not directly paid for their KDE work, there are many who work as C++ and Qt developers full-time. There are still volunteers, especially among the newer contributors, but the core group consists of professionals.[‖]

Because of the universal importance of PIM infrastructure for most computer users, be it in a personal or business context, the guiding principle of the technical decision-making process in KDEPIM has become practicability. If an idea cannot be made to work reliably and in a reasonable timeframe, it is dropped. Changes are always scrutinized for their potential impact on the core functionality, which must be maintained in a working state at all times. There is very little room for experimentation or risky decisions. In this, the project is very similar to most commercial and proprietary product teams and somewhat dissimilar from other parts of KDE and Free Software in general. As noted in the introduction, this is not always purely positive, as it has the potential to stifle innovation and creativity.

The Evolution of Akonadi

The KDEPIM community meets quite regularly in person, at conferences and other larger developer gatherings, as well as for small hacking sprints where groups of 5 to 10 developers get together to focus on a particular issue for a few days of intensive discussion and programming. These meetings provide an excellent opportunity to discuss any major issues that are on people's minds in person, to make big decisions, and to agree on roadmaps and priorities. It is during such meetings that the architecture of a big concerted effort such as Akonadi first emerges and later solidifies. The remainder of this section will trace some of the important decision points of this project, starting from the meeting that brought the fundamental ideas forward for the first time.

When the group met for its traditional winter meeting in January 2005, parts of the underlying infrastructure of KDEPIM were already showing signs of strain. The abstraction used to support multiple backend implementations for contact and calendaring information, KResources, and the storage layer of the email application KMail were built around a few basic assumptions that were starting to no longer hold. Specifically, it was assumed that:

- There would be only a very limited number of applications interested in loading the address book or the calendar: namely the primary applications serving that purpose, KAddressbook and KOrganizer. Similarly, there was the assumption that only KMail would need to access the email message store. Consequently there was no, or very limited, support for change notification or concurrent access, and thus no proper locking.

[‖] This might be surprising, since it goes against the intuitive idea of Free Software being produced largely by students with too much time on their hands, but lately there have been substantiated claims that by now the majority of successful Free Software is in fact written and maintained by professional software engineers. See for example Karim Lakhani, Bob Wolf, Jeff Bates, and Chris DiBona's Hacker Survey v0.73 at *http://freesoftware.mit.edu/papers/lakhaniwolf.pdf* (24.6.2002, Boston Consulting Group).