

# Free Software Groupware Project

## Architecture Paper



**Erlwein, Frank, Konold & Partner**  
**Beratende Ingenieure und Physiker**  
**Germanenstr. 15**  
**70563 Stuttgart**  
**Germany**  
**<http://www.erfrakon.de>**  
**email: [info@erfrakon.de](mailto:info@erfrakon.de)**

## Free Software Groupware Project: Architecture Paper



by

Erlewein, Frank, Konold & Partner  
Beratende Ingenieure und Physiker  
Germanenstr. 15  
70563 Stuttgart  
Germany  
<http://www.erfrakon.de>  
email: [info@erfrakon.de](mailto:info@erfrakon.de)

Published September 19th, 2002

This documentation was written in SGML using the DocBook DTD. HTML and Postscript output is generated automatically and depends on the tools used.

Windows, Exchange, and Outlook are registered trademarks of Microsoft Corporation Inc. Insight Connector is a registered trademark of Bynari Inc. HotSync is a registered trademark of Palm Inc. All other herein mentioned trademarks belong to their respective owners.

### Revision History

Revision 1.0 September 10th 2002

Revision 1.0.1 September 19th 2002

# Table of Contents

<b>1. Free Software Groupware Overview .....</b>	<b>1</b>
1.1. Email Functionality .....	1
1.2. Contacts.....	1
1.3. Address Book .....	1
1.4. Calendar Entries .....	1
1.5. Notes .....	2
1.6. Task Lists .....	2
1.7. Shared Resources .....	2
1.8. Print Services .....	2
1.9. Palm PDA Synchronization .....	2
<b>2. Communication between Client and Server.....</b>	<b>3</b>
2.1. Protocols.....	3
2.2. Used File Standards .....	3
2.3. Sending Email .....	3
2.4. Receiving Email .....	4
2.5. Vacation Functionality .....	4
2.6. Contacts.....	4
2.7. Address Book .....	4
2.8. Groupware Calendar .....	5
2.9. Notes .....	5
2.10. Task Lists .....	6
2.11. Management of Shared Resources .....	6
2.12. Remarks .....	6
<b>3. The Communication between the Clients.....</b>	<b>7</b>
3.1. Group Event Notifications.....	7
3.1.1. A Simple Calendar Event .....	7
3.1.2. Modify an Existing Calendar Event .....	8
3.1.3. Free-Busy Lists.....	9
3.1.4. Using Free-Busy Lists .....	10
3.2. Message Receive Confirmation.....	10
3.3. Sending Notes .....	10
3.4. Attached Business Cards.....	12
3.5. Multi-User Task Lists.....	12
3.6. HotSync PDA Synchronization.....	13
<b>4. Kolab Server Components .....</b>	<b>14</b>
4.1. OpenLDAP2 Directory Server .....	14
4.1.1. Top Level LDAP Structure .....	14
4.1.2. LDAP Business Card .....	15
4.2. Postfix Mail Server.....	15
4.2.1. LDAP Connection .....	15
4.3. Cyrus IMAP Daemon.....	16
4.3.1. LDAP Connectivity .....	16
4.3.2. Cyrus Sieve.....	17
4.4. ProFTP Daemon.....	17
4.5. Kolab Server Backup Strategy .....	17

4.6. Administrator User Interface.....	18
<b>5. Windows Client.....</b>	<b>19</b>
5.1. Language dependencies .....	19
<b>6. The KDE Client.....</b>	<b>20</b>
6.1. Language dependencies .....	20
<b>A. The KDE client File Formats.....</b>	<b>21</b>
A.1. Calendar Event .....	21
A.2. Note .....	22
A.3. Contact .....	22
A.4. Task Lists.....	23
A.5. Free-Busy Lists .....	24
<b>B. Legacy File Formats.....</b>	<b>26</b>
B.1. Calendar Event .....	26
B.2. Note .....	27
B.3. Contact.....	28
B.4. Task.....	29

# List of Tables

3-1. Calendar Event .....	7
3-2. Maintaining the Free-Busy List.....	9
3-3. Using the Free-Busy List.....	10
3-4. Sharing a Note .....	11
3-5. Sharing a Business Card.....	12
3-6. Assigning Tasks to Other Users .....	12

# Chapter 1. Free Software Groupware Overview

This highly scalable and secure solution is strictly based on a client/server architecture. The server part of the groupware solution, herein referred to as "Kolab Server " runs on GNU/Linux, namely on a current distribution of Debian, Red Hat or SuSE. We minimized the usage of Linux distribution dependencies in order to gain maximal portability. In order to be able to apply readily available security updates in a timely manner employment of a well-maintained GNU/Linux distribution like the above mentioned seems desirable. The goal is that standard security updates as provided by these vendors can be installed without affecting the Kolab server functionality. The free software groupware client application will be developed as a native KDE 3 application. Microsoft Windows NT 4 or Windows 2000 based clients access the Kolab groupware server using Microsoft Outlook 2000 with the Bynari Insight Connector Plugin 1.09 installed (reference platform). A high-level description of the complete free software groupware solution seen from the users point of view follows. The list was created with existing proprietary groupware products in mind, namely Microsoft Exchange Server with Outlook 2000 clients.

## 1.1. Email Functionality

The common internet email functionality is provided: sending email via SMTP over TLS and receive it via IMAP over TLS (preferably) or plain SMTP and IMAP (alternatively for backward compatibility reasons) support for strong cryptography for email bodies and attachments in addition to the security features provided by the transmission protocol (as can be found in the former project "Ägypten") optional support of message receive confirmations (the user gets prompted for an acknowledgment) adding priorities to emails (importance header) vacation message functionality (via web interface) receiving and sending business cards along with email Every standard-conformant mail user agent (MUA) and web browser can be used to access the Kolab server for this functionality.

## 1.2. Contacts

The client application maintains a private address book ("Contacts", German: "Kontakte"). Note that this is very different from a global address book. A users private address book is stored on the Kolab server. A bidirectional vCard interface is provided. In consequence, vCards received as email attachments can be easily added to a users contacts repository.

