Handbuch

4. Juli 2011

Inhaltsverzeichnis

1. Einführung

Die entwickelten Anwendungen sollen die Nutzer beim kollaborativen, projekt- und gruppenbezogenen Arbeiten über Unternehmensgrenzen hinweg unterstützen. Hierfür versetzt das Softwaresystem den Anwender in die Lage, Arbeitsabläufe im Unternehmen abzubilden und kollaborative Elemente, wie Versionskontroll- und Abstimmungssysteme, zu verwenden.

Auf Grundlage des Ansatzes von Buddycloud, einem verteilten sozialen Netzwerk auf der Basis von XMPP, und einer föderierten Infrastruktur von Servern, die ohne zeit- und kostenaufwändige zentrale Administration in den nutzenden Unternehmen direkt betrieben werden kann, können diese Funktionen umgesetzt werden. Im Kontext des bestehenden XMPP-Ökosystems sollen die Implementierungen unabhängig von bereits bestehender Buddycloud-spezifischer Infrastruktur sein.

1.1. Funktionsweise und -umfang

Das endgültige dezentrale soziale Netzwerk bietet den Nutzern die Möglichkeit kollaborativ an Projekten zu arbeiten, sodass diese mit geringem sozialen Aufwand bearbeitet werden können. So werden sie befähigt, das Projekt innerhalb des Projekt-Bereichs zu bearbeiten, sofern die entsprechenden Rechte gegeben sind. Diese Daten sind auf dem jeweiligen Host der Jid des Projekts-Administrators hinterlegt und werden auf dem Host aller Beteiligten abgebildet. Dieses Abrufen der Projektdaten via Versionskontrolle ermöglicht effizientes und paralleles Bearbeiten der Aufgaben der einzelnen Partizipanten. Ebenso können Abstimmungen unter den Projekt-Mitarbeitern eingesetzt werden, um zum Beispiel Termine festzulegen oder eventuell Arbeitsschritte zu koordinieren. Diese Arbeitsschritte werden in Workflows abgebildet, damit die Planung des Projekts eindeutig nachzuvollziehen ist.

1.2. Zielgruppe

Die speziellen Anforderungen mittelständischer Unternehmen stehen im Mittelpunkt der explorativen Software.

Dieses Handbuch wendet sich in Kapitel 3 direkt an den Anwender, während das Kapitel zur Installation (Kap. 2) umfangreiche Kenntnisse des Linux/Unix-Systems voraussetzt und somit eher für fortgeschrittene Nutzer oder Systemadministratoren gedacht ist.

1.3. Systemvoraussetzungen

1.3.1. Server

Auf dem Server wird sowohl eine Apache [CouchDB] dokumentenorientierte Datenbank, als auch ein -das in [CompP] definierte Komponentenprotokoll unterstützender- XMPP-Server (in unseren Beispielen [ejabberd]) benötigt. Zusätzlich ist die ECMAscript Laufzeitumgebung node.js in einer aktueller Version verlangt. Die einzelnen serverseitigen

Komponenten benutzen zusätzliche Bibliotheken, die in den jeweiligen Untersektionen beschrieben werden.

Um den webbasierten Client zu nutzen, muss sowohl ein HTTP-Server, als auch ein BOSH Connection Manager auf einem öffentlich zugänglichem System installiert sein. Dieses muss nicht notwendigerweise mit dem XMPP-Server identisch sein.

Auf dem Host sollte ein modernes Linux mit aktuellen Systembibliotheken installiert sein. Node.js ist nur unter der glibc bzw. eglibc funktionsfähig, die Header von OpenSSL müssen installiert sein.

1.3.2. Anwender

Der Webclient Ingemore ist plattformunabhängig auf allen gängigen Betriebssystemen lauffähig. Auf dem System des Endanwenders muss lediglich ein moderner Webbrowser mit HTML5 und Javascript Support installiert sein. Bisher wurde Ingemore mit den neusten Versionen von Firefox und Chromium getestet. Von der Benutzung des Internet Explorers wird aus Sicht der Standardkonformität explizit abgeraten.

2. Installation

Die Installationsanweisungen beschreiben nur eine mögliche Methode, um einen Knoten im Netzwerk aufzusetzen. Mit verschiedenen XMPP-Servern, BOSH-Konnektoren etc. ist eine Vielzahl von anderen Konfigurationen denkbar, die die Verfügbarkeit und Fehlertoleranz deutlich erhöhen können. Hier wird nur der einfachste Fall betrachtet: Sämtliche Software wird auf einem Server betrieben.

2.1. Server

2.1.1. ejabberd

Ejabberd ist ein in Erlang geschriebener hochgradig skalierender XMPP-Server. Aufgrund der komplexen Konfiguration und der fortgeschrittenen Architektur ist es empfehlenswert den Anweisungen genau zu folgen, da sonst Sicherheitsschwankungen auftreten können.

Ejabberd sollte über die Paketverwaltung der benutzten Linuxdistribution installiert werden. Falls dies nicht möglich ist oder andere Probleme auftreten, wird auf [EGuide] verwiesen.

Die Datei ejabberd.cfg muss nach der Installation noch auf das Hostsystem angepasst werden. Zuerst muss ein Hostname gewählt werden unter dem die einzelnen XMPP-Logins erzeugt werden sollen.

```
%% Hostname
{hosts, ["localhost"]}.
```

Ein ejabberd kann mehrere Domains verwalten, die dann jeweils mit Komma abgetrennt werden müssen. Falls gewünscht, kann die Registrierung von Accounts erlaubt werden. Hierfür muss

```
{access, register, [{deny, all}]}.
```

auf

```
{access, register, [{allow, all}]}.
```

geändert werden. Sämtliche weitere Konfiguration wird in den Sektionen zu den Komponenten abgehandelt.

2.1.2. node.js und npm

[node.js] ist eine ECMAscript Laufzeitumgebung, die speziell für die Entwicklung asynchroner Server entwickelt wurde. Um node.js zu installieren muss zuerst der Quellcode heruntergeladen, danach die letzte stabile Revision ausgecheckt und letztendlich diese kompiliert werden.

```
$ git clone git://github.com/joyent/node.git
$ git checkout $version
$ ./configure && make
# make install
```

Mittels

```
$ git tag
```

kann die letzte stabile Version ermittelt werden. Node.js benötigt *libev* und *OpenSSL* (inklusive Headern).

Die node.js Paketverwaltung [npm] lässt sich einfach mittels eines automatisierten Skriptes als root installieren.

```
# curl http://npmjs.org/install.sh | sh
```

2.1.3. BOSH-Connection Manager

Es gibt grundsätzlich zwei Möglichkeiten, BOSH (wodurch XMPP über HTTP erst möglich wird) serverseitig zu implementieren. Die erste Variante erfordert nur ejabberd, hat jedoch den Nachteil, dass nur mit XMPP-Entitäten kommuniziert werden kann, die von diesem Server verwaltet werden. Die zweite Variante benötigt einen sogenannten BOSH Connection Manager der unabhängig von bestehenden XMPP-Servern die BOSH-Verbindung auf natives XMPP umsetzt.

Als Connection Manager benutzen wir node-xmpp-bosh. Grundsätzlich gibt es jedoch eine Vielzahl anderer BOSH Proxies, welche mehr oder weniger die gleiche Funktionalität bieten.

