

# Kolab - A OSS Groupware Solution

7. Juni 2004

## Rechtlicher Hinweis

Dieser Beitrag ist lizenziert unter der GNU Free Documentation License.

## Rechtlicher Hinweis

Dieser Beitrag ist lizenziert unter der GNU General Public License.

## Zusammenfassung

Kolab is a comprehensive Groupware solution for Linux/UNIX, based on proven applications like Apache, ProFTPD, Postfix, OpenLDAP (on the server side) and KOrganizer and KMail and others on the client side. With Kolab, work groups can schedule meetings, check each other's availability, assign tasks, and of course handle email. The communication is using open and well-defined protocols and formats, such as IMAP, SSL/TLS, iCal, etc. By means of a plugin, it is even possible to use Outlook as a Kolab client.

The first phase of Kolab development in 2003, commissioned by the German Federal Agency for IT Security (BSI), led to a base groupware system, that OSS developers are now building upon to create an even better solution that should be usable in most Groupware settings. Among our goals are a "secretary mode" (scheduling appointments for your boss, by extension accessing anybody's calendar to which you have access rights), the possibility of keeping multiple calendars in the same client, comprehensive rights management, and many other goodies.

In my talk, I will present the technical and organisational challenges of Kolab development, present our goals, and even show an alpha version (or maybe beta, if we have come that far by then) of the new client and server.

## 1 Kolab - An OSS Groupware Solution

### 1.1 What is Groupware - an Attempt at a Definition

If you look at the plethora of applications that call themselves Groupware, anything that lets computer users communicate with each other would qualify as a Groupware solution, including a mere Email or IRC client. Definitions like „Software that enables a group of users to collaborate on a project by means of network communications.“ ( <http://www.liebert.com> <<http://www.liebert.com/>> ) would indicate that even a tool like CVS is a Groupware tool. Other definitions are even broader, such as the one found on <http://pip.med.umich.edu> <<http://pip.med.umich.edu/>> : „Any technology that allows people to collaborate electronically, including email, real-time networking and conference tools based on telephony, video or the Web. Workflow automation, enterprise resource planning and even telemedicine systems are all groupware at the root.“

For this paper, however, we will assume a more specific definition of the term Groupware, and say that Groupware is a software solution that enables a group of computer users to plan their work, schedule appointments and meetings, assign tasks, and keep track of each other. The best-known solution in this area is probably the MS-Outlook/MS-Exchange combination, followed by Lotus Notes (which goes a lot further by being a general replicated database with applications for scheduling, address books etc. on top of it.

## 1.2 The Linux Situation

Traditionally, there have not been many Groupware solutions on Linux, neither open-source nor proprietary. Recently, and probably partly triggered by the existence of the Kolab project, a few have sprung into existence. On the proprietary side, there are packages like Novell/Suse's GroupWise (born as WordPerfect Office), on the OSS side, there are, besides Kolab, web-based solutions like PHPGroupware, MoreGroupware, and Evolution (the latter having proprietary components that are required in order to make it work fully, like a plugin that lets Evolution users work with an Exchange server). None of these would completely fulfill all the requirements that the initial sponsors of the Kolab project had, which is why the Kolab project was started.

## 1.3 Requirements

In this section, we will list the requirements on a Groupware solution that have guided the Kolab project.

### 1.3.1 Integrated User Interface

All components like email, scheduling, tasks, and address book should be contained in one user interface. This is more than integrating everything in one top-level window, it also includes unifying menus, dialogs, work flows, etc.

### 1.3.2 Email Client

Of course, a large part of collaborating consists in communicating, and despite all instant messaging, a powerful email client is clearly a requirement. Given that a large software package like Groupware will often be used in a corporate setting and by nomadic users, IMAP support is mandatory; for Kolab, disconnected IMAP (IMAP that does not require a permanent server connection) was added as an additional requirement.

### 1.3.3 Support for Sphinx Encryption

Sphinx is a standard for email encryption and signatures developed by the German Federal IT Security Agency (BSI) for the German government. It is a mandatory requirement that clients in a Kolab system can send and receive encrypted and signed email messages according to this standard. An OSS implementation of this standard was jointly developed by Klarälvdalens Datakonsult AB, Intevation GmbH and g10code GmbH in the *Ägypten* project, using GNU Privacy Guard as its cryptography engine.

### 1.3.4 Multi-Account Capability

Kolab clients must be able to support multiple accounts per user, each of which possibly containing email messages, calendaring data, contacts, tasks, etc. Account owners can assign different access rights to other users, and accounts may even be „functional“, i.e., they belong to a group of people working together (e.g., the members of a project). Assignment of access rights must be possible both at account level and at folder level.

