Entwurfsbeschreibung

Allgemeines

Evolution ist ein freier Personal Information Manager (PIM), der unter anderem E-Mail-, Adressbuch- und Kalenderfunktionen beinhaltet, läuft unter GNU/Linux, Mac OS X, Windows und anderen Unix-ähnlichen Betriebssystemen und ist Bestandteil des GNOME Projekts. Da Evolution erweiterbar ist und Datenquellen von der Benutzeroberläche trennt, wurde es zur Implementierung des Prototyps gewählt.

Produktübersicht

Der zu entwickelnde Prototyp sollte eine zeitlich durch eine Abstimmungsfrist begrenzte Terminabstimmung und nachfolgende Terminankündigung ermöglichen. Dieser Anwendungsfall wird vollständig durch iCalendar (RFC 5545[1]) und iTIP (RFC 5546[2]), auf denen Evolutions Kalender aufsetzt[3], abgedeckt.

Evolution implementiert mit dem itip-formatter Plugin auch die iMIP[4] Spezifikation und erlaubt dadurch unter anderem die Abstimmung und Veröffentlichung von iCalendar Daten, was für die Umsetzung des Prototypen den Vorteil bietet, dass E-Mail als Transportschicht schon existiert und funktioniert und, da iTIP unabhängig von der Transportschicht ist, später die Transportschicht ersetzt werden kann.

Da Evolution die iCal und iTIP Spezifikation nicht vollständig in der Benutzeroberfläche umgesetzt wird, mussten Anpassungen und Erweiterungen am bestehenden Programm vorgenommen werden. Dies geschah in C unter Verwendung von GLib und GTK+. Die Änderungen sind:

- Einfügen eines Veröffentlichungsbuttons, der eine Zusammenfassung abschickt, um nach Ablauf der Frist (manuell) alle Teilnehmer über den