Zuerst muss der Quellcode heruntergeladen und alle Abhängigkeiten installiert werden.

```
$ svn checkout http://node-xmpp-bosh.googlecode.com/svn/trunk/ node-xmpp-
    bosh
$ cd node-xmpp-bosh
$ npm install .
```

Die bosh.conf.example.js muss im Normalfall nicht weiter editiert werden. Ein Umbenennen in bosh.conf.js genügt völlig.

```
$ mv bosh.conf.example.js bosh.conf.js
```

Schlussendlich muss das Skript run-server.js ausgeführt werden, um den BOSH Connection Manager zu starten.

```
$ node ./run-server.js
```

2.1.4. CouchDB

CouchDB ist in allen modernen Linuxdistributionen vorhanden. Unter Debian kann sie einfach mit apt-get installiert werden.

```
# apt-get install couchdb
```

Weitere Konfiguration ist nicht nötig.

2.1.5. vcs-server

Der vcs-server wird direkt aus seinem Sourcecodeverzeichnis ausgeführt. Zuvor müssen jedoch alle Abhängigkeiten installiert werden. Dies erfolgt über den node.js Packetmanager npm. Durch das Ausführen von

```
$ npm install .
```

in der Wurzel des Quellcodeverzeichnisses werden alle Abhängigkeiten automatisch lokal (d.h. nur für den vcs-server) installiert (einige der Abhängigkeiten benötigen den icu-config Befehl). Eine genaue Auflistung ist in der Datei README zu finden. Aufgrund fehlender Funktionalität der libgit2 wird zusätzlich eine Installation von *git* im Pfad des Benutzers, unter dem die Komponente laufen soll, benötigt. Im Normalfall trifft dies zu, wenn git über das Paketverwaltungssystem der Linuxdistribution installiert wurde.

Nach dem Installieren der Abhängigkeiten muss eine CouchDB Datenbank für die Metadaten der einzelnen Repositories angelegt werden. Dies muss mit der REST-API, z.B. mittels curl, geschehen:

```
$ curl -X PUT http://127.0.0.1:5984/$dbname
```

Der Name der Datenbank kann frei gewählt werden.

Jetzt muss ein Verzeichnis gewählt werden, in dem die Repositories angelegt werden. Legen Sie dies ganz normal an und geben sie dem Nutzer der vcs-server Schreibrechte auf dem Verzeichnis.

Letztendlich muss der vcs-server konfiguriert werden und der ejabberd auf die Existenz der VCS-Komponente hingewiesen werden. Dafür muss in der ejabberd.cfg eine Listenkonfiguration für die Komponente angelegt werden:

```
{listen,[
...
{5234, ejabberd_service, [{hosts, ["vcs.localhost"], [{password, "secret"
    }]}]},
...
```

Der Port, Hostname und das Passwort können frei gewählt werden. Es ist wichtig, ein gutes Passwort (vgl. Kap. A.2) für die Verbindung zwischen Komponente und Server zu wählen. Der FQDN sollte eine Subdomain der XMPP-Domain sein, sonst ist die Wahl des Namens unbedeutend; Clients werden die Funktionalität der Komponente immer nutzen können. Es ist sinnvoll die maximale Stanzagrösse zu erhöhen, damit auch grössere Dateien abgerufen werden können.

```
{max_stanza_size, 99999999999}
```

Die Konfiguration muss im Quellcodeverzeichnis in der Datei *config.js* angesiedelt werden. Sie sollte wie folgt aussehen:

```
exports.xmpp = { jid: 'vcs.localhost',
                 password: 'secret',
                 host: 'localhost',
                 port: 5234
```

```
        };

/* git with couchdb metadata backend */
exports.modelBackend = 'git';
exports.modelConfig = {
        /* database config */
        host: 'localhost',
        port: 5984,
        database: 'vcs-server',
        /* git repository config */
        root: './git'
};
```

Hierbei sind die entsprechenden Felder an die bisher erzeugte Konfiguration anzupassen. Der erste Teil der Konfiguration beschäftigt sich mit der Verbindung zum XMPP-Server und sollte ohne Erklärung verständlich sein. Als Backend kann zur Zeit nur git benutzt werden. In der modelConfig muss das database Feld auf den Namen der erzeugten Datenbank gesetzt werden. root ist der Pfad, unter dem die Repositories angelegt werden.

Nach dem Befolgen der Schritte kann die Komponente einfach mit

```
$ node main.js
```

im Quellcodeverzeichnis gestartet werden. Im Fehlerfall wird sowohl beim Initialisieren als auch im normalen Betrieb ausführlicher Debug-Output erzeugt und angezeigt.

2.1.6. poll-server

Der poll-server wird analog zum vcs-server installiert und ausgeführt. Zuvor müssen jedoch alle Abhängigkeiten installiert werden. Dies erfolgt über den node.js Packetmanager npm. Durch das Ausführen von

```
$ npm install .
```

in der Wurzel des Quellcodeverzeichnisses werden alle Abhängigkeiten automatisch lokal (d.h. nur für den vcs-server) installiert. Eine genaue Auflistung der Pakete ist in der Datei README zu finden.

Nach dem Installieren der Abhängigkeiten muss eine CouchDB-Datenbank für die Daten der einzelnen Abstimmungen angelegt werden. Dies muss mit der REST-API, z.B. mittels curl, geschehen:

```
$ curl -X PUT http://127.0.0.1:5984/$dbname
```

Der Name der Datenbank kann frei gewählt werden.

Letztendlich muss der poll-server konfiguriert werden und der ejabberd auf die Existenz der Abstimmungskomponente hingewiesen werden. Dafür muss in der ejabberd.cfg eine Listenkonfiguration für die Komponente angelegt werden:

```
{listen,[
...
{5233, ejabberd_service, [{hosts, ["polls.localhost"], [{password, "secret"
    }]}]},
```

```
...
```

Der Port, Hostname und das Passwort können frei gewählt werden. Auch hier ist es wichtig ein gutes Passwort für die Verbindung zwischen Komponente und Server zu wählen. Der FQDN sollte eine Subdomain der XMPP-Domain sein, sonst ist die Wahl des Namens unbedeutend; Clients werden die Funktionalität der Komponente immer nutzen können.

Die Konfiguration muss im Quellcodeverzeichnis in der Datei *config.js* angesiedelt werden. Sie sollte wie folgt aussehen:

```
exports.xmpp = { jid: 'polls.localhost',
                 password: 'secret',
                 host: 'localhost',
                 port: 5233
               };

exports.modelBackend = 'couchdb';
exports.modelConfig = {
    host: 'localhost',
    port: 5984,
    database: 'poll-server'
};
```

Hierbei sind die entsprechenden Felder an die bisher erzeugte Konfiguration anzupassen. Der erste Teil der Konfiguration beschäftigt sich mit der Verbindung zum XMPP-Server und sollte ohne Erklärung verständlich sein. Im letzten Teil der Konfiguration wird das Datenbankbackend konfiguriert. Als Backend kann zur Zeit nur CouchDB benutzt werden. In der modelConfig muss das database-Feld auf den Namen der erzeugten Datenbank gesetzt werden.

Nach dem Folgen der Schritte kann die Komponente einfach mit

```
$ node main.js
```

im Quellcodeverzeichnis gestartet werden. Im Fehlerfall wird sowohl beim Initialisierenals auch im normalen Betrieb- ausführlicher Debug-Output erzeugt.

