How to handle (extended) Freebusy Lists - Concept for improving Kolab

Version 0.92+dev20080503

WARNING – this is a work in progress, the document will change, especially it's structure.

It is made available early as it contains some rationale how Server 2.2 will be/already is implemented.

Intevation GmbH

KLARÄLVDALENS DATAKONSULT AB

12.11.2007	0.91	Added Design principles. Changed towards one read-access control-list. Added needed implementation section.	Bernhard Reiter
26.3.2008	0.92	Considered feedback and restruction suggestions from Gunnar Wrobel. Describing Outlook's FB issues.	Bernhard Reiter
X.2008	0.92+dev20080503	It will be „pxfb“ and „pxfb-readable-for". Added TODO for client behaviour.	Bernhard Reiter

Notations

fb freebusy list
pfb partial freebusy list
xfb extended freebusy list
pxfb partial extended freebusy list

TODO: This document shall supplement freebusy.txt not supercede it.
TODO: Clients should not display old freebusy lists without warning. See kolab/issue2622 (usability problem with cached freebusy lists: Very old ones might be shown without network.)

Related documents

http://kolab.org/cgi-bin/viewcvs-kolab.cgi/doc/architecture/freebusy.txt, Rev 1.17 ((2005-06-30) and 1.18 (2008-03-26). Rev. 1.17 has been the current spec for server and clients implementation for freebusy lists.

https://wald.intevation.org/plugins/scmsvn/viewcvs.php/trunk/research/outlook-fbtype.txt?root=kolab describes how Outlook handles different FBTYPES in freebusys lists.

[Storage20rc5] Kolab2 Storage Format Specification, Version 2.0rc5, available from http://www.kolab.org/documentation.html This describes the Kolab-XML storage format.

[Architecture2006] Kolab2 Architecture Draft Version Draft cvs20060921, available from http://www.kolab.org/documentation.html .

RFC 2445, defines the syntax for freebusy lists.

"IMAP METADATA Extension", Cyrus Daboo, 27-Feb-07,
`http://www.ietf.org/internet-drafts/draft-daboo-imap-annotatemore-11.txt`

RFCs 2446, 2739, 4791

Aim

With Kolab we want to be able to generate and download freebusy (fb) lists describing the free and busy times of a user. The plain fb is mainly used together with time overviews.

The most prominent use case is inviting participants or resources to an event. In this case a tabular ganttchart is the preferred way to present the availability of the invited participants. The status of a participant is typically visually coded into either colours or patterns. This use of freebusy lists can be considered standard use, their format is described in RFC 2445 and several clients already implement this. Most important is Outlook.

A further use case is gaining a quick overview about the availability of people and resources. E.g. a team leader is interested in quickly finding for a specific point in time, what her team members are busy with and where. It should be possible for users to have private appointments for which the team leader does not see the details.

For this use case, the standard freebusy lists were extended for the Kolab 2 Concept. Currently the extended freebusy lists contain the topic of an event and are accessible to those that have read access to the folder in question.
But for the use case described above, additional functionality is needed.

The need to model this use case with an extended freebusy list arises from privacy concerns and the technical limitation of Outlook 2003, which can only handle one main calender folder fully. Other folders can be displayed in Outlook 2003, but they behave differently regarding reminders, Outlook's internal account of freebusy lists and receiving email updates to the appointments.

This concept focusses on the extended attributes "location" and "subject". If a team leader requires further details he is requested to obtain read access to the calendar folder instead.

Design principles

There are two important design principles applied here

1. „as complex as necessary and as simple as possible“, this is important so that users can develop an understanding who has access to which part of their data. Each new access mode or configuration will raise the chance that users get confused about the options and reveal information to the wrong party.
2. Be prepared for changes of this concept in the future. After users will have been exposed to the solution there will be a much better understanding of the use case. It is important that the concept presented here does not constitute a dead-end then.

Problems

With Kolab we offer the user many different calendar folders. The freebusy list of an user shall reflect the real situation as good as possible in order to provide a useful hint to other users who intend to interact with them.

Let us consider one use case: There is a group account with a calender folder. Three people have write access and agree to each other to use this folder in a way that they all three need to participate to the events saved in there. Any of them can just write directly in the folder. A confirmation from the others is not necessary. No invitation email is send from one to the other two. To be useful the events would need to be freebusy relevant to all three people – thus show up in their fb list. There are other use cases where events from a folder a user as access to shall not end up in the fb list of the user.

The user must be able to explicitly grant extended freebusy information access to other users and groups for each specific calendar folder. The default mode shall be to deny access.

We require a different folder-specific access mode.