### 1.3.5 Multi-Address Capability

A Kolab user can have multiple email addresses which are all handled by the same Kolab installation. One email address is the designated primary address, the one which is always used as the sender address.

### 1.3.6 Multi-Location Capability (Load Distribution)

Configuration data, user databases, etc. are replicated to external servers. Accounts can be distributed over several servers for the purpose of load distribution or for the case where users are located in offices that are only connected via slow lines.

### 1.3.7 Group Management

Kolab clients can define groups (sometimes also called distribution lists). A group can both be the recipient of e.g. an email message or a task, and the holder of certain access rights.

### **1.3.8 LDAP Data Storage**

Given the distributed nature of most Kolab installations, a central networked data storage for things like public address books, user settings, etc. is necessary. It is a requirement for a Kolab system to store its information in a LDAP server according to a published LDIF scheme.

### **1.3.9 Shared Folders**

Shared folders can be created and accessed by every Kolab user with sufficient access rights. No separate logon is necessary for accessing such a shared folder.

### **1.3.10 Appointments and Meetings**

Users can define appointments for themselves; these are entered into their calendar.

Users can schedule Meetings by creating an appointment and inviting people to it (also see below, Free/Busy Lists). Invitation messages are sent to all invitees, which in turn send answer messages containing an acceptance, a conditional acceptance or a rejection of the invitation. The user scheduling the meeting can keep track of all received answers. Some users (like managers) may be given the power to send invitations to meetings that are automatically accepted, and thus entered into the invitees' calendars.

### **1.3.11 Free/Busy Lists**

When planning a meeting, it is important to know the availability of the intended participants at a certain date. For this, so-called free/busy lists are created that contain the times at which a certain Kolab user is already scheduled to be in a meeting. Normally, free/busy lists only contain the binary information of whether a user is busy or not, but extended free/busy lists even allow to see what a user is busy with. Users shall have the possibility, however, to mark certain appointments as personal; the contents of these will never be shown.

An additional requirement is that the generation of free/busy lists must even work when the user in question does not have a client started, e.g. when he or she is on vacation. This means that the generation of free/busy lists must be done on the server.

### **1.3.12 Resource Management**

It is possible to define resources (e.g. conference rooms, shared company cars, or beamers). In order to reserve one of these resources, they invited to a meeting just like other users. However, in the case of resources, the Kolab server itself handles the acceptance (in case the resource is available) or rejection (in case the resource is already booked for the requested date and time).

### **1.3.13 Tasks**

Tasks can be created and assigned to other Kolab users. The creator of the task can follow the completion state, which is updated by the assignee as work progresses.

### **1.3.14 Address Books**

Kolab knows global and local address books. Global address books can be used by all users in the installation, are kept on the Kolab LDAP server, and are maintained via a special interface. Local address books are stored in the user's mail account on the IMAP server (as well as replicated on the local hard disk) and are maintained by the user via the ordinary address book interface. When searching for an address for address completion, both the global and the local address book shall be used, the search order is configurable.

### **1.3.15 Out-of-office Messages**

The user shall be able to configure out-of-office messages that are even generated when the user has no client open (e.g., when on vacation). This means that those messages need to be generated on the server.

### 1.3.16 Virus and Spam Filters

The server is capable of integrating virus and spam filters like amavis and spamassassin. This is configurable by the administrator by means of a web interface.

### 1.3.17 Quota Management

Users can be assigned quotas, a maximum amount of storage their mailboxes may occupy on the server. Quota assignment is done by the administrator via a web interface. When a so-called threshold value („soft quota“, often 80% of the real quota) is reached, the server starts to send daily email messages to the user, asking him or her to clean up his or her mailbox. If the user does not react and the hard quota is reached, neither sending nor receiving email messages will be possible until the user has deleted some mail.

### 1.3.18 Clustering

Kolab servers can be run in clusters of two or more computers.

### 1.3.19 Outlook(tm) Collaboration

Kolab was especially conceived to allow coexistence with Outlook(tm) clients. The idea is that by using the Kolab server, users can choose whether to run MS-Windows(tm) with Outlook or Linux with the Kolab client. This requirement poses some particular challenges of its own, as we will see. Of course, as Linux enthusiasts, the Kolab developers see this not so much as a collaboration scenario, but as a migration path („first, we'll get them on our server, then we'll get them on our client“). This even includes providing a migration tool that lets users feed their .pst files (the way Outlook stores data on the local hard disk) into the Kolab server.

### 1.3.20 German Localization

Since the first major installation of the Kolab system was in the German federal government, German localization of all on-screen texts and user documentation was mandatory.

### 1.3.21 Easy Installation

The Kolab server and client should have easy installation packages, preferably one single package for the server and one for the client with as few dependencies on other installed software as possible. Installation should be as easy as running one command from the command line.