2.1.7. Ingemore

Ingemore besteht aus einer Javascript-Applikation, die im Browser des Nutzers ausgeführt wird und einen vollständigen XMPP-Client implementiert. Das heißt, es findet keine wirkliche Installation statt. Ein Nutzer könnte einfach die lokalen Dateien öffnen und dem Netzwerk beitreten. Im Normalfall wird auf dem Server ein HTTP-Server laufen, der die Dateien, aus denen der Client besteht, öffentlich verfügbar macht. Das einzige variable Element besteht in der Konfiguration des zu nutzenden BOSH-Proxies.

Hierfür muss die Datei *config.js* einfach um den gerade installierten BOSH-Proxy ergänzt werden.

2.2. Anwender

2.2.1. Ingemore

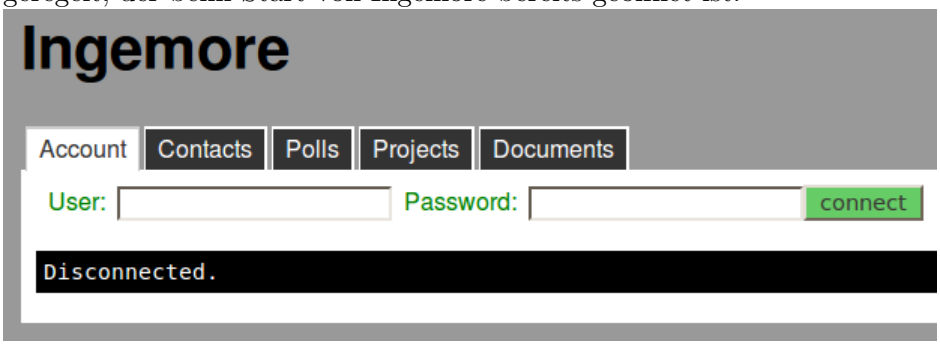
Der Webclient Ingemore besteht aus einigen HTML- und Javascript-Dateien und wird lokal auf dem Rechner betrieben. Eine Installation ist hierbei nicht erforderlich.
Um den Webclient zu starten, müssen die notwendigen Dateien ggf. entpackt werden. Anschließend wird die im Unterordner *view* befindliche Datei *index.html* im Browser geöffnet.

3. Bedienung: Ingemore

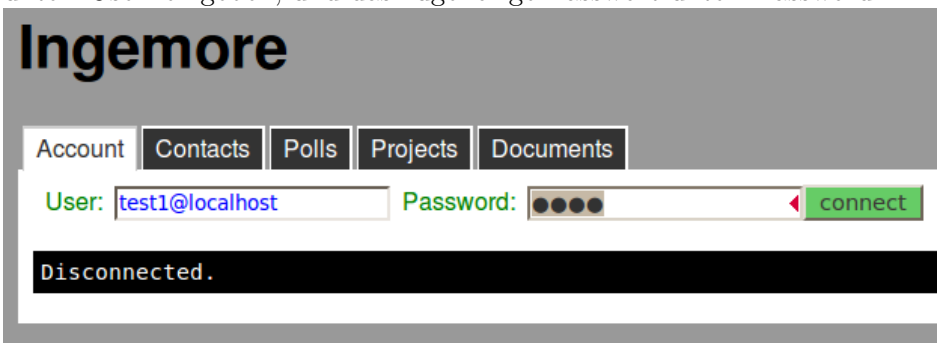
Es wird davon ausgegangen, dass der Betreiber des Jabber-Servers bereits dafür gesorgt hat, dass Benutzerkonten angelegt und mit Zugangsdaten ausgestattet sind. Das Registrieren von Benutzern wird dem Betreiber überlassen, der hierfür selbstständig eine Lösung finden muss.

3.1. An-/Abmeldung an Ingemore

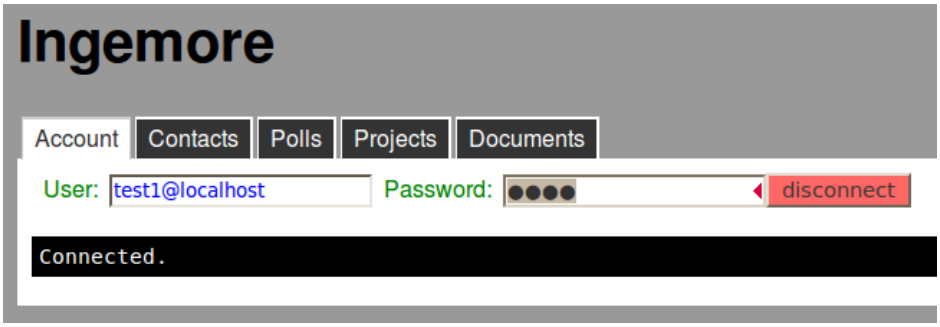
Wenn Ingemore aufgerufen wird, ist die Anmeldung der erste zu erledigende Schritt. Ohne Anmeldung am benutzerspezifischen Dienst ist es nicht möglich, andere Funktionen von Ingemore zu nutzen. Die An- und Abmeldung wird über den Reiter 'Account' geregelt, der beim Start von Ingemore bereits geöffnet ist.



Um das Anmelden am System durchzuführen muss der Benutzer seine JabberID (Jid) unter 'User' eingeben, und das zugehörige Passwort unter 'Password':



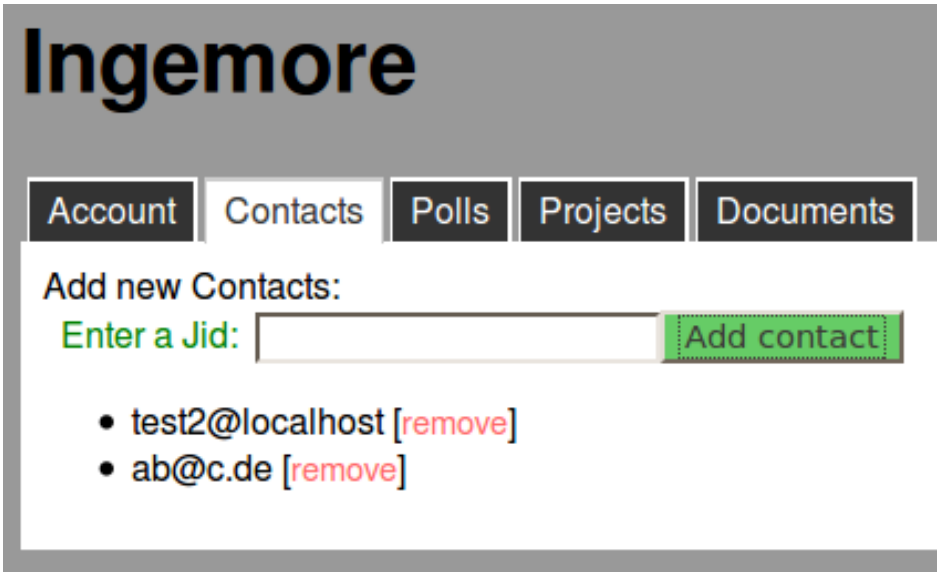
Die Anmeldung selbst wird über den Button 'connect' vorgenommen, der angezeigte Status 'Disconnected.' wechselt bei erfolgreicher Anmeldung zu 'Connected.' und an Stelle des Buttons 'connect' steht der Button 'disconnect', mit dem sich ein Benutzer vom System abmelden kann.



3.2. Die Kontaktliste

Unter dem Reiter ‘Contacts’ findet sich die Kontaktliste eines Benutzers.