New folder-specific access controls

- **none** this is the **implicit** default for all users and groups
- **pxfb-readable-for** explicitly grant access to the extended freebusy list to a list of users and groups

It would be possible to add a permission for each additional field, in this case for „subject" and „location". This is disregarded because it is not yet clear if the increased complexity will be necessary for the potential use case. There are many appointments where „subject" and „location" belong together or can be concluded from each other. An example is an appointment at the „Dentist" on „Brady Street 10". The user would need to keep in mind in which folder the xfb readers could see one or the other and act accordingly.

In addition to these folder specific settings the privacy needs are to be handled for each individual event, when the pxfb is generated.

Event specific control of freebusy list generation

The normal and extended freebusy lists information are filtered out from regular event objects. There are settings in the event to influence what is saved in the lists. We need to offer something that maps from the options that current client implementations are already offering. [Storage20rc5] Chapter 9 Format of Events, describes four values for the tag **show-time-as** being one of free, tentative, busy, or outofoffice. The value have the following effect on the fb list generation:

- **busy, tentative, outofoffice** The time of the event will be marked as BUSY
- **free** The event will not be entered in the fb list.

Some clients only offer the explicit choice **busy** or **free** (e.g. Kontact Proko2 2.1.12 or enterprise35 20080321).

Some clients additionally offer GUI elements to set the sensitivity of an event. [Storage20rc5] section 4.1. Common Fields In All Types, reflects this in the attribute **sensitivity** being one of private, confidential and public. Kontact offers to set all three values, while Outlook 2003 only as a bi-value toggle for „privat". The sensitivity value affect the creation of attributes towards the xfb.

- **public** this is the implicit default, will add extended attributes
- **private** will **not** add extended attributes
- **confidential** will **not** add extended attributes

For the extended attributes, we SHOULD use the already offered settings to determine „subject" and „location" information will be generated into the pxfb.

Types of Calendar folders

User folder Calendars

A user may have several personal folders in his account. One folder is called the default calendar folder. The default folder is used when accepting invitations within Microsoft Outlook. Kolab clients are encouraged to ask the user if an invitation should be stored in a different calendar folder.

The structure of user calendar folders is a folder hierarchy below the users top level folder and looks like

(1) `user/user1/calendar` fb, default
(2) `user/user1/private/calendar`
(3) `user/user1/anotherhierarchy/calendar` fb
(4) `user/user1/yetanotherhierarchy/calendar` fb

The numbering is for internal reference purposes of this paper. Below the folder names are indicators for their flags. „fb" means it is freebusy relevant. „default" means that the folder in question is the default folder.

Group Calendars

A group may have several calendars (often only one). Technically a group calendar is just a special user account with more than one user having access permissions. A group as opposed to a user shall be used to give folders to several users on an equal level. Each folder has an admin, which is the person which can give out rights to the folders.

```
(5) user/group1/calendar fb
(6) user/group1/myhierarchie/calendar
(7) user/group2/myhierarchie/calendar fb
(8) user/group2/otherhierarchy/calendar
```

Multi-Group Calendars

On a Kolab server there may be several "public folders" containing calendars. Technically those folder are similar to "anonymous" user accounts and a better name might be "multi-group folders".

Their intended use is to include a larger number of people, like all members a company.

Multi-group folders will not be relevant for freebusy lists.

The reason is that within larger groups of people almost never all
of them have to be at an event. E.g. it is better to work with invitations
or a more specific group folder when wanting half of the company to a meeting.

(9) shared.calendar
(10) shared.ahierarchie/calendar
(11) shared.anotherhierarchy/calendar
(12) shared.yetanotherhierarchy/deeper/calendar

Partial (x)fb

Whenever a client writes to a calendar it MUST trigger the server based generation of the freebusy and the extended freebusy information in the scope of the corresponding folder. We call these folder-specific freebusy information partial freebusy or short pfb. For extended freebusy we call them pxfb.

The user specific (x)fb is then generated transparently to the requesting application on demand from the user specific p(x)fb information. Specific details about how
to collect the partial information is specified below in the chapter "pxfb collector".

The following rules apply:

- We add to the Kolab Format Specifiction rules for annotations which allows to express the fact that this folder is freebusy relevant. See section 2.8 Groupware in [Architecture2006] which describes the „incidences-for" annotation.

- The same annotation shall also make that folder alarm relevant.

We define the following cases for annotations with regards to freebusy relevance:

- events in this folder are freebusy relevant for nobody
- events in this folder are freebusy relevant for administrators of this calendar folder (default)
- events in this folder are freebusy relevant for everyone with read access to this folder

E.g. in the notation and the example above we have fb flags
on the following folders:

```
user/user1/calendar (1, fb)
user/user1/anotherhierarchy/calendar (3, fb)
```

```
user/user1/yetanotherhierarchy/calendar (4, fb)
user/group1/calendar (1, fb)
user/group2/myhierarchie/calendar (3, fb)
```

If a client changes the contents of one of those calenders (e.g. adding an event or modifying some event) it must trigger the creation of the pfb and xpdf as described below.
The creation of the p(x)fb is executed with the credentials of the user that modified the calendar. This implies that this user has write permissions to the specific
folder.