## 1.3. Address Book

A global shared address book is available independently from the contacts. A user can change his own contact information in this address book by using a web interface on the Kolab server. Address book entries are maintained inside a LDAP directory on the Kolab server and can be exported to vCal format. The necessary conversion is done by the groupware client application.

## 1.4. Calendar Entries

A user can have private calendar events. Although these events are visible for other users, the exact event information is hidden from them. They only see that a private event exists. Private events are saved on the Kolab server. Users can schedule group events and invite other users. An existing group event can be modified by the user who originally created it. Examples for group events are: Group meetings  
Conferences  
Group events are saved like private calendar entries on the Kolab server and differ only by their attributes and access permissions. When creating a new group event, one can check the availability of the desired attendees. We refer to this later as "publishing and checking free-busy lists".

## 1.5. Notes

A user may take private notes which are then stored on the Kolab server. Notes can easily be mailed to other users and therewith shared with them, as copies.

## 1.6. Task Lists

Users can maintain task lists with priorities. Task lists are saved on the Kolab server. Items on the task list can be assigned to different users and are added to their task lists as well, as they receive and accept them. Task lists are private if no items are assigned to other users.

## 1.7. Shared Resources

Shared resources (e. g. meeting rooms or pool cars) can be managed with the solution in the same way as regular group calendar events.

## 1.8. Print Services

Emails, calendar events, task lists, and notes are easily printable by the client application.

## 1.9. Palm PDA Synchronization

The contacts, calendar events, notes, and task lists can be synchronized with a personal digital assistant (PDA) bidirectionally. The HotSync protocol is used and guarantees compatibility to a wide range of PDA devices. The reference platform is a 3Com Palm V running Palm OS v3.1.

# Chapter 2. Communication between Client and Server

A technical description of the communication between KDE clients and the Kolab server follows.

## 2.1. Protocols

The protocols were selected with the following criteria in mind:

proper standardization e.g. by the Internet Engineering Task Force (IETF, <http://www.ietf.org>) open standard in the sense of the term open source existing free software implementations must scale very well in general simplicity over complexity - in other words limitation to a small set of well-established and widely used protocols

This leads to the following choice of protocols used in the project:

- Lightweight Directory Access Protocol Version 3 (LDAP, IETF RFC 2251)
- File Transfer Protocol (FTP, IETF RFC 765)
- Simple Mail Transfer Protocol (SMTP, IETF RFC 2821), SMTP over Secure Socket Layer / Transport Layer Security (SMTP over SSL/TLS, IETF RFC 2246)
- Internet Message Access Protocol (IMAP, IETF RFC 1730), IMAP over SSL/TLS (IETF RFC 2595)
- Post Office Protocol Version 3 (POP3, IETF RFC 1939), POP3 over SSL/TLS (IETF RFC 2595)
- Hyper Text Transfer Protocol (HTTP, IETF RFC 2616), HTTP over SSL/TLS (IETF RFC 2818)
- HotSync Protocol (this does not meet some of the above criteria but seems to be widely accepted and free implementations exist)

## 2.2. Used File Standards

- multi-part MIME email as standardized by the IETF
- vCal and vCard as standardized by the Internet Mail Consortium (<http://www.imc.org>)

## 2.3. Sending Email

The Kolab server runs the Postfix Mail Transfer Agent (MTA) and acts as SMTP relay for all users. It is however possible to decouple this functionality from the mailboxes and establish it on another server, for load-sharing and high availability purposes. Security is provided using two methods:

1. on application level: Email can be encrypted using PGP/GPG

2. on the transport level: the SMTP relay accepts SMTP over TLS connections

For compatibility reasons, plain SMTP is also supported.

## 2.4. Receiving Email

Users read their email via IMAP over TLS on the Kolab server. New user accounts are created on the Kolab server only within IMAP (not as regular GNU/Linux accounts), using the mailbox name `user.<username>`. The latter is a requirement resulting from the Cyrus IMAP daemon. The Postfix MTA on the groupware server delivers email for local users via the Cyrus deliver agent to their respective IMAP mailboxes, which are implemented in the maildir format (not traditional mbox style, every email is a regular GNU/Linux text file). The emails are then further processed by the users client desktop application. Note that Cyrus maps the above mentioned mailbox `user.<username>` to a configurable directory tree on the Kolab server. The email messages are transferred via IMAP synchronization to the KDE client. This synchronization functionality also provides the required offline functionality. The messages are identified via a unique id. Optionally, users can physically receive their email messages via POP3 over SSL from the groupware server and the further processing is done on the client locally. For compatibility reasons, plain IMAP and unencrypted POP3 message transfers are also supported. Note that at this point every SMTP and IMAP capable email Client can access a Kolab server's mailboxes and distribute email messages via the Kolab server, provided the server IP address, the username and the corresponding password.

## 2.5. Vacation Functionality

The vacation functionality is configured by the user via a simple web interface available via HTTPS provided by the Kolab server. The actual vacation functionality is implemented on the server using Sieve (IETF RFC 3028). A free implementation of this scripting language for the Cyrus IMAP daemon is available. Note that Sieve's capabilities reach far beyond a simple vacation functionality. Therefore we gain great flexibility for future extensions. The server side scripting is a sensitive area though, as such activities run contrary to server scalability and performance and therefore must be used with caution if we intend scalability to many thousands of users on commodity hardware.

## 2.6. Contacts

The client application stores contacts in the IMAP sub folder named "Contacts" (German: "Kontakte") on the Kolab server. The actual entries are represented as multi-part MIME emails with included vCard address data as MIME parts. See the appendix for the exact file format.

## 2.7. Address Book

The global address book is stored inside a LDAP directory running on the Kolab server, driven by OpenLDAP v2 (implementing LDAP protocol version 3). The clients access it by using the LDAP v3 protocol. The GUI on the client provides read only access to all address data stored inside the directory. The LDAP scheme is designed in order to allow easy bidirectional conversion to vCard. The need to change someones own address book entry is considered to be rare and therefore a plain web interface is sufficient. A user can modify his own LDAP entry using a HTTPS protected web interface. The look and feel is analogous to the public phone book dialog.

