutral

Database backends

- Def file backend implemented in Python, replicates current CCP4i
- SQL backend implemented in SQLite (embedded backend)

CCP4i BIOXHIT Database: Database schema

Currently developing an SQL schema for “rich database” backend

- Consists of “knowledge base” (i.e. crystallographic data) + tracking (i.e. project history)
- Being developed in conjunction with HAPPy and XIA projects

Connections with other databases

- ISPyB (DNA) and PIMS: discussed at meeting in DL (October 2005)
- Deposition: looking at SQL tables from EBI
- need to revisit once our version 1 schema is finalised

Connections with other applications

- CCP4mg/Coot/CRANK: discussed at meeting in York (October 2005)
- Brief discussion of possible implementation at DIAMOND (Nov 2005)
- need to extend to other systems e.g. AutoRickshaw at EMBL-Hamburg

CCP4i BIOXHIT Database: What it means for users

Short term: separation of db handler from CCP4i means

- non-CCP4i applications (e.g. Coot, CCP4mg) can also leave records in the database
- multiple instances of CCP4i can use the same database simultaneously

Medium term: improved functionality for handling projects

- options to import/export/synchronise projects
- options to split/merge projects

Longer term:

- improved visualisers help to review projects (e.g. flowchart-like views in addition to current list based view)
- CCP4i tasks could be partially or completely populated using data from the project knowledge base

CCP4i BIOXHIT Database: Current Status and Plans

- Developmental “handler-compatible” version of CCP4i available
  - works with “scaffold” Tcl handler
  - Tcl client API developed from this work
  - Still some issues to resolve

- Python handler with def file and SQLite backends now being developed
  - next step: replace scaffold handler with Python version in CCP4i
  - will resolve many technical issues (CCP4i is a demanding client)
  - finalise Tcl client API for def file backend

- Initial “soft” API for knowledge base SQL schema also under development
  - next step: make this available to HAPPy and XIA
  - several iterations will be required to stabilise schema and API

CCP4i BIOXHIT Database: Longer term plans

Integrate Python handler into CCP4i for release 6.1

Provide Tcl client API to CCP4mg to allow interaction with the database

Expand def file backend to accommodate requests from CRANK

Develop tool to migrate from def files to SQL backend

Begin developing visualisation clients

- already some work based on graphviz

The longer view of the future of CCP4i

- Some strengths and successes of CCP4i
- Some limitations and weaknesses
- Some considerations for moving forward
- Proposals for modifications to address these
- How do we get there?

Some strengths and successes of CCP4i

CCP4i has been – and continues to be - very successful:

- Customised interfaces tailored to user's expectations
  - "natural language" interfaces
  - unifies interactions by giving a common front end to diverse programs
  - simplified usage of MTZ files
- Extensive integrated help system
  - bridging the hardcore program documents and more general crystallographic theory
- Task-rather than program centric
  - automates away many tedious peripheral jobs (e.g. file conversions)
- Relatively lightweight; no user lock-in
- Project tracking system "for free"
- Makes it easy to interact with files ("View file from job"/"View any file")

Some limitations and weaknesses

- Customised interfaces can be intensive to write and maintain (particularly for major programs

- Poor architecture for extension e.g.
  - lack of separation of graphical parts from scripts
  - interface toolkit is closely tied to Tcl/Tk (difficult to impossible to incorporate interfaces in other languages)

- Batch mode model works poorly for interactive/semi-interactive tasks
  - particularly true currently for file manipulations e.g. MTZ editing
  - also likely to be an issue for interfaces to automated tasks

- Supporting applications can be quite limited

- No support for plugging in CCP4mg or Coot as helper applications

- Interactions with task output is generally weak
  - Only “bare bones” presentation of most logfile output from tasks

Some considerations for moving forward

- Migration is preferable to full rewriting
  - Want to reuse what we already have whilst allowing development of new components
  - Expand the possible range of programming languages

- Likely to require large commitment of staff over a long time period
  - Modular/incremental approach offers best chance of success

Proposal #1: CCP4i resource manager

- **Split CCP4i into non-graphical server part (“resource manager”) and graphical components (main window, task interfaces)**
  - Open architecture: applications (interfaces, scripts etc) communicate with server via sockets using language neutral protocol
  - Would allow interfaces and applications to be written in any language
  - Easy to reuse existing interfaces and scripts
  - New “replacement” components (e.g. a new main window) could be developed alongside existing ones.
  - Existing non-CCP4i interfaces could be slotted more naturally into the CCP4i system
- Build on the technologies currently being used in the database work
- Would pave the way for development of new GUI toolkits for automation work

Proposal #2: integrating external applications

- **Provide a "plug-in definition" mechanism**
    - This would allow external programs to be easily associated with arbitrary files, or to be launched for specific sets of files output from a particular task
    - e.g. "View files from job" option would include a button to "View refinement results in Coot"

- Looking further ahead: provide API libraries for external applications like CCP4mg or Coot to "control" aspects of CCP4i's behaviour

Proposal #3: smart logfile browser

- **Provide a logfile/task output browser that presents the information to the user in a "smart" way**
  - smart browser would "know" about the output of specific programs
  - filter content to present only the most relevant details, and then allow "drill down" to more details if desired
  - show graphs and tables in-line with text
  - link to output files and allow launching of appropriate viewers
  - offer analysis of key results with links to appropriate documentation or other help resources
- Possible routes include
  - building on XML output from programs and tasks in automation
  - development of a library for parsing logfiles and tools for "rerendering" content such as tables and graphs

Some other possibilities

- More sophisticated task browser
  - e.g. flowchart view
  - data-driven browser suggests next task based on DB content
  - graphical programming interface…!
- GUI toolkit for semi-interactive tasks
  - suitable for interfacing to automation
- Run main window on local machine and server process remotely
- …

How do we get there?

There is currently no roadmap

- Biggest issue is lack of manpower at DL
- Projects need prioritisation and commitment of effort over long timescale
- Without some commitment there is a danger that CCP4i will stagnate
- Question is: how important is this to CCP4?

Comments from the notes of the June 2005 Automation meeting:

- CCP4i “is a key element of automation”
- “… it is a crucial CCP4 flagship feature.”
- “This is the public face of CCP4”

- Working Party to be set up to investigate (after STAB)
  - Need to decide on membership
  - Should consider current CCP4i, graphics, automation …

Acknowledgements

- Liz Potterton (original developer of CCP4i)

- Steven Ness (Crank plus useful suggestions)
- Kevin Cowtan (Pirate and Clipper utilities)
- Anne Baker and Airlie McCoy (Phaser interface)
- Francois Remacle (Database search, project switching, custom colours)

- Wendy Yang (CCP4i Database work, funded by the EU FP6 “BIOXHIT” Integrated Project)

- Daresbury staff: Charles Ballard, Dan Rolfe, Norman Stein, Martyn Winn

- All users and developers who have contributed ideas, bug reports, fixes and new code