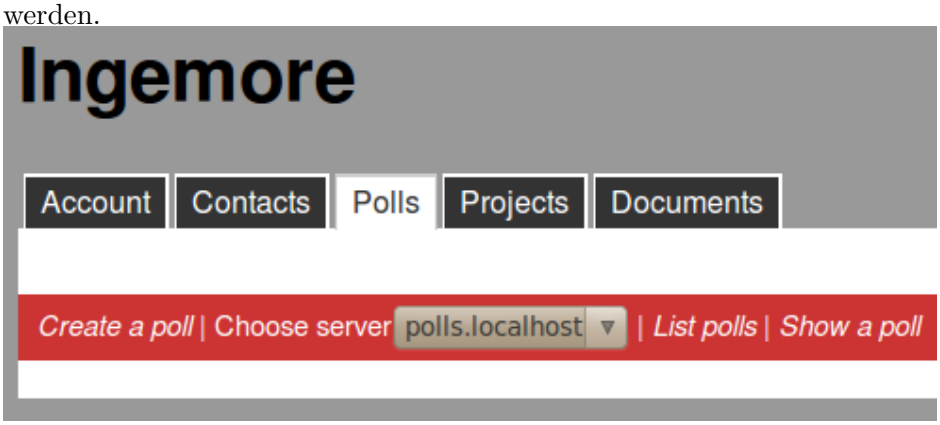
Hier gibt es die Möglichkeit, bestehende Kontakte (test2@localhost im Beispiel) anhand ihrer Jid einzusehen, neue Kontakte über ihre Jid hinzuzufügen, oder bestehende Kontakte aus der Kontaktliste zu entfernen. Um die Ressourcen eines anderen Kontakts mitbenutzen zu können, muss dieser zur Kontaktliste hinzugefügt werden, sodass Ingemore in die Lage versetzt wird, dessen Server im Netzwerk zu finden. Außerdem können mit Kontakten in der Kontaktliste leicht gemeinsame Projekte erstellt werden, was sonst nicht möglich ist.



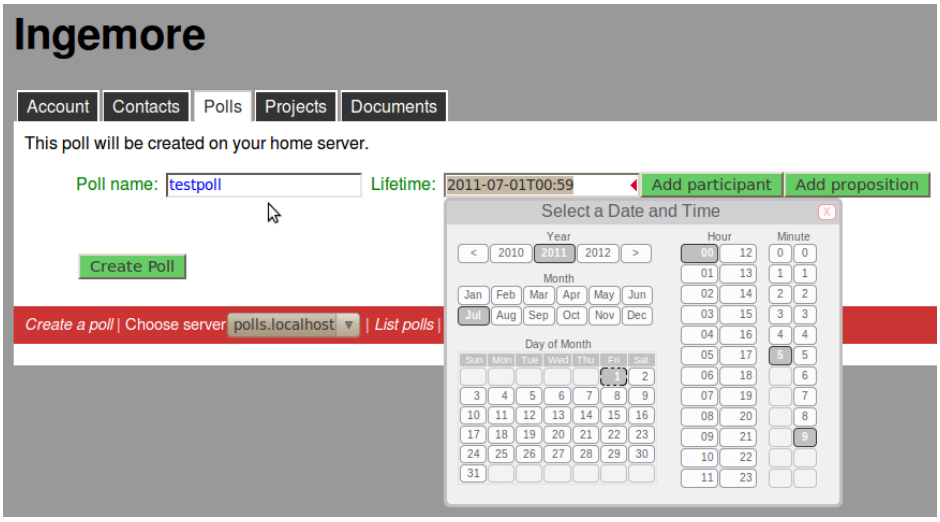
Nach dem Einfügen eines Kontaktes in die Kontaktliste ist dieser sofort einsehbar, und kann auch wieder entfernt werden.

3.3. Polls - Abstimmungen im Netzwerk

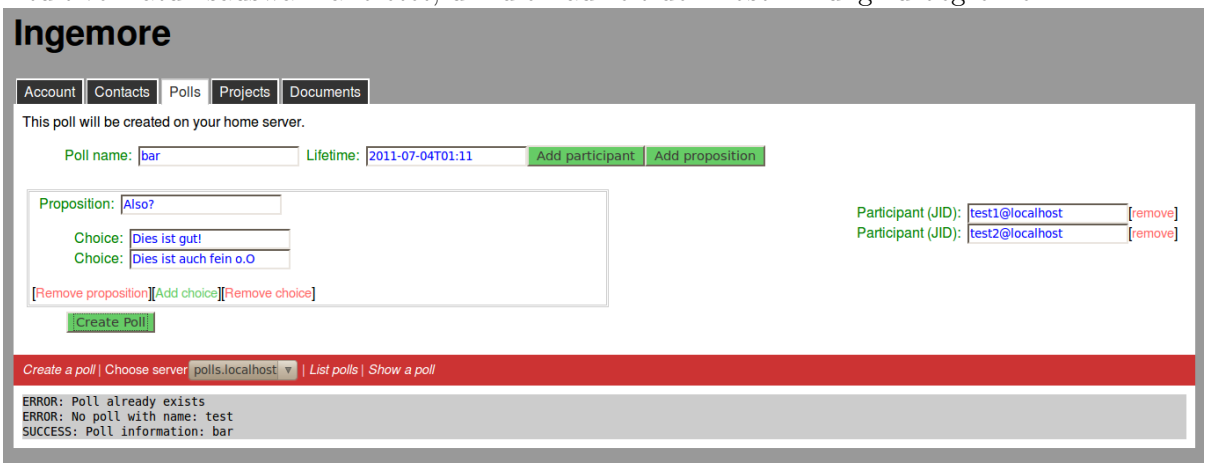
Unter dem Reiter 'Polls' können gemeinsame Abstimmungen im Netzwerk durchgeführt werden.



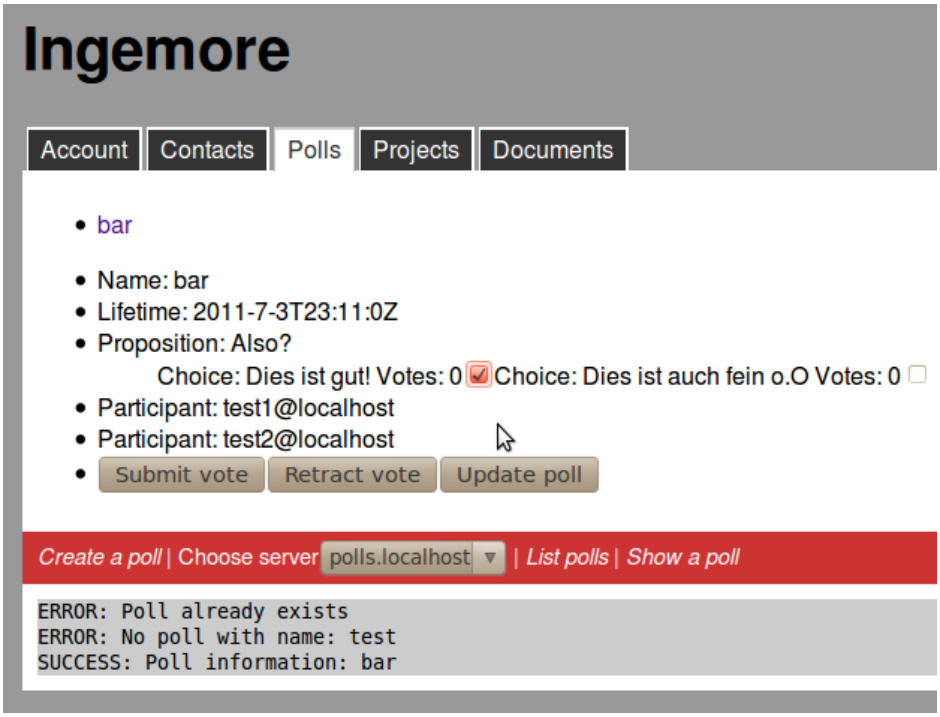
Dazu ist es möglich, mit 'Create a poll' neue Abstimmungen zu erstellen, bestende Abstimmungen mit 'List polls' abzurufen und eine bestimmte Abstimmung mit 'Show a poll' anzuzeigen. Dies kann für unterschiedliche Server vorgenommen werden, wobei unter 'Choose Server' der gewollte Server aus einer Liste von von Ingemore selbstständig entdeckten Servern ausgewählt werden muss.