Time spans (x)fb

The duration of a p(x)fb and (x)fb is determined by the sum of the account specific attribute kolabFreeBusyFuture which is stored in the LDAP directory and a globally configurable constant kolabFreeBusyPast. While the former is defined for the kolabInetOrgPerson LDAP objectclass the later is stored in the kolab objectclass. As all calculations are done on the server the resulting time is always calculated relative to the servers local timezone.

p(x)fb cache

The pfb and pxfb are written to the accounts pfb cache hierarchy.

FIXME: There are some .FIXME.acl files already.

Empty p(x)fb files are to be avoided.

```
user1/user1/calendar.pfb
user1/user1/calendar.pxfb
user1/user1/anotherhierarchy/calendar.pxfb
user1/user1/anotherhierarchy/calendar.pfb
user1/user1/yetanotherhierarchy/calendar.pfb
user1/user1/yetanotherhierarchy/calendar.pxfb
group1/group1/calendar.pfb
group1/group1/calendar.pxfb
group2/myhierarchie/calendar.pfb
group2/myhierarchie/calendar.pxfb
```

If a calendar folder is removed then the client must also trigger the generation of the corresponding p(x)fb file. In the case of a deleted or empty calendar the p(x)fb cache file is removed from the system.

The purpose of the p(x)fb cache is to hide the cost in generating the information from a latency point of view and in order to provide a clean security implementation as
the p(x)fb cache is generated using the credentials of a priviledged user with write permissions.

Actually the term pfb cache is a little bit misleading as it actually is an intermediate store which does not only cache the results of p(x)fb trigger processes but which implicitly also defines which p(x)fbs are relevant to create the user specific (x)fb.

The cache files with the .pxfb extension contain in addition to the information contained in the .pfb files the subject and the location attributes as extracted from the corresponding calendar folder. When generating the pxfb the script also adds the public/privat/confidential attribute to the pxfb cache file.

It is useful to generate the pfb and pxfb in the same run as technically the pfb files are a subset of the pxfb.

Access to the pxfb and xfb information is limited to authenticated users and subject to further access control while pfb and fb is either accessable for anonymous users or all authenticated users without further access control. The access rules for (p)fb is a server wide setting.

Access control to (p)xfb

xfb information is only available to authenticated users. Direct access to the .pxfb files is denied. The php script providing access to the (p)xfb has to follow the (p)xfb folder specific access controls.

The xfb access controls are set per folder and can affect either users or groups. The default is no access. All ACLs are positive and explicit. This means it is not possible to grant access to everyone in a group except a specific user. If such a szenario is required the ACLs must be explicitly granted to the users without the use of a group ACL.

Privacy settings are already provided for when generating the pxfb and they are therefor not relevant for ACLs.

The xfb ACLs are implemented as IMAP annotations using the the IMAP annotatemore extension.

In order to add extended attributes to the generated xfb we require explicit ACLs like

/vendor/kolab/pxfb-readable-for with the attribute **value.shared** set to **who**, a space character seperated list of groups and users who have read-access.

```
who={$user|$group} *{<SPACE> {$user|$group}}

Note that the size of the value is limited, but servers should
minimally accept 1024 bytes.
```

This ACL value is followed when generating the final xfb out of the pxfbs according to which user requests the xfb.

Format of (x)fb

The format of (x)fb follows the iCalendar standard. The fb must be explicitly compatible to the Outlook ifb format.
The xfb is identical to the fb but extended with the subject and location attributes

```
FREEBUSY;X-UID=bGlia2NhbC0xODk4MjgxNTcuMTAxMA==;X-
 SUMMARY=RW1wbG95ZWUgbWVldGluZw==;X-LOCATION=Um9vb
 SAyMTM=:20080131T170000Z/20080131T174500Z
```

Triggering p(x)fb

On the server we use mod_rewrite to parse URLs like.

https://servername/freebusy/trigger/user1@domain.tld/calendar.pfb
https://servername/freebusy/trigger/group1@domain.tld/calendar.pfb

We use SSL secured basic authentification for transfering the credentials to the server side code. The server side code then uses these credentials to access the calendar folders and to write the p(x)fbs. Due to the fact that triggers are only useful for users having write access to a calendar folder these access permissions are always sufficient.

In order to trigger creation of a pfb (and pxfb) a HTTP GET request is launched like:

https://servername/freebusy/trigger/X/PATH/FOLDERNAME.pfb

with X being one of
a) primary email address
Clients triggering for folders of other users could derive this from the corresponding imap path component = name and then adding "@maildomain" to it.

b) the uid
c) the corresponding imap patch component of the user name
(the server will try to add @maildomain to it.)
and PATH and FOLDERNAME being UTF-8(*) encoded IMAP foldernames.
d) any valid email alias

All pfbs are readable for every (authenticated) user though normal users don't require to read the pfbs. The p(x)fb collector must be able to read all p(x)fbs from potentially many servers (multi-location setups).