## 1.4 The History and Present of the Kolab Project

Around LinuxTag 2002, the consortium of Klarälvdalens Datakonsult AB (Sweden, Denmark, Germany, France), Intevation GmbH (Germany) and Erfrakon GbR (Germany) was awarded a development contract for the first phase of the Kolab client and server by the Bundesamt für Sicherheit in der Informationstechnik (BSI, German Federal Agency for IT Security). The project was nicknamed Kroupware, and the resulting software called the Kolab server and the Kolab client. In conjunction with LinuxTag 2003, Kolab 1.0 was released. Since then, development was continued by an international group of volunteers, and in early 2004, the same consortium was awarded a contract by a new customer for securing professional continued development. The current work distribution is:

- Klarälvdalens Datakonsult AB: Client Design, Client Implementation, Server Implementation (led by Bo Thorsen)
- Erfrakon GbR: Server Design, Outlook Compatibility (led by Martin Konold)
- Intevation GmbH: Project Management (led by Bernhard Reiter)

## 1.5 The Concept

Kolab is a fully open-source solution for Linux, using both open-source software and following open standards such as IMAP and vCal. It is based on existing server and client parts, that are improved and adapted for the needs of the Kolab project. The goal of the project is to solve groupware workflows in companies and projects.

Besides the technical building blocks, Kolab is a defined way of doing groupware scheduling. This means that all clients are able to use Kolab as a mail and web server, but so far only Outlook (with a proprietary plugin, see below) and Kontact, the new KDE-based Kolab client implement the full work flow. It should be fairly easy to adapt Evolution and Mozilla, as well as other open-source email clients and organizers, to support it, too.

### 1.5.1 The Server

The server is based on Postfix, Apache, OpenLDAP, Cyrus IMAPD and an additional Kolab engine. The Kolab engine controls the configuration of the other parts and also takes care of handling things that need to work even when the user in question has no client running (like out-of-office replies or mandatory acceptance of meeting invitations). All configuration data is stored in the LDAP server. Of course, all email clients that support IMAP (including text-based ones like mutt) can use the email part of the Kolab server.

### 1.5.2 The Client

The KDE Kolab client is called Kontact, and is built upon KOrganizer, KMail, KAddressBook, and other components, such as KPilot for synchronization with Palms PDAs. The Kolab project has added the support for the Kolab way of scheduling, i.e., implemented an easy communication between KMail, KOrganizer, and KAddressBook, the possibility to store appointments, contacts, etc., in KMail; implemented the scheduling algorithms, implemented the handling of resources; implemented legacy support for broken Outlook parts; and much more. Using the Ägypten software (see above), Kontact can send invitations with S/MIME encryption and/or signatures (and of course with OpenPGP encryption and/or signatures as well).

Another group of developers is working on extending the Turba and Horde web clients to work with Kolab, resulting in a complete web client solution, something that has been asked for a long time.

## 1.6 Current Work

We hope to release Kolab 2.0 around november 2004, including all the features specified above in the Requirements section. The focus for this release is on two major topics:

- Cleaning up some leftovers from Kolab 1.0, to get a solution that overall feels much better.
- Supporting people in work groups.

The last part means that we will solve the „secretary“ and „project leader“ scenarios. These both mean that you have the potential to write in other people's calendars, if they choose to give you permission to do this. Full ACL support has been added KMail, so users can give others access to their folders.

A very important point is that we will support people who use both Kontact and Outlook, so that they can use Kontact one day and Outlook the next. It will also be possible to have Outlook and Kontact users work with the same folders.

### 1.6.1 kdepim 3.3

It has been decided in the kdepim community to release kdepim 3.3 independently of the rest of KDE. This gives the kdepim people more flexibility regarding release dates, and the kdepim project has decided to stay compatible with KDE 3.2 for easier upgrading. The Kolab 2.0 KDE client is developed in kdepim HEAD currently, so kdepim 3.3 will contain the full Kolab client.

Lots of enhancements have been put into kdepim 3.3 in addition to the new Kolab development, such as:

- Kitchensync (syncs with all kinds of devices)
- HTML mail composing in KMail

- IMAP client-side filtering
- eGroupware support
- full-text indexing in KMail
- per-contact crypto settings
- built-in anti-spam support in KMail
- and much more...

## 1.7 Outlook Compatibility

One of the key challenges in the Kolab project is the compatibility with MS-Outlook(tm). As described above, users are supposed to be able to freely switch between the Kolab client and Outlook, while using the same account.

