Building on CDAT: Application development

Why build on top of CDAT?

- It has a standard and universal scripting language interface (our old friend *Python*).
- It provides access to a range of useful packages such for data plotting, I/O, manipulation, processing etc,.
- Python binds easily to other languages and software packages.
- A critical mass of scientists working on CDAT packages will result in a suite of fantastic freely available routines that can be distributed in future releases.

What would you build on top of CDAT?

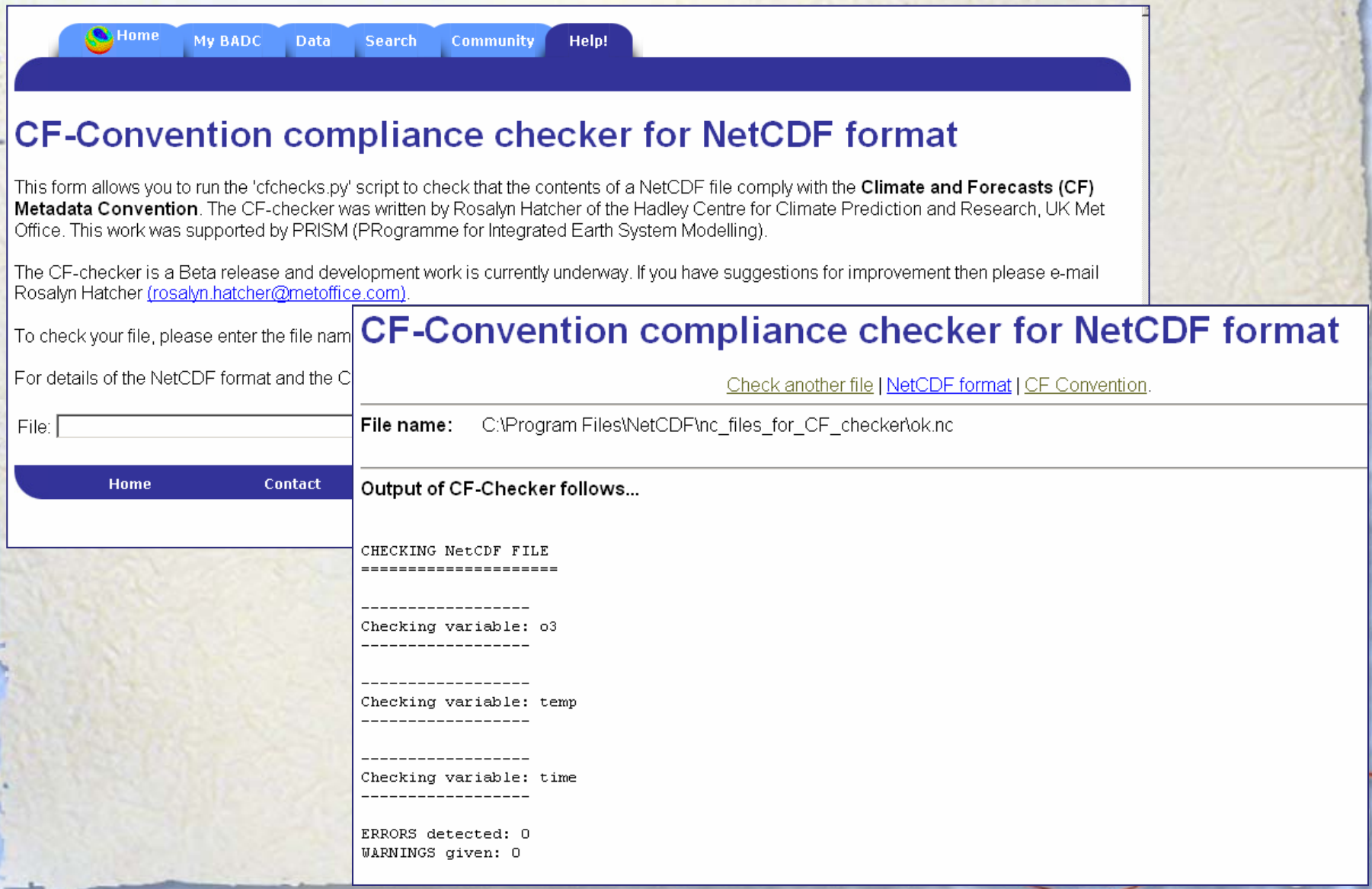
- Some examples are:
  - The CF-checking utility (Hadley Centre/BADC)
  - BADC ERA-40 delivery and caching system & LAS
  - The BADC's Data Extractor web-interface
  - IaGraph
  - and VCDAT of course!