## 2.8. Groupware Calendar

Calendar entries are stored in the users IMAP sub folder "Calendar" (German "Kalender") on the Kolab server. Physically they are represented as multi-part MIME emails with the information stored in vCal format encapsulated in a MIME part. See the appendix for the exact file format and examples.

Group calendar entries are stored just like personal events in the same IMAP sub folder on the Kolab server. The difference between private and group calendar events is simply a matter of the vCal information being part of several users calendars. They are shared via exchanging multi-part MIME emails and have to be accepted by the attendees and therewith put it into their respective calendar folders. The individual free-busy lists consist of a compacted subsets of the vCal data covering a user definable time range into the future. They are published to a shared ressource on the Kolab server. The free-busy lists are referenced via URLs and are retrievable by all registered clients from the Kolab server via HTTPS and optionally via HTTP. The free-busy lists are named as <username>.vcf and follow the vCal file format.

The KDE clients store and lookup free-busy lists using HTTPS on the Kolab server.

Windows Clients must store their free-busy lists via anonymous FTP on the server, for historical reasons. For lookup of the free-busy lists, HTTP is used by the Outlook/Bynari Clients. In this project we look not further into this drawback. Instead, a so-called legacy mode is provided on the Kolab server. Using this compatibility mode the dedicated storage area for free-busy lists on the Kolab server is also accessible via anonymous FTP (upload own free-busy list) and HTTP (retrieving all free-busy lists). This is an optional setting that will be disabled by default.

The logic behind the calendar events and their handling is entirely done by the client applications. The server mainly acts as a network storage device in this regard.

An exception to this rule is the dealing with automatic shared ressources (rooms, technical equipment, cars, etc.).

## 2.9. Notes

Notes are stored on the Kolab server inside the users IMAP sub folder "Notes" (German: "Notizen"). Physically, they are represented as multi-part MIME emails with the actual note being a MIME part. See the appendix for the exact file format.

## 2.10. Task Lists

Task lists are stored on the Kolab server inside the users IMAP sub folder "Tasks" (German: "Aufgaben"). Physically, they are multi-part MIME emails with the actual task list data being a MIME part and following the vCal standard (vToDo, IETF RFC 2446). See the appendix for the exact file format.

## 2.11. Management of Shared Resources

The Kolab server assigns a dedicated IMAP identity to every shared resource. These identities do not differ technically from real users. Reserving a car or a room for example is just arranging a meeting with the shared resource's assigned IMAP user. Two modes of operation are supported:

1. manual mode: a real user monitors a shared resources mailbox in addition to his own mailbox and accepts or declines events on behalf of the shared resource
2. automatic mode: via Sieve scripting the resource mailbox is monitored; the scripting takes care of automatically publishing it's free-busy list and accepts or declines events on the basis of availability of the resource

## 2.12. Remarks

The support for message receive confirmations and attaching business cards is a matter of the end to end communication between the clients. The print services and the Palm OS (HotSync) Synchronization depend fully on the client installation and as such are not related to the client server communication.

# Chapter 3. The Communication between the Clients

In principle all information is exchanged via multi-part MIME email messages between the KDE clients. The receiving user decides about every incoming event, note, task list, etc. and depending on the decision the KDE client application moves the email to the corresponding IMAP folder for further processing. Therefore the client application has means to detect the type of an incoming email and classify it into one category out of note, task list, contact, calendar event, and ordinary email.

The national language support is handled by the clients. UTF-8 encoding is used for special characters. Common ASCII characters are a subset of the UTF-8 encoding. Outlook and the KDE client are interoperable in this respect. This has been verified using german umlaut characters using Outlook 2000 on Windows 2000 and the KDE client on Linux 2.4.

## 3.1. Group Event Notifications

Invitations to a group event are sent via a multi-part MIME email with vCal information included as a MIME part. The user can decide to either accept the event (and thereby import it into his own calendar) or decline it. When an event is declined, an according email is generated and sent back to the originator of the group event. When an event is accepted the event gets added to the users personal calendar and the users own free-busy list gets updated and later published.

There is a special mode that must be used by the KDE client when communicating with a legacy Windows Outlook/Bynari client. In this case calendar events are sent as plain MIME email with the vCal data being the body of the email. This non RFC-conform mode is not recommended by the IETF but the only way to communicate with broken Windows clients. In fact the KDE client can send and receive calendar events in this format, in addition to it's own format as outlined above. We will refer to this mode of operation as "legacy support".

In order to be most compatible to real world szenarios this legacy mode will be the default.

The most important calendar use cases follow. Other cases can be derived from them and therefore are not included herein.

### 3.1.1. A Simple Calendar Event

**Table 3-1. Calendar Event**

Step	Action	Remarks
------	--------	---------

Step	Action	Remarks
1	Originator sends Email with attached vCal to attendees	Calendar event gets saved in originator's own calendar
2	Attendee's receive the calendar event as Email in their INBOX (German "Posteingang")	Attendee has to open the email and is prompted to accept, conditionally accept or decline the event; no storage operation takes place so far
3a	Attendee A accepts the event and sends back a confirmation; attached is the slightly modified vCal (attributes used: DTSTAMP and ATTENDEE)	Calendar event gets saved in attendee A's calendar (subset of original event including participants; only originator has full information e.g. the state of accepts/declines)
3b	Originator gets email with the slightly modified vCal attached	Originator learns about the update and refreshes the event in his calendar after the users confirmation (mark attendee as having accepted)
3c	Attendee B declines the event and sends back a refusal; attached is the slightly modified vCal (attributes used: DTSTAMP and ATTENDEE)	Calendar event is not saved into attendee B's calendar
3d	Originator gets email with the slightly modified vCal attached	Originator learns about the update and refreshes the event in his calendar after the users confirmation (mark attendee as having declined)

### 3.1.2. Modify an Existing Calendar Event

Note that the whole group calendar system does not automatically keep consistency over different users. This is impossible because of the accept/decline choice and the fact that a calendar event can be privately modified.