By default, Outlook connects to an MS-Exchange(tm) server and transfers the data as TNEF-encoded, undocumented MAPI properties, something that does not fit at all into an open source, open protocols scenario. The solution to this is to use a special plugin that converts Outlook into an IMAP client; Outlook will then speak to an email server using the IMAP protocol. Depending on the plugin, the data will even be converted from the proprietary MAPI properties into standard formats like vCal.

Kolab 1.0 used the Bynari Connector for this, which unfortunately had a number of technical problems. Kolab 2.0 will be tested against the Toltec connector, but of course any plugin (there are others in the making, such as the one by Source eXtreme, which will even be open source) that follows the Kolab standards can be used.

## 1.8 Further Information

Further information about the Kolab project is available from the project's web site at <http://www.kolab.org> <<http://www.kolab.org/>>. Further information about Klarälvdalens Datakonsult AB's professional KDE and Qt development services are available from <http://www.kdab.net> <<http://www.kdab.net/>>.

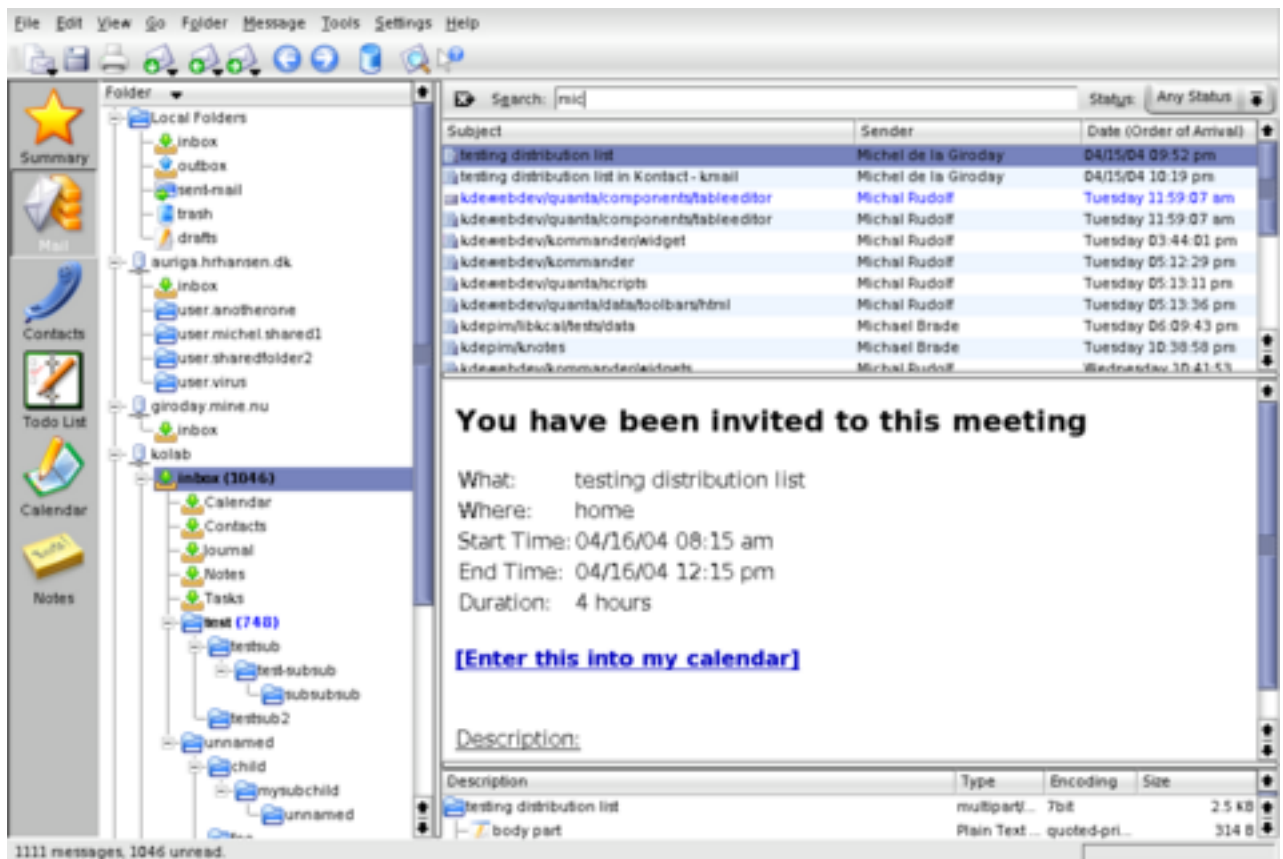


Illustration The contact screen, showing the email component where a meeting invitation was just received. The user can click on the „Enter this into my calendar“ link, which will record the meeting in the user’s calen-



dar; this is an example of the tight integration between the components.

Illustration An example of the web-based administration interface; in this case the user administration. All data entered here is stored in the LDAP server.