Beim Erstellen einer neuen Abstimmung unter ‘Create a poll’ sind zuerst bei ‘Poll name:’ ein Name für die neue Abstimmung und eine Laufzeit unter ‘Lifetime:’ anzugeben. Für die Laufzeit der Abstimmung öffnet sich dazu ein eigenes Dialogfenster, dass eine intuitive Datumsauswahl anbietet, um die Laufzeit der Abstimmung zu begrenzen.



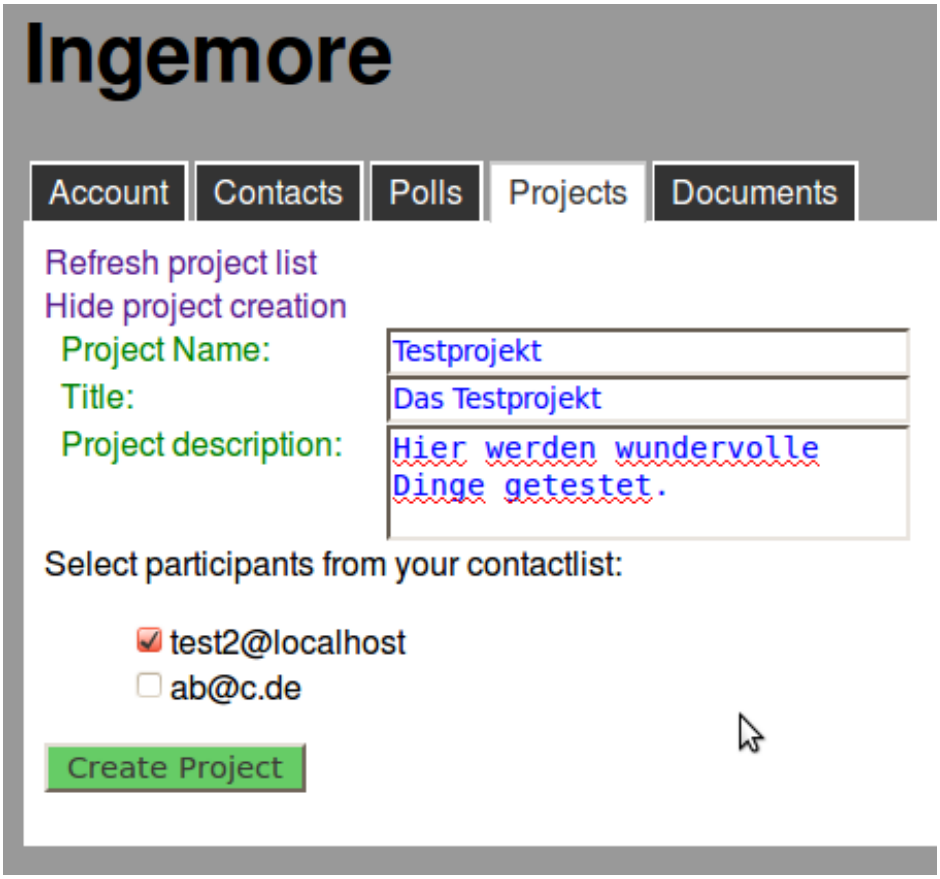
Danach können mit ‘Add participant’ mehrere Teilnehmer an der Abstimmung angegeben werden. Hier ist es im Allgemeinen sinnvoll, die eigene Jid mit anzugeben, wenn man selbst auch informationen zur Abstimmung erhalten möchte, und auch in der Lage sein will, eine Stimme abzugeben. Desweiteren können Punkte, über die abgestimmt werden soll, mit ‘Add proposition’ hinzugefügt werden. Ein solcher Abstimmungspunkt wird neben der bei ‘Proposition:’ angegebenen Frage durch mehrere Entscheidungsmöglichkeiten ‘Choice’ charakterisiert. Mit dem Button ‘Create Poll’ wird die Abstimmung dann auf dem ausgewählten Server erstellt, und alle Teilnehmer werden informiert.



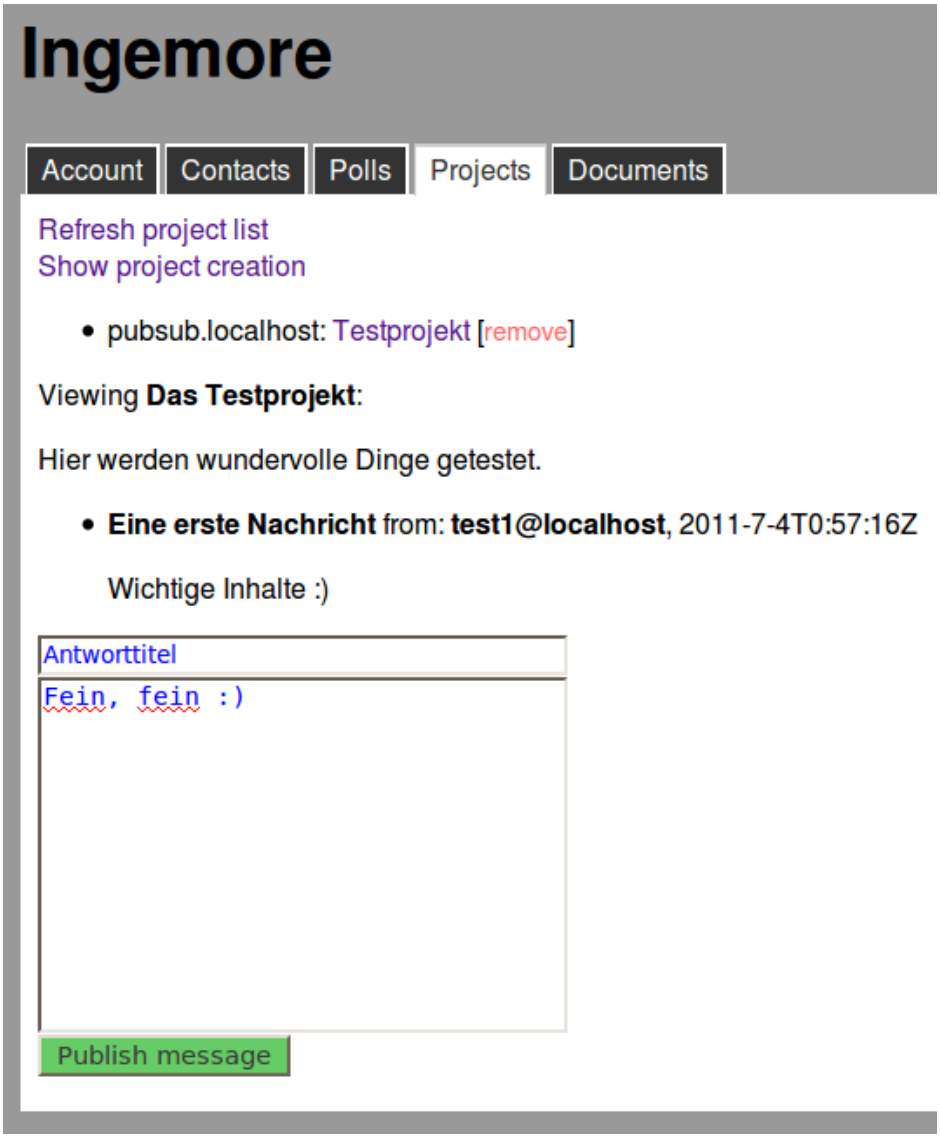
Unter ‘List polls’ kann ein Benutzer bestehende Abstimmungen ansehen, und durch Mausklick auf deren Namen die Details dazu einsehen. Diese Detailansicht ermöglicht auch das Abstimmen für die mit einem Häkchen versehenen Entscheidungsmöglickeiten (’Choices’) über den Button ‘Submit vote’. Eine bereits abgegebene Stimme kann über ‘Retract vote’ zurückgezogen werden, sodass ein Benutzer sich auch nachträglich umentscheiden kann. Die Funktionen ‘Submit vote’ und ‘Retract vote’ sind nur möglich, solange eine Abstimmung noch innerhalb ihrer Laufzeit ist. Über ‘Update poll’ kann ein Benutzer die aktuellen Ergebnisse einer Abstimmung abfragen, sodass der Verlauf einer gemeinsamen Entscheidungsfindung ersichtlich wird. Unter ‘Show a poll’ kann ein Benutzer gezielt auf die Informationen zu einer Abstimmung zugreifen, mit denen dann analog zu ‘List polls’ umgegangen wird. Dazu wird allerdings der Name einer Abstimmung benötigt, den der Benutzer selbst angeben muss.