#### 3.1.2.1. Originator modifies Event

The originator may change any aspect of a calendar event after it was settled between the attendees. If for example a calendar event's duration gets modified, the above process (3.1.1) starts again. In the originator's calendar the state of the attendees therefore gets reset. By having a unique message ID it is guaranteed that the vCal matches again to the same calendar event. This guarantees that a calendar event

can be shifted without ending up to be created multiple times. When new attendees are added or some are deleted, the originator gets prompted on whether the notification (again a vCal entry) should be send to all people involved or only to those who are actually affected by the change.

### 3.1.2.2. Attendee modifies Event

An attendee can modify the time parameters of a calendar event which doesn't originate from himself. But then his calendar falls out of sync. The change will be overwritten if the originator sends an update. A warning concerning this is presented to the user.

### 3.1.3. Free-Busy Lists

KDE clients can optionally (and should) publish their own calendar summary information on the Kolab server. Using this information, other users can check for availability of desired attendees for a proposed calendar event before issuing the actual invitation. Therewith a meeting can be scheduled more efficiently and according to the attendee's free time slots.

A free-busy file follows the vCal format and contains only the calendar event time data for a given user. The user can choose how long into the future his free-busy information should be published on the Kolab server. On the Kolab server all free-busy data is stored and is accessible in different ways for legacy and KDE clients.

The KDE clients store their "Free-Busy" information on the Kolab server using Web-DAV for upload and HTTPS for download. The legacy clients use anonymous FTP to upload their free-busy information into the very same directory as the KDE clients do. Accessing the free-busy information is done via HTTP on the legacy clients.

In configurable intervals, the client application publishes free-busy data:

**Table 3-2. Maintaining the Free-Busy List**

Step	Action	Remarks
1	Get through all calendar data from "now" to a time T in the future (default: two months)	the KDE client holds most of it's calendar data in the process memory, so this does not necessarily mean an overhead
2	Compute a consolidated vCalendar based on the vCal data	a collection of vCal data is called vCalendar

Step	Action	Remarks
3	Publish the vCalendar free-busy list on the Kolab server	This is preferably done using Web-DAV via HTTPS; Windows clients use anonymous FTP and retrieve via HTTP (Kolab legacy mode)

### 3.1.4. Using Free-Busy Lists

When a user proposes a new calendar event, he chooses the participants and at the same time, transparently, the free-busy information of the according users is retrieved from the Kolab server for display during the creation of the event.

**Table 3-3. Using the Free-Busy List**

Step	Action	Remarks
1	the KDE client learns about attendees in a dialog	User enters a list of email addresses in the calendar event dialog
2	Display attendee's calendar schedule (only times) using a simple bar diagram and let the user choose a free time frame	Retrieve free-busy list of that user (filename is computable: <user>.vcf) by HTTPS or, in the windows case with HTTP (Kolab legacy mode)

## 3.2. Message Receive Confirmation

When a message with a request for receive confirmation arrives, the user is prompted whether a confirmation shall be send to the originator of the according email. The system is based on the corresponding IETF RFC standards and supports the following actions:

- disallow generation of the receive confirmation (privacy mode)
- allow generation of the receive confirmation (collaboration mode)
- the user on who's behalf the receive message is generated can assign a different email address to receive the confirmation

### 3.3. Sending Notes

Usually Notes contain only text and they are saved on the Kolab server in a special IMAP subfolder of the creating user. Notes can be categorized. German standard categories are:

- Favoriten
- Feiertag
- Festtagsgrüße
- Geschäftlich
- Geschenke
- Hauptkunde
- Ideen
- International
- Konkurrenz
- Persönlich
- Schlüsselpersonen
- Status
- Strategien
- Telefonanrufe
- Verschiedenes
- Wartet
- Wichtige Kontakte
- Zeit und Ausgaben
- Ziele
- Zulieferer

The list of categories can be edited by the user. New categories can be added, existing entries can be modified or deleted.

Notes are saved as multi-part MIME messages on the Kolab server. The first part of the email is of type text/plain and has the textual representation of the note. The second part of the message consists of the note with further attributes encoded as SMTP headers. For sending notes to Windows clients a special legacy format must be readable and writeable by the KDE client. See the appendix for the exact file format.

Step	Action	Remarks
------	--------	---------

Table 3-4. Sharing a Note

Step	Action	Remarks
1	Forward note to another user	Note is sent as multi-part MIME email message (two parts - readable text)
2	As the recipient opens the email he can import the note into his own note folder	Note is moved into notes folder, so it's shared by creating a copy

### 3.4. Attached Business Cards

Users can exchange business cards (contacts or his own personal data). The card can be exchanged by email - thereby using the format VCARD inside a part in a multi-part MIME email (content type text/x-vcard). Business cards can contain categories (see above). Windows clients can attach OLE objects like MS-Word, Excel, PowerPoint or similar. The free software client does ignore OLE data, as it is not usable on non-Microsoft platforms. Attaching complete files or sending URL references is the preferred portable and more efficient way. For compatibility reasons, platform independence and severe security concerns the free software client does not make use of OLE objects.

Table 3-5. Sharing a Business Card

Step	Action	Remarks
1	Forward business card to another user	Note is sent as multi-part MIME email
2	As the recipient opens the email he can import the business card into his contacts	multi-part MIME email gets stored inside the users private contacts folder

### 3.5. Multi-User Task Lists

A task can be send to other users via a multi-part MIME email. The task list is represented by a MIME part containing a vCal (vToDo). For sending tasks to Windows clients a special legacy format must be readable and writeable by the KDE client. See the appendix for the exact file format.

**Table 3-6. Assigning Tasks to Other Users**

<b>Step</b>	<b>Action</b>	<b>Remarks</b>
1	Create a task for another user in the own task list and send	The task is sent as multi-part MIME email containing a vToDo
2	As the recipient opens the email he can import the task into his own task list	Task (multi-part MIME email) is moved into the users tasks folder

Windows clients can attach OLE objects like MS-Word, Excel, PowerPoint or similar. The free software client does ignore OLE data, as it is not usable on non-Microsoft platforms. Attaching complete files or sending URL references is the preferred portable and more efficient way. For compatibility reasons, platform independence and severe security concerns the free software client does not make use of OLE objects.