The CF-Checker (1)

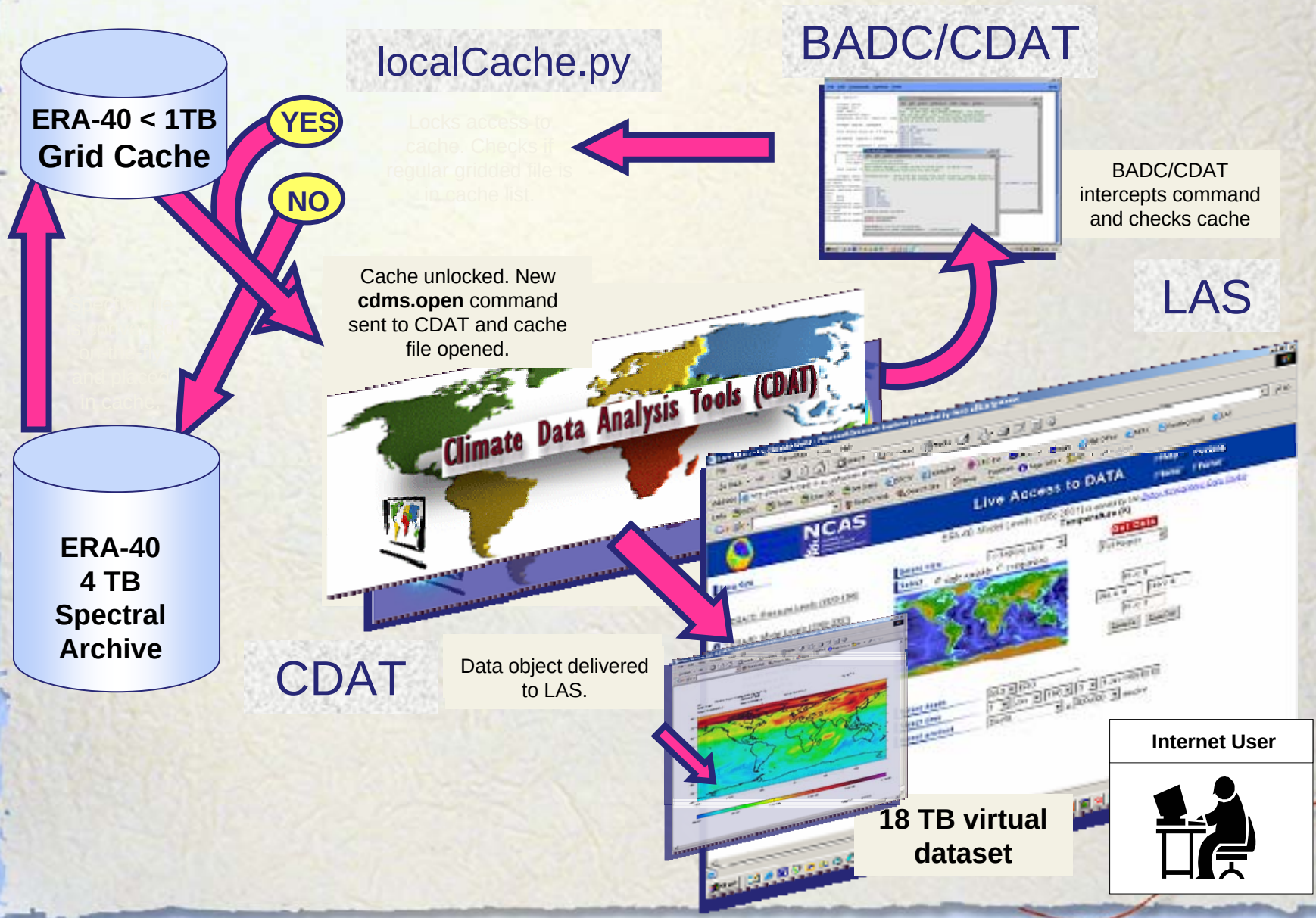
- Code developed in Python calling CDAT for NetCDF interface.
- BADC sub-classed the main version and bound it to a web application.
- Allowing users to upload a file to be checked for CF-compliance.
- Available at:

  http://titania.badc.rl.ac.uk/cgi-bin/cf-checker.pl

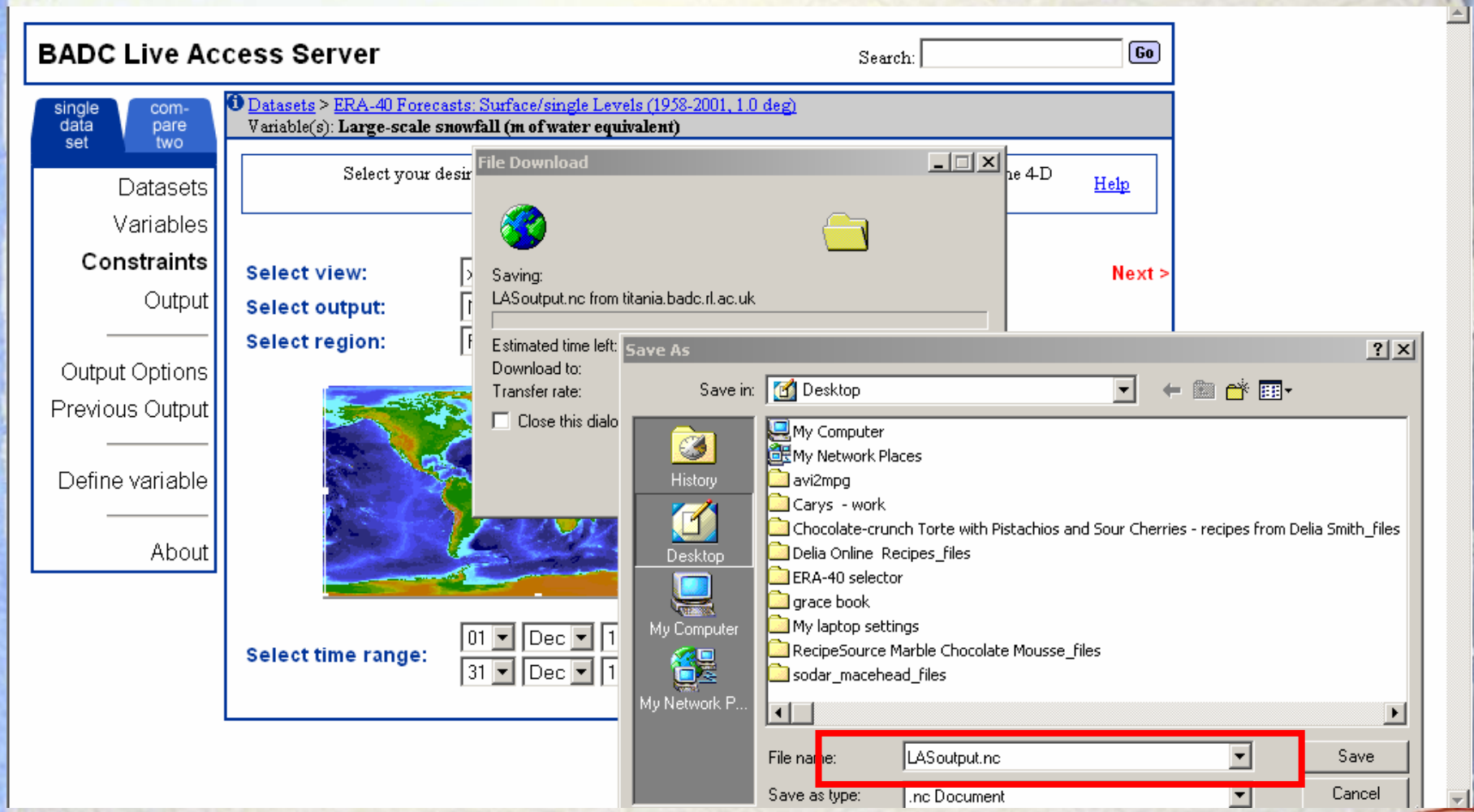
The CF-Checker (2)