3.4. Projekts - Projektbezogene Kommunikationsräume

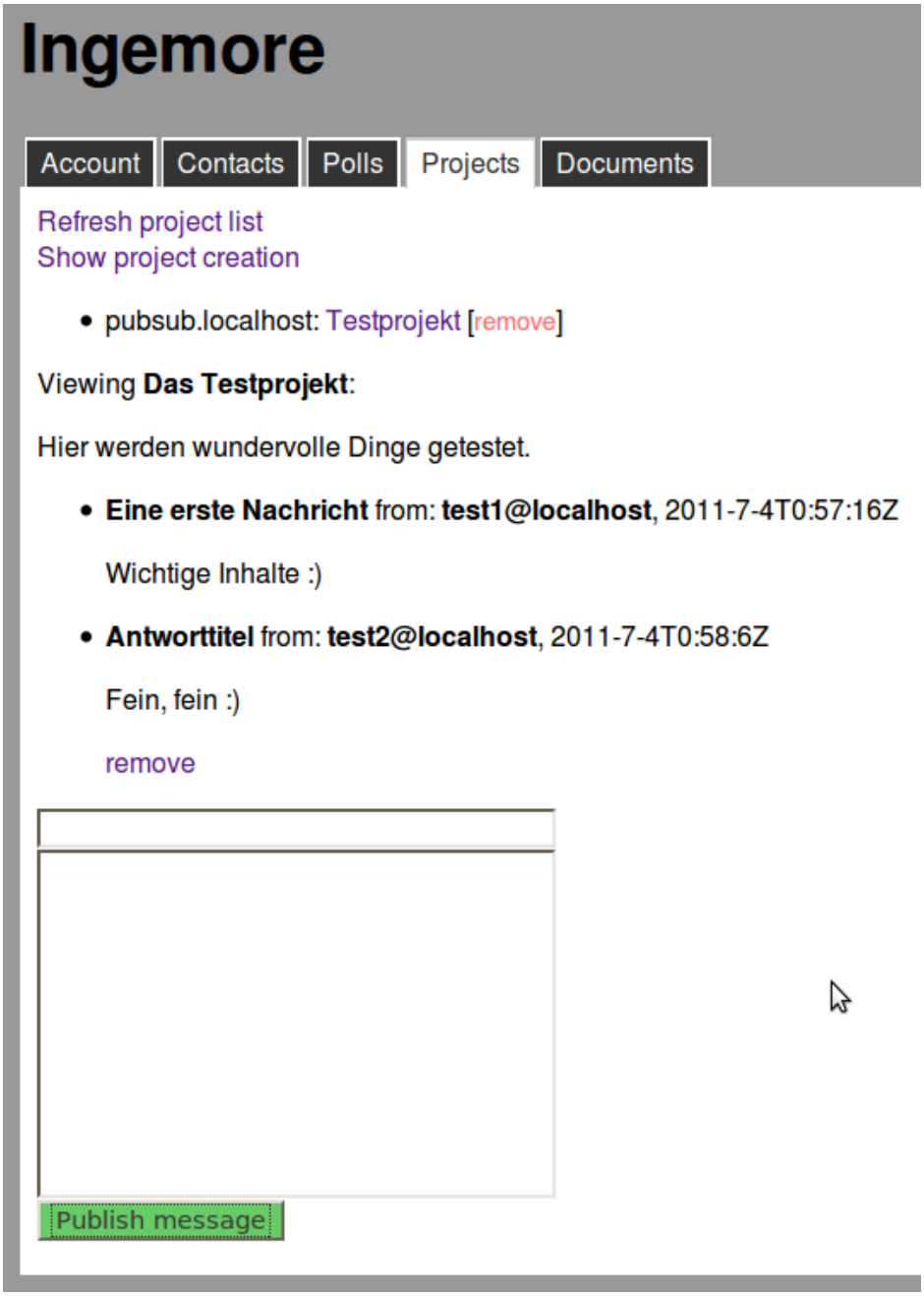
Unter dem Reiter ‘Projects’ lassen sich gemeinsame Projekte verwalten. Bestehende Projekte werden in einer Liste angezeigt, die von Ingemore selbstständig gefüllt wird, aber auch vom Benutzer manuell über ‘Refresh project list’ aktualisiert werden kann. Mit ‘Show project creation’ wird der Benutzer in die Lage versetzt, selbst Projekte zu erstellen.



Dazu muss bei ‘Project Name:’ ein Projektname angegeben werden, unter dem das Projekt in der Projektliste auffindbar ist. Desweiteren können bei ‘Title:’ ein Projekttitel und bei ‘Project description:’ eine Projektbeschreibung angegeben werden, die mit dem projektinternen Kommunikationsverlauf angezeigt werden können. Nun sollten unter ‘Select participants from your contactlist:’ die Jids ausgewählt werden, die am Projekt teilnehmen können sollen. Schlussendlich wird mit ‘Create Project’ das Projekt erstellt. Der projekterstellende Benutzer ist dabei automatisch Teilnehmer des Projektes. Alle Teilnehmer eines Projektes werden beim Erstellen eines neuen Projektes automatisch von diesem benachrichtigt, und sind absofort in der Lage, Nachrichten auszutauschen. Der Dialog zum Erstellen von Projekten kann über ‘Hide project creation’ wieder ausgeblendet werden.