### **3.6. HotSync PDA Synchronization**

The KDE client synchronizes using existing HotSync software (KDE Kitchensync and Conduits). Windows clients synchronize using existing proprietary software, which probably also make use of the HotSync Protocol. Further investigation is not done here.

# Chapter 4. Kolab Server Components

Whatever configuration data can be held inside the LDAP directory will be stored there. Still, there is the need for individual configuration files of these services, that are run on the Kolab server:

1. Postfix
2. Cyrus
3. Apache
4. Inet Daemon
5. ProFTPD
6. OpenLDAP2

## 4.1. OpenLDAP2 Directory Server

LDAP keeps the central user data including credentials. Cyrus authenticates against LDAP via the Cyrus-sasl library and pam-ldap. It is part of the security concept of the Kolab Groupware server to not actually have the groupware users as GNU/Linux shell accounts on the server.

Alternativ implementations may also use different ways for authentication but then these must also create their own derivatives of the administration tools.

### 4.1.1. Top Level LDAP Structure

```
The LDAP scheme for the administrator:  
dn: cn=administrator, c=DE  
objectclass: organizationalPerson  
objectclass: person  
objectclass: Top  
cn: administrator  
uname: administrator  
userpassword: <password>
```

The top hierarchy looks like:

```
1. dn: c=DE  
objectclass: country  
objectclass: Top  
c: DE
```

```

2. dn: o=BSI, c=DE
   objectclass: organization
   objectclass: Top
   o: BSI
3. dn: ou=EDV, o=BSI, c=DE
   objectclass: organizationalUnit
   objectclass: Top
   ou: EDV

```

### 4.1.2. LDAP Business Card

Attributes of an LDAP Business Card are similar to the following example:

```

dn: cn=Hans Schmid, ou=EV, o=BSI, c=DE
objectclass: organizationalPerson
objectclass: person
objectclass: Top
display-name: Hans
givenname: Hans
sn: Schmid
mail: schmid@bsi.de
userpassword: <password>
passdate: 1014557980
cn: Hans Schmid
uname: hschmid
mailalias: hans.schmid@bsi.de

```

Further entries can be easily added to the structure as the project advances and all requirements are identified.

## 4.2. Postfix Mail Server

Postfix is a scalable, secure implementation of a SMTP mail transfer agent for UNIX operating systems. It is part of nearly all Linux distributions today. We try to avoid strong dependencies on individual or version dependent features. The client server architecture is modeled in such a way that we can even replace Postfix with another capable MTA like Exim or Zmailer later, if desired.

### 4.2.1. LDAP Connection

Postfix uses the LDAP storage for the following options:

```
ldapsource_server_host  LDAP Server IP
ldapsource_server_port  TCP Port
ldapsource_timeout     Query Timeout
ldapsource_result_attribute
ldapsource_bind
ldapsource_bind_dn
ldapsource_bind_pw
ldapsource_search_base
ldapsource_query_filter
```

At runtime Postfix retrieves the following parameters from the LDAP directory:

```
alias_maps  replaces /etc/aliases
canonical_maps Canonical Address Mapping (inc. envelope)
lmtp_sasl_password_maps
local_recipient_maps
recipient_canonical_maps
relocated_maps
sender_canonical_maps
smtp_sasl_password_maps
transport_maps Static SMTP Routes
virtual_maps Virtual Address Mapping
```

## 4.3. Cyrus IMAP Daemon

Cyrus is a widely-spread IMAP daemon for Unix environments. It stores emails using the maildir format. That basically results in a directory structure with each email represented by single file. The IMAP users are managed completely separate from the GNU/Linux system accounts. User quotas and access control list are provided by the daemon independent of the operating system. Cyrus' most interesting feature however is its excellent scalability. A single Cyrus instance can serve up to thousand simultaneous interactive IMAP users. Usually, Cyrus IMAPD is basically I/O bound. This means that the performance of the filesystem storage and network bandwidth/latency are mainly determining the maximal scalability. Both of them need to be maximized to optimize the IMAP server's performance on a given hardware. During runtime, approximately one megabyte of main memory per concurrent IMAP instance (that is one interactive user) is to be estimated.

### 4.3.1. LDAP Connectivity

Cyrus can make use of an LDAP directory as backend for its authentication mechanism. This is a widely used feature and can be determined to be very practical and functional.

### 4.3.2. Cyrus Sieve

The historic Unix tool procmail can not be used to filter incoming email for Cyrus IMAP users. This is due to a special delivery procedure which must be followed to deliver mail to a local IMAP user. In addition heavy usage of procmail limits the scalability of the server significantly. The Cyrus IMAP daemon supports therefore a special scripting language that is tightly coupled to the running daemon. This language is called Sieve and it is fully standardized by the IETF. Sieve is a general purpose message filtering language. In this project, we use its capability for creating vacation messages and to handle managed shared resources automatically.

Server scripting and scalability are a sensitive issue. The scalability of the whole groupware solution depends heavily on the fact that all processor intensive logic is implemented on the client side. If the server has to process additional CPU- or I/O-intensive tasks then a performance degradation is to be expected. Putting the load on the clients basically means that the total processing power is increased with the number of clients.

The design principle for employing server side scripting in scalable solutions is: "avoid resource intensive server operations - smart clients scale better than smart servers"

## 4.4. ProFTP Daemon

ProFTPD offers good security features such as change root environment and a fine granular access configuration. Herein it can be configured to allow a store-only incoming directory for the free-busy lists for all groupware users. Apart from that it is completely replaceable by any other standard compliant FTP daemon, if desired. Its only functionality on the Kolab server is the legacy mode to enable Windows clients to publish their free-busy lists via anonymous FTP on the server. If only KDE clients connect, the FTP functionality is not needed. It is deactivated by default, for security reasons.

## 4.5. Kolab Server Backup Strategy

One of the biggest advantages of using the maildir format on the Kolab server is, that no special procedures need to be designed to deal with the well known open file issues of other solutions. A differential backup strategy with a few incremental data backups can afford to just ignore the fact that some files throughout the IMAP server filesystem structure may actually be opened during backup time. It is also possible to restore single mailboxes or even single emails with small to zero additional effort, compared to restoring any other file on the system. This is one of the biggest benefits over commercially

available solutions which sometimes operate on a single large binary object, that actually would require a shutdown for a save backup procedure.

