```
fetch->fetchScope().fetchPayloadPart( MessagePart::Envelope );
connect( fetch, SIGNAL(result(KJob*)), SLOT(fetchDone(KJob*)) );

...

typedef boost::shared_ptr<KMime::Message> MessagePtr;

void MyMessageModel::itemAdded(const Akonadi::Item & item)
{
    if ( !item.hasPayload<MessagePtr>() )
       return;
    MessagePtr msg = item.payload<MessagePtr>();
    doSomethingWith( msg->subject() );
    ...
}
```

The First Release and Beyond

When the group congregated in reliably cold and rainy Osnabrueck once more, in January 2008, the first application uses of Akonadi could be presented by their developers, who had been invited to attend. The authors of Mailody, a competitor to the default email application in KDE, had decided some time before that Akonadi could help them build a better application, and they had become the first to try out its facilities and APIs. Their feedback proved very valuable in finding out what was still too complicated, where additional detail was needed, and where concepts were not yet well documented or not well implemented. Another early adopter of Akonadi present at the meeting was Kevin Krammer, who had taken up the interesting task of trying to allow users of the legacy libraries for PIM data in KDE to access Akonadi (as well as the other way around, to access the data stored with the old infrastructure through Akonadi) by providing compatibility agents and resources for both frameworks. The issues he encountered while doing that exposed some holes in the API and validated that at least all of the existing functionality would be possible with the new tools.

A notable outcome of this meeting was the decision to drop backward compatibility with IMAP in the protocol. It had evolved so far away from the original email-only standard that maintaining the ability of the Akonadi server to act as a standard conforming IMAP server for email access was a burden that outweighed the benefits of that feature. The IMAP protocol had served as a great starting point, and many of its concepts remain in the Akonadi access protocol, but it can no longer justifiably be called IMAP. It is possible that this mechanism will return in later versions of the server, probably implemented as a compatibility proxy server.

With the KDE 4.1 release rapidly approaching, the team met again in March 2008 to do a full review of the API before its first public release, which would commit them to keep it stable and binary compatible for the foreseeable future. Over the course of two days, an amazing number of small and large inconsistencies, missing pieces of documentation, implementational quirks, and unhappy namings were identified, and they were rectified in the weeks that followed.