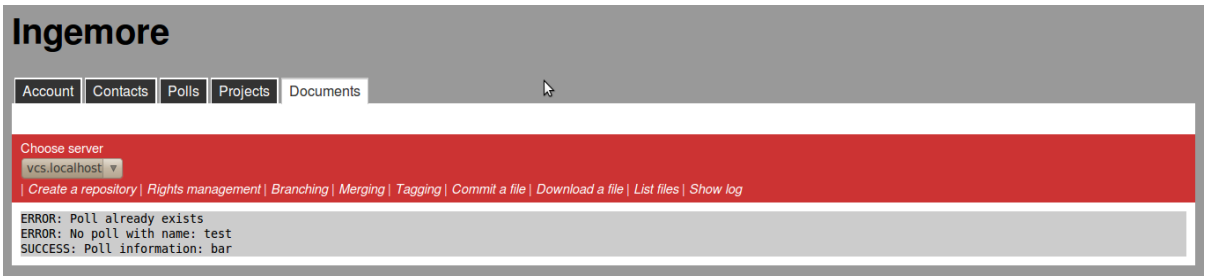


Bekannte Projekte werden, inklusive des Servers auf dem sie erstellt wurden, aufgelistet. Bei Klick auf den Projektnamen ('Testprojekt' im Beispielbild) wird der Projektinhalt unter 'Viewing' angezeigt. Darauf folgt eine Liste von innerhalb des Projektes ausgetauschten Nachrichten, die neben Titel und Inhalt auch Informationen über den Autor und ihr Erstellungsdatum enthalten. In den darunter folgenden Eingabefeldern kann eine Nachricht für das Projekt verfasst werden, die mit 'Publish message' versendet wird.



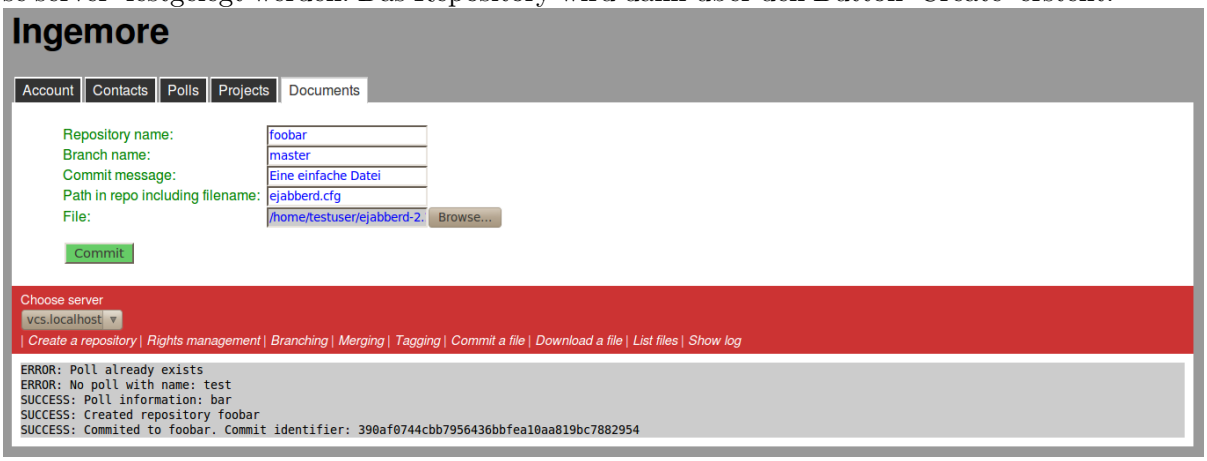
Nachrichten erscheinen nach dem Versenden in chat-ähnlichen Geschwindigkeiten bei allen Teilnehmern, die das betreffende Projekt zur Zeit beobachten. Sie sind aber auch für die gesamte Projektlaufzeit immer wieder abrufbar, solange der jeweilige Autor sie nicht über den mit ‘remove’ betitelten Link löscht.

3.5. Documents - Gemeinsame, versionierte Dateiverwaltung



Unter dem Reiter ‘Documents’ können Dokumente gemeinsam verwaltet werden. Hierfür werden beim Benutzer grundlegende Kenntnisse des Versionskontrollsystems Git vorausgesetzt.

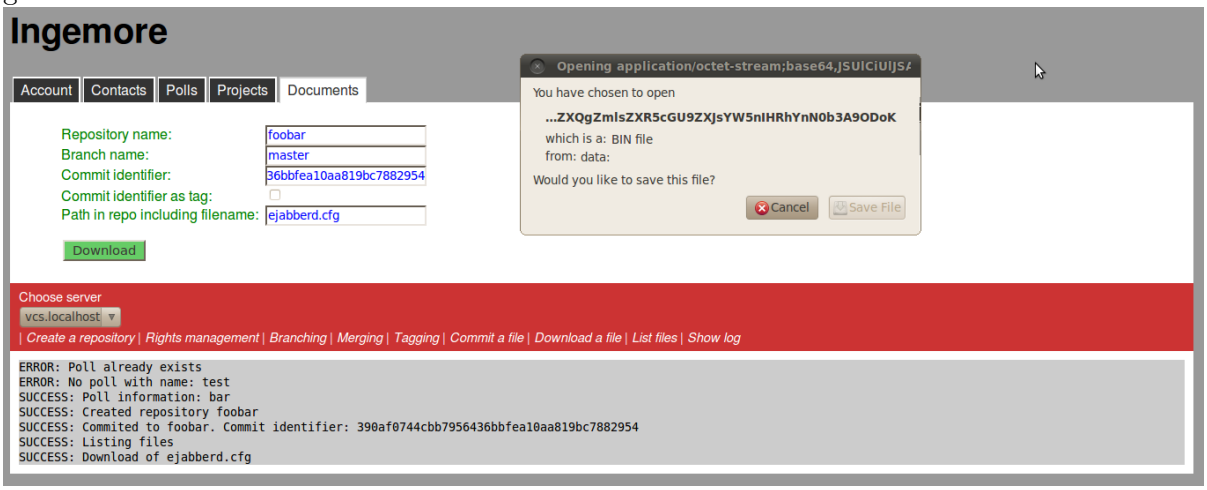
Unter ‘Create a repository’ kann ein Benutzer ein neues Repository zur Dateiverwaltung anlegen. Dafür muss ein entsprechender Name für das neue Repository angegeben, das Zugriffsmodell bei ‘Access model:’ ausgewählt, und der entsprechende Server bei ‘Choose server’ festgelegt werden. Das Repository wird dann über den Button ‘Create’ erstellt.



Mit ‘Commit a file’ kann ein Benutzer eine Datei in ein Repository hochladen. Dazu muss der Name des gewünschten Repositorys angegeben werden, der entsprechende Branch (Voreinstellung ist ‘master’), eine ‘Commit message:’ die die genannte Datei beschreibt, sowie ein Dateiname für die Datei im Repository. Mit ‘Browse’ kann der Benutzer dann eine Datei zum hochladen auswählen, und diese über ‘Commit’ zum Server senden. Unten in der Nachrichtenleiste erscheint daraufhin bei Erfolg des Commits eine Notiz, die den ‘Commit Identifier’ des eben getätigten Commits anzeigt. Dieser wird benötigt, um

die Datei in der gewünschten Version abzurufen.

Unter ‘List files’ kann sich ein Benutzer unter Angabe der Daten Repository, Branch und Commit Identifier die Datein zu einem Commit in einem Branch des Repositorys anzeigen lassen.



Unter ‘Download a file’ kann ein Benutzer dann Datein aus einem bestehendem Repository herunterladen. Dazu werden der Name des Repositorys, des Branches, der Commit Identifier sowie der Dateipfad im Repository benötigt. Nach einem Klick auf ‘Download’ versucht der Browser ein Popup zu öffnen, um dem Benutzer zu erlauben, die Datei an einer von ihm gewählten Stelle zu speichern.