## 4.6. Administrator User Interface

The Kolab server is administrated via a PHP based webinterface protected by HTTPS. The administrator interface is divided into the following subsection:

1. Server Setup
2. Setup of Server Name, Domain Name, General Email Option, Enable/Disable of Legacy modes
3. Logging
4. Setup, View and Modify Logging Options.
5. View and download logfiles.
6. Vacation Setup
7. Administrator interface for vacation messages
8. User Accounts: individual accounts with all user specific data including password, electronic business cards and the language setting for Microsoft Outlook.

# Chapter 5. Windows Client

The Windows client application to cooperate with the Kolab server and the KDE client is Outlook 2000 with the Bynari Insight Connector 1.09 plugin installed. No modifications are made to this proprietary software. But some optional settings in the "Extras->Options" dialog must be made to insure interoperability:

1. "Einstellungen->Kalenderoptionen" "Besprechungsanfragen standardmäßig als iCAL senden"  
"Frei/Gebucht Optionen->Unter dieser URL veröffentlichen" URL: ftp://<kolab server>/freebusy/%NAME%.vcf "Frei/Gebucht Optionen->Unter dieser URL suchen" URL: http://<kolab server>/freebusy/%NAME%.vcf
2. "Email Format" "Senden im Nachrichtenformat" : "Nur Text"
3. "Internet Mail" "MIME : Quoted Printable"

Outlook with the Bynari Insight Connector operates on the IMAP folder significantly different. The file format for storing notes, task lists, calendar events, and contacts is a multi-part MIME message with TNEF (Transport Neutral Encapsulation Format) encoded MIME part being the actual information carrier. This is a proprietary Microsoft format. However, with the above adjustments made to the configuration of Outlook it can be advised to send and receive calendar events, notes, contacts, and task lists via the almost open legacy format we describe later in the appendix. The journal functionality of Outlook/Bynari will not be supported by the KDE client as of now. Windows users can however use this without the group functionality.

## 5.1. Language dependencies

Due to the fact that Microsoft Outlook does not only localize the user visible interface but also internal data including non user visible filenames, directory names, etc. it is not possible to change the language of the Microsoft Outlook without loosing the old data.

The language used for an account is set either with the first use of Outlook with the corresponding language or set by the administrator in advance in case the KDE client will be used before Outlook.

# Chapter 6. The KDE Client

The KDE client is implemented using these existing Free Software KDE projects and their technologies. It is intended to feed back this development into the existing Free Software projects:

- KMail (with Ägypten/SPHINX encryption extensions)
- KOrganizer (used as KPart)
- KAddressbook
- Kitchensync with Conduits

The following implementation activities have been identified so far:

- Couple KOrganizer with Kmail
- Add offline support to the IMAP kioslave
- Extend the GUI to support further functionality
- simplify and standardize the VCAL parser implementation throughout KDE
- implement logic for groupware functionality
- implement further features as given in the previous chapters
- development of migration tools to convert mailboxes between usage with the KDE client and Outlook/Bynari (Outlook/Bynari stores all information using the Microsoft proprietary TNEF format)

A more detailed description is out of the focus of this document and will come available as the project advances.

## 6.1. Language dependencies

In order to stay compatible to Microsoft Outlook clients it is required that the KDE client determines the language used to access folders on the Kolab server. The KDE client shall be able to use a different language for storing than for its own user interface. Basically this means that a single binary of the KDE client will be interoperable to any localized version of MS Outlook.

The language used for an account is set either with the first use of Outlook with the corresponding language or set by the administrator in advance in case the KDE client will be used before Outlook. The administrator hints the KDE client by creating the required folders in the corresponding language. The KDE client creates a unique hash over the folder names and therefore can safely determine the correct mapping of the folder names to its internal data structures.

# Appendix A. The KDE client File Formats

## A.1. Calendar Event

(conformant to IETF RFC 2445)

```
Received: ...
From: "Tassilo Erlewein" <tassilo@kolabserver.kde.org>
To: <test1@kolabserver.kde.org>
Subject: WG: Wichtige Notiz
Date: Mon, 9 Sep 2002 17:26:22 +0200
Message-ID: ...
MIME-Version: 1.0
Content-Type: multipart/mixed;
        boundary="-----_NextPart_000_0013_01C25826.04F609F0"
Importance: Normal
```

This is a multi-part message in MIME format.

```
-----_NextPart_000_0013_01C25826.04F609F0
Content-Type: text/plain;
        charset="iso-8859-1"
Content-Transfer-Encoding: 7bit
```

This is an alternative representation of a TEXT/CALENDAR MIME Object

```
When: 7/1/1997 10:00AM PDT - 7/1/97 10:30AM PDT
Where:
Organizer: fool@example.com
Summary: Phone Conference
```

```
-----_NextPart_000_0013_01C25826.04F609F0
Content-Type: text/calendar; method=REQUEST;
        charset="utf-8"
Content-Transfer-Encoding: quoted-printable
```

```
BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VEVENT
ORGANIZER:mailto:fool@example.com
ATTENDEE;ROLE=CHAIR;ATTSTAT=ACCEPTED:mailto:fool@example.com
ATTENDEE;RSVP=YES;TYPE=INDIVIDUAL:mailto:foo2@example.com
DTSTAMP:19970611T190000Z
DTSTART:19970701T170000Z
DTEND:19970701T173000Z
SUMMARY:Phone Conference
```

```

UID:calsvr.example.com-8739701987387771
SEQUENCE:0
STATUS:CONFIRMED
END:VEVENT
END:VCALENDAR
-----_NextPart_000_0013_01C25826.04F609F0--

```

## A.2. Note

```

Received: ...
From: "Tassilo Erlewein" <tassilo@kolabserver.kde.org>
To: <test1@kolabserver.kde.org>
Subject: WG: Wichtige Notiz
Date: Mon, 9 Sep 2002 17:26:22 +0200
Message-ID: ...
MIME-Version: 1.0
Content-Type: multipart/mixed;
        boundary="-----_NextPart_000_0013_01C25826.04F609F0"
Importance: Normal

