Modular Support for Proliferating File Formats

Daniel Y. Kimberg
University of Pennsylvania

“The wonderful thing about standards...

...is that there are so many to choose from”

This state of affairs is not unique to imaging, and is not going to be solved by changing human nature:

- you can't force everyone to adopt your file format
- you can't (and shouldn't) prevent people from developing new file formats
- you can't force people to interpret existing standards (e.g., DICOM) consistently

Modular file format support

- Boil support for each file format down to the smallest possible irreducible piece – a chunk of code conforming to some standard API. Then write enough glue code to ensure that merely having the right chunk of code will lend support for that format to all of your conforming software.

Benefits of modular suport (1)

- improves code re-use
  - just a bit of glue for each language or environ
- makes tools more interoperable
  - simple policies, like producing output in same format as input, go a long way
- makes updates easier to distribute/get/install
  - file format updates are common
- facilitates local customization
  - proprietary formats
  - local customization of common formats
  - local formats

Benefits of modular support (2)

- genericizes code development
  - readFile() and writeFile()
- makes the use and development of new formats practical, since support can be added instantly to a huge toolset, including converters
- a few new problems
  - platform issues (nspr?)
  - distribution
  - versioning

Implementation details

- libvbio – c++ library that connects plug-ins to a generic i/o API
- plug-ins may also be compiled in
- needs a low-level mechanism that sidesteps this by loading image data into buffers and provides symbolic access to metadata
- for other languages, we write a language-specific library that either loads the plug-ins directly or that communicates with an external program via... FIFOs? sockets?

If this is such a great idea, why doesn't our software do it?

- it did for a while, but we've disabled plug-ins in recent releases.
- lessons learned:
  - don't do anything on the (virtual) build box that will break binary compatibility
  - proper versioning of releases is very important
  - keep the build process simple
  - consider a layer like NSPR to make cross-platform support easier
- we hope to bring it back soon