BADC ERA-40 delivery system



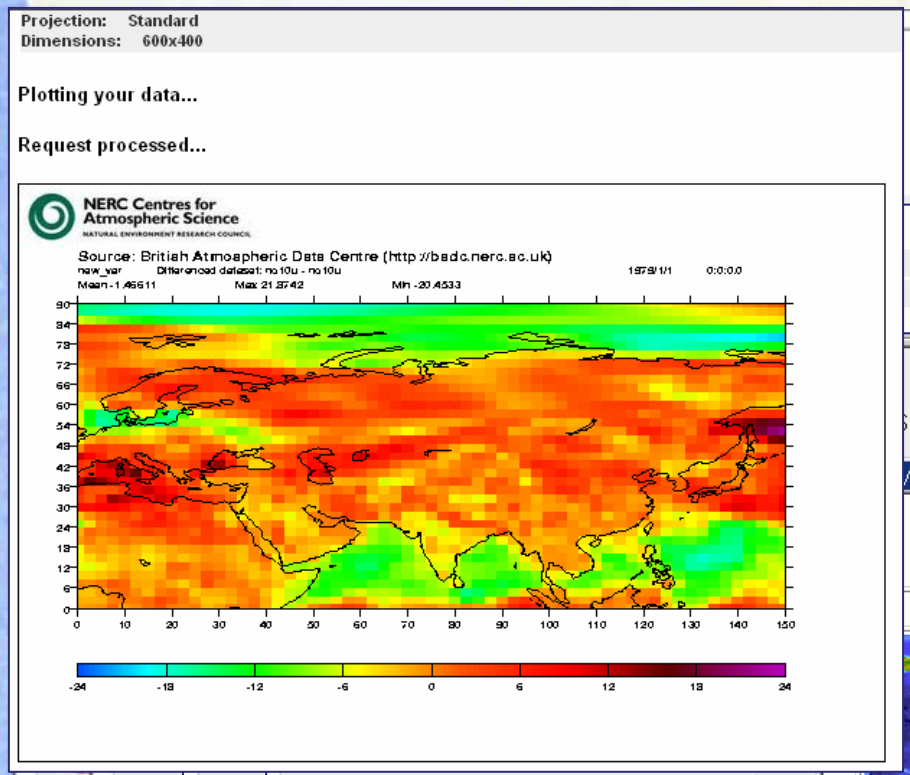
BADC's LAS Demo: 1 month to NetCDF



The BADC's Data Extractor web-interface (1)

- We found it hard to make progress with the LAS, due to:
  - dependence on many other software products (such as MySQL and Tomcat)
  - complex multi-language code – *very hard to locate where to modify code for minor changes.*
- So, we built our own…using **all-Python** of course…

The BADC's Data Extractor web-interface (2)



IaGraph - A Python Package for Quick Interactive Graphing