A. Sicherheitshinweise

A.1. ejabberd

Ejabberd ist für einen verteilten Betrieb konzipiert. Deshalb wird per default der epmd -ein Daemon, der für die Synchronisation zwischen Erlang nodes zuständig ist- auf allen Interfaces gestartet. Zur Zeit gibt es keine Möglichkeit dies über eine Kommandozeilenoption abzustellen. Der Port 4369 sollte deswegen von einer Firewall nach aussen hin geblockt werden, da sonst beliebiger Code auf dem Server ausgeführt werden kann.

A.2. sichere Passwörter

Sichere Passwörter haben mindestens 8 Zeichen. Sie sind aus einer möglichst zufälligen Kombination von Groß- und Kleinbuchstaben, Zahlen und Sonderzeichen zusammengesetzt und enthalten nach Möglichkeit keine vollständigen, lexikographisch sinnvollen Worte.
Sie sollten schwer zu erraten sein, um die vertrauliche Information, zu der sie Zugang bieten, zu schützen. Überdies empfiehlt es sich, diese Passwörter regelmäßig zu ändern.

B. Glossar

Anticloud

Eine cloudähnliche Anwendung, die so realisiert ist, dass Daten nicht zentral gesammelt werden, sondern es keinen zentralen Server gibt und die Nutzer möglichst viel Kontrolle über ihre Daten behalten. Im Gegensatz zur klassischen Cloudarchitektur ist zu jedem Zeitpunkt klar, auf welchen Knoten im Netz sich die Daten eines Nutzers befinden bzw. wohin sie repliziert werden dürfen.

BOSH

Bidirectional-streams Over Synchronous HTTP ist ein Transportschichtprotokoll, dass das Verhalten von langlebigen, bidirektionalen TCP-Verbindungen zwischen Client und Server über das HTTP emuliert. Hierfür werden mehrere synchrone HTTP request/response Paare genutzt, wodurch ineffizientes Polling vermieden wird.

BOSH sollte unter anderem vom Buddycloud Webclient genutzt werden, um einen vollständigen →*XMPP*-Client im Browser zu implementieren.

Components

Components stellen eine standardisierte (→*XEP*-0114) Schnittstelle zwischen →*XMPP*-Servern und Anwendungen, die zusätzliche Funktionalität implementieren, dar. Alle weit verbreiteten XMPP Server unterstützen diese Technologie. Sie erlaubt eine dynamischere Entwicklung von Protokollerweiterungen als im Hauptzweig des Servers mit seinen gewöhnlich ungleich längeren Entwicklungszyklen.

Föderation

Das Zusammenschließen einzelnder Serverinstanzen, um deren lokale Daten und Dienste gemeinsam nutzen zu können. Im Gegensatz zu Peer-to-Peer-Netzwerken kommunizieren Nutzer nicht direkt miteinander, sondern nutzen föderierte (verteilte) Server. Das wahrscheinlich bekannteste Beispiel einer solchen Infrastruktur ist Email.

Gruppe

Ein Zusammenschluss von Akteuren, in unserem Kontext Firmen, um bezüglich eines Projekts einen *kooperativen Kommunikationsraum* zu bilden.

Ingemore

Der letztendlich eigens entwickelte WebClient zur entstandenen Software heißt Ïngemore"; ein Name, der auf dem scherzhaften Ursprungs-Prototyp-WecClient Ïngeborg"basiert. Von diesem ausgehend wurde Ingemore entwickelt. Ingemore stellt nun die graphische und funktionale Schnittstelle zwischen Nutzer mittels Browser zum Netzwerk her.

Jid

Auch: JabberID, bezeichnet eine eindeutige Kennung eines Akteurs innerhalb des →*XMPP*-Netzwerks. Eine Jid hat die Form *Name@Server* und ist vergleichbar mit einer eMail-Adresse.

Kollaborative Elemente

Technisch realisierte Funktionen, die die Zusammenarbeit unterstützen. Beispiele hierfür sind Abstimmungs- und →*Versionskontrollsysteme*.

Kooperative Komunikationsräume

Zusammenhänge, in denen Akteure zum gemeinsamen Profit kommunizieren.

Notarfunktionalität

Die Möglichkeit, Artefakte, ggf. unter der Aufsicht Dritter, technisch zu bestätigen bzw. zu unterschreiben, sodass diese Bestätigung insbesondere von anderen zur Kenntnis genommen werden kann, rechtlich verbindlich und manipulationssicher ist.

Notarfunktionalität wird oft durch kryptographische Methoden realisiert. Hierbei kann entweder ein hierarischer Ansatz, in der Praxis am besten durch SSL-Zertifikate bekannt, oder ein verteiltes System mit einem Netz des Vertrauens, wie in OpenPGP erfolgen.

Projekt

Der Prozess des zielorientierten Zusammenarbeitens in einer Gruppe bezeichnen wir als Projekt, welches auch in der Bezeichnung des namensgleichen Reiters im WebClient Ingemore wiederspiegelt.

Technosoziales System

Ein System, das sowohl aus technischen als auch aus sozialen Elementen zusammengesetzt ist.

Verteiltes Versionskontrollsystem

Ein Versionskontrollsystem ist ein System, das zur Erfassung von Änderungen an Dokumenten oder Dateien verwendet wird. Alle Versionen dieser Dateien werden in einem Repository mit Zeitstempel gespeichert und können zu einem späteren Zeitpunkt wiederhergestellt werden. Typischerweise werden Versionskontrollsysteme in der Softwareentwicklung eingesetzt, um Quelltexte zu verwalten. Mit Einschränkungen können sie jedoch auch zur Verwaltung beliebiger Dokumente genutzt werden.

Verteilte Versionskontrollsysteme speichern das Repository nicht (nur) auf einem zentralen Server; Nutzer erzeugen eine lokale Kopie des gesamten Repositories und nehmen lokal Änderungen vor, die nicht unbedingt wieder auf die Quelle zurückrepliziert werden.

Workflow

Der Überbegriff Ärbeitsablaufist eine vordefinierte Abfolge von Aktivitäten in einer Organisation. Ein →*Arbeitsfluss* (englisch: workflow) ist eine inhaltlich abgeschlossene, zeitlich und sachlogisch zusammenhängende Folge von Funktionen, die zur Bearbeitung eines betriebswirtschaftlich relevanten Objektes notwendig sind und deren Funktionsübergänge von einem Informationssystem gesteuert werden.

XMPP

Das *Extensible Messaging and Presence Protocol* ist ein offenes Instant Messaging Netzwerkprotokoll, das aber aufgrund seiner Erweiterbarkeit mittlerweile auch für VoIP und andere Anwendungen genutzt wird. Mehrere verteilte soziale Netzwerke nutzen XMPP als Grundlage, da es bereits Authentifizierung, Verschlüsselung und eine →*föderierte* Serverarchitektur bereitstellt.

XEP

XMPP Extension Protocol sind standardisierte Erweiterungen des →*XMPP*. Sie werden in einem ähnlichem Verfahren wie *Request for Comments* vor einem Konsortium involvierter Ingenieure eingebracht und verabschiedet.

Literatur

[ejabberd] Website: http://www.ejabberd.im

[EGuide] http://www.process-one.net/docs/ejabberd/guide_en.html (accessed June 2011)

[node.js] Website: http://node.js.org

[npm] Website: http://npmjs.org

[CouchDB] Website: http://couchdb.apache.org

[CompP] Peter Saint-Andre, XEP-0114: Jabber Component Protocol, http://xmpp.org/extensions/xep-0114.html, 2011