```

This is a multi-part message in MIME format.

```

-----_NextPart_000_0013_01C25826.04F609F0
Content-Type: text/plain;
        charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

```

(text representation of the note, yet to be determined)

```

-----_NextPart_000_0013_01C25826.04F609F0
Content-Type: text/x-note
        charset="utf-8"
Content-Transfer-Encoding: quoted-printable

```

Here comes the note text ...

```

-----_NextPart_000_0013_01C25826.04F609F0--

```

## A.3. Contact

```

Received: ...
From: "Tassilo Erlewein" <tassilo@kolabserver.kde.org>
To: >test1@kolabserver.kde.org>

```

Subject: WG: Wichtige Notiz  
 Date: Mon, 9 Sep 2002 17:26:22 +0200  
 Message-ID: ...  
 MIME-Version: 1.0  
 Content-Type: multipart/mixed;  
     boundary="-----\_NextPart\_000\_0013\_01C25826.04F609F0"  
 Importance: Normal

This is a multi-part message in MIME format.  
 -----\_NextPart\_000\_0013\_01C25826.04F609F0  
 Content-Type: text/plain;  
     charset="iso-8859-1"  
 Content-Transfer-Encoding: 7bit

(text representation of the vcard, to be determined)

-----\_NextPart\_000\_0013\_01C25826.04F609F0  
 Content-Type: text/x-vcard;  
     name="Kalle Dalheimer.vcf"  
 Content-Transfer-Encoding: 7bit  
 Content-Disposition: attachment;  
     filename="Kalle Dalheimer.vcf"

```
BEGIN:VCARD
VERSION:2.1
N:Dalheimer;Kalle
FN:Kalle Dalheimer
ORG:Klarvielens Kunsult
TITLE:CEO
TEL;WORK;VOICE:1413413241
REV:20020909T150816Z
END:VCARD
```

-----\_NextPart\_000\_0013\_01C25826.04F609F0--

## A.4. Task Lists

(conformant to IETF RFC 2445)

Received: ...  
 From: "Tassilo Erlewein" <tassilo@kolabserver.kde.org>  
 To: <test1@kolabserver.kde.org>  
 Subject: WG: Wichtige Notiz  
 Date: Mon, 9 Sep 2002 17:26:22 +0200  
 Message-ID: ...  
 MIME-Version: 1.0  
 Content-Type: multipart/mixed;

```

        boundary="-----_NextPart_000_0013_01C25826.04F609F0"
Importance: Normal

```

This is a multi-part message in MIME format.

```

-----_NextPart_000_0013_01C25826.04F609F0
Content-Type: text/plain;
        charset="iso-8859-1"
Content-Transfer-Encoding: 7bit

```

(text representation to ToDo, yet to be determined)

```

-----_NextPart_000_0013_01C25826.04F609F0
Content-Type: text/calendar
Content-Transfer-Encoding: 7bit

```

```

BEGIN:VCALENDAR
PRODID:-//ACME/DesktopCalendar//EN
METHOD:REQUEST
VERSION:2.0
BEGIN:VTODO
ORGANIZER:Mailto:A@example.com
ATTENDEE;ROLE=CHAIR:Mailto:A@example.com
ATTENDEE;RSVP=TRUE:Mailto:B@example.com
ATTENDEE;RSVP=TRUE:Mailto:C@example.com
ATTENDEE;RSVP=TRUE:Mailto:D@example.com
DTSTART:19970701T170000Z
DUE:19970722T170000Z
PRIORITY:1
SUMMARY:Create the requirements document
UID:calsrv.example.com-873970198738777-00@example.com
SEQUENCE:0
DTSTAMP:19970717T200000Z
STATUS:Needs Action
END:VTODO
END:VCALENDAR

```

```

-----_NextPart_000_0013_01C25826.04F609F0--

```

## A.5. Free-Busy Lists

(conformant to IETF RFC 2445, note that this is not an email format)

```

BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//RDU Software//NONSGML HandCal//EN
BEGIN:VFREEBUSY

```

ORGANIZER:MAILTO:jsmith@host.com  
DTSTART:19980313T141711Z  
DTEND:19980410T141711Z  
FREEBUSY:19980314T233000Z/19980315T003000Z  
FREEBUSY:19980316T153000Z/19980316T163000Z  
FREEBUSY:19980318T030000Z/19980318T040000Z  
URL:http://www.host.com/calendar/busytime/jsmith.ifb  
END:VFREEBUSY  
END:VCALENDAR

# Appendix B. Legacy File Formats

To visualize the format of the appropriate message type we give examples of messages which were sent by Outlook with Bynari plugin and stored on the Cyrus IMAP server. In addition to the formats presented here it is necessary that the KDE client can handle multi-part MIME emails with TNEF encoded attachments (known as "winmail.dat", containing MAPI object information). To explain the TNEF format is out of the scope of this document.

## B.1. Calendar Event

```
Received: ...
From: "Achim Frank" <achim@kolabserver.kde.org>
To: <test1@kolabserver.kde.org>
Subject: test1 - meeting
Date: Sun, 8 Sep 2002 18:49:21 +0200
Message-ID: ...
MIME-Version: 1.0
Content-Type: text/calendar; method=REQUEST;
    charset="utf-8"
Content-Transfer-Encoding: quoted-printable
X-Priority: 3 (Normal)
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook CWS, Build 9.0.2416 (9.0.2911.0)
Importance: Normal
X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2919.6700

BEGIN:VCALENDAR
PRODID:-//Microsoft Corporation//Outlook 9.0 MIMEDIR//EN
VERSION:2.0
METHOD:REQUEST
BEGIN:VEVENT
ATTENDEE;CN=3Dtest1@kolabserver.kde.org;ROLE=3DREQ-PARTICIPANT;RSVP=3DTRUE:MAILTO=
:test1@kolabserver.kde.org
ORGANIZER:MAILTO:achim@kolabserver.kde.org
DTSTART:20020908T193000Z
DTEND:20020908T200000Z
LOCATION:daheim
TRANSP:OPAQUE
SEQUENCE:0
UID: ...
DTSTAMP:20020908T164921Z
DESCRIPTION:Zeit: Sonntag\, 8. September 2002 21:30-22:00 (GMT+01:00)
    Amsterdam\, Berlin\, Bern\, Rom\, Stockholm\, Wien.\nOrt:
    daheim\n\n*~*~*~*~*~*~*~*~*~*\n\ntest1 - einladung
SUMMARY:test1 - meeting
PRIORITY:5
CLASS:PUBLIC
BEGIN:VALARM
```

