be fun to work on, better documented, less arcane, and more modern. It was hoped that this would attract new contributors and allow creative work in the PIM space again by limiting the unpleasantness and complexity involved to those who wished to make use of the infrastructure form. The client/server approach seemed to facilitate this.

It should be noted that this fundamental decision to rework the whole data storage infrastructure for KDEPIM in a very disruptive way essentially entailed completely rewriting large parts of the system. We quite consciously accepted the fact that this would mean a lot less resources would be available to focus on maintaining the current codebase, keeping it stable and working, and also making it available as part of the KDE 4.0 release somehow, since that would almost certainly happen before such a major refactoring could possibly be finished. As it would turn out, this meant that KDE 4.0 was in fact released without KDEPIM, a somewhat harsh but probably necessary sacrifice in retrospect.

Once the group had agreed on the general direction, they produced the following mission statement:

> We intend to design an extensible cross-desktop storage service for PIM data and meta data providing concurrent read, write, and query access. It will provide unique desktop-wide object identification and retrieval.

The Akonadi Architecture

Some key aspects that would remain in the later iterations of the architecture were already present in the first draft of the design produced in the meeting. Chief among those was the decision to not use DBUS, the obvious choice for the IPC mechanism in Akonadi, for the transport of the actual payload data. Instead, a separate transport channel and protocol would be used to handle bulk transfers, namely IMAP. Some reasons for this were that it could control traffic out of band with respect to the data, thus allowing lengthy data transfers to be canceled, for example, since the data pipe would never block the control pipe. Data transfers would have a lot less overhead with IMAP compared to pushing them through an IPC mechanism, since the protocol is designed for fast, streaming delivery of large amounts of data. This was a reaction to concerns about the performance characteristics of DBUS in particular, which explicitly mentioned in its documentation that it was not designed for such usage patterns. It would allow existing IMAP library code to be reused, saving effort when implementing the protocol, both for the KDEPIM team itself and for any future third-party adopters wishing to integrate with Akonadi. It would retain the ability to access the contents of the mail store with generic, non-Akonadi-specific tools, such as the command-line email applications, pine or mutt. This would counter the subjective fear users have of entrusting their data to a system that would lock them in by preventing access to their data by other means. Since IMAP only knows about email, the protocol would need to be extended to support other mime types, but that seemed doable while retaining basic protocol compatibility. An alternative protocol option that was

Download at WoweBook.Com