The pxfb is not directly available to users. When an xfb is requested the xfb script runs with the users credentials and makes use of another priviledged script which has full access to the pxfbs. It is the job of the priviledged script to implement the xfb ACLs. In case an xfb is requested by an anonymous user the fb is silently returned instead.

Creation of pxfbs happens simulataneously when pfbs are created.

In general all clients writing to calendar folders including the KDE Client Kontact, the Outlook Clients, the web client and the resource scripts must trigger the generation of pfbs.

p(x)fb collector

A script which is executed with the credentials of the requesting authenticated user collects the pfbs in order to create the real fb. This script is also able to run as the low priviledged Apache user (e.g. kolab-n). A server side setting defines if anonymous access to fb is granted.

A potentially different script is executed when a user specific xfb is requested. It uses the credentials as provided by the caller.
This xfb script collects the pxfb information for the specified user deploying another priviledged script. This priviledged script checks the provided credentials and provides a subset of the available pxfb information.

If the requesting and authenticated user lacks access permissions she is returned the plain fb information without any extended attributes.

If a freebusy list is requested the script can just simply have to recursively get all p(x)fb under the users fb cache hierarchy and then the ones in the group calendar folders which the user can read and which are marked fb relevant.

If an extended freebusy list is requested, the script has to use the

given credentials of the users to try to access the pxfbs in the same way.
If access to the pxfb is not possible, use the corresponding pfb.

The URLs for collecting freebusy info are

```
https://servername/freebusy/group1@domain.tld.ifb
https://servername/freebusy/user1@domain.tld.xfb
```

Deletion of Calendar folders

When deleting a calendar folder a Kolab client must first delete all appointments on the server and then immediately call the pfb script

```
https://servername/freebusy/trigger/X/PATH/FOLDERNAME.pfb
```

After calling the pfb generation script the Kolab client may immediately delete the emtpy folder.

On the other hand the server side pfb creation script MUST be able to handle empty or unexisting calendar folders. A trigger call to an unexisting or empty folder leads to deletion of any traces related to this folder in the pfb storage.

(*) It is the job of the p(x)fb creation script to convert the UTF-8 encoded paths and folder names into the imapd specific naming conventions.

FIXME: actually an empty, but existing calender should result in an fb showing "FREE", semantically. So why include the „empty" calender in here? Maybe because the script cannot find out if the calender is not there in contrast to having no access to it?

Future

Ideas for possible future enchancements:

- add referal type files for the cache hierarchy that make the collector script search for an pfb elsewhere. Advantage: Client could use other servers or local files for fb, too.
- add a deep parsing mode for the multi-group folders that creates pfbs by looking into all appointments and copies them to a special place in the cache hierarchy. Possible advantages: Easing the modelling of larger groups.
- add encryption for privacy/confidential purposes.
- Propose a standard how freebusy information can be found from an email address. Something like an MX record?

Needed implementations

Server-side

The code that generated the pxfb files needs to add the „location“ information to the pxfb, unless the event is „private“. The location is already added since

http://kolab.org/cgi-bin/viewcvs-kolab.cgi/server/kolab-resource-handlers/kolab-resource-handlers/freebusy/Attic/freebusy.class.php.in Rev 1.13 (2007-07-05), now module „kolab-freebusy“.

Also the ACL information for the pxfb need to be saved close to it after being read out from the IMAP annotation.

Access to the pxfbs must be checked against the ACL.

ACL maintenance

For each folder there must be a way for the responsible user to set the xfb ACLs annotations. An interface for this SHOULD be offered by the clients. As webclient and KDE client are Free Software this can be done for sure. For other clients it would be good for users to have a fallback system. The webadmin build into the Kolab Server should have a link to the subpage that does this. This should be usable even when the full Horde webclient is not installed and enabled on the server.

The server root can already use „cyradm“ to set the ACL as annotation. Webadmin Administrators which can change „multi-group folders“ should be able to also set the xfb ACLs for those folders (which contain calenders).

Display of xfbs

In order for the team leader to get the overview, the xfb information must be presented in a compact way for a set of groups.

The „/fbview“ must be extended to also show the location, both in the text and in the mouseover tooltip.

The KDE Kolab Client must be extended to use the xfb information in the chart when selecting a time for a specific appointment. In addition there should be a way to display the xfb/fb information for a set of users. Those users would need to be configured. Ideas: a) Use a local distribution list for configuration of a group of users do display. b) Add a specific list of users for which there are only xfb information to the horizontal calender view. c) as alternative to b) add a new view for xfb information.