TRIGGER:PT15M  
 ACTION:DISPLAY  
 DESCRIPTION:Reminder  
 END:VALARM  
 END:VEVENT  
 END:VCALENDAR

## B.2. Note

Received: ...  
 From: "Tassilo Erlewein" <tassilo@kolabserver.kde.org>  
 To: <test1@kolabserver.kde.org>  
 Subject: WG: Wichtige Notiz  
 Date: Mon, 9 Sep 2002 17:26:22 +0200  
 Message-ID: ...  
 MIME-Version: 1.0  
 Content-Type: multipart/mixed;  
     boundary="-----\_NextPart\_000\_0013\_01C25826.04F609F0"  
 X-Priority: 3 (Normal)  
 X-MSMail-Priority: Normal  
 X-Mailer: Microsoft Outlook CWS, Build 9.0.2416 (9.0.2911.0)  
 Importance: Normal  
 X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2919.6700

This is a multi-part message in MIME format.

-----\_NextPart\_000\_0013\_01C25826.04F609F0  
 Content-Type: text/plain;  
     charset="iso-8859-1"  
 Content-Transfer-Encoding: 7bit

-----\_NextPart\_000\_0013\_01C25826.04F609F0  
 Content-Type: message/rfc822  
 Content-Transfer-Encoding: 7bit  
 Content-Disposition: attachment

Subject: Wichtige Notiz  
 Date: Tue, 3 Sep 2002 18:13:16 +0200  
 MIME-Version: 1.0  
 Content-Type: text/plain;  
     charset="iso-8859-1"  
 Content-Transfer-Encoding: 8bit  
 X-Priority: 3 (Normal)  
 X-MSMail-Priority: Normal  
 X-Mailer: Microsoft Outlook CWS, Build 9.0.2416 (9.0.2911.0)

Importance: Normal  
 X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2919.6700

hier eine sehr wichtige Notiz:  
 bla bla bla

-----\_NextPart\_000\_0013\_01C25826.04F609F0--

## B.3. Contact

Received: ...  
 From: "Tassilo Erlewein" <tassilo@kolabserver.kde.org>  
 To: <test1@kolabserver.kde.org>  
 Subject: WG: Kalle Dalheimer  
 Date: Mon, 9 Sep 2002 17:08:26 +0200  
 Message-ID: ...  
 MIME-Version: 1.0  
 Content-Type: multipart/mixed;  
     boundary="-----\_NextPart\_000\_0002\_01C25823.836243B0"  
 X-Priority: 3 (Normal)  
 X-MSMail-Priority: Normal  
 X-Mailer: Microsoft Outlook CWS, Build 9.0.2416 (9.0.2911.0)  
 Importance: Normal  
 X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2919.6700

This is a multi-part message in MIME format.

-----\_NextPart\_000\_0002\_01C25823.836243B0  
 Content-Type: text/plain;  
     charset="iso-8859-1"  
 Content-Transfer-Encoding: quoted-printable

-----\_NextPart\_000\_0002\_01C25823.836243B0  
 Content-Type: text/x-vcard;  
     name="Kalle Dalheimer.vcf"  
 Content-Transfer-Encoding: 7bit  
 Content-Disposition: attachment;  
     filename="Kalle Dalheimer.vcf"

BEGIN:VCARD  
 VERSION:2.1  
 N:Dalheimer;Kalle  
 FN:Kalle Dalheimer  
 ORG:Klarvielens Kunsult  
 TITLE:CEO

TEL;WORK;VOICE:1413413241  
 REV:20020909T150816Z  
 END:VCARD

-----=\_NextPart\_000\_0002\_01C25823.836243B0--

## B.4. Task

Received: ...  
 From: "Tassilo Erlewein" <tassilo@kolabserver.kde.org>  
 To: <test1@kolabserver.kde.org>  
 Subject: WG: sdljlsjsg  
 Date: Mon, 9 Sep 2002 17:29:17 +0200  
 Message-ID: ...  
 MIME-Version: 1.0  
 Content-Type: multipart/mixed;  
     boundary="-----=\_NextPart\_000\_0018\_01C25826.6D250760"  
 X-Priority: 3 (Normal)  
 X-MSMail-Priority: Normal  
 X-Mailer: Microsoft Outlook CWS, Build 9.0.2416 (9.0.2911.0)  
 Importance: Normal  
 X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2919.6700

This is a multi-part message in MIME format.

-----=\_NextPart\_000\_0018\_01C25826.6D250760  
 Content-Type: text/plain;  
     charset="iso-8859-1"  
 Content-Transfer-Encoding: 7bit

-----=\_NextPart\_000\_0018\_01C25826.6D250760  
 Content-Type: message/rfc822  
 Content-Transfer-Encoding: 7bit  
 Content-Disposition: attachment

From: "Tassilo Erlewein" <tassilo@kolabserver.kde.org>  
 To: <tassilo@kolabserver.kde.org>  
 Subject: sdljlsjsg  
 Date: Tue, 3 Sep 2002 18:29:56 +0200  
 MIME-Version: 1.0  
 Content-Type: text/plain;  
     charset="iso-8859-1"  
 Content-Transfer-Encoding: 7bit  
 X-Priority: 3 (Normal)  
 X-MSMail-Priority: Normal

X-Mailer: Microsoft Outlook CWS, Build 9.0.2416 (9.0.2911.0)  
Importance: Normal  
X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2919.6700

-----\_NextPart\_000\_0018\_01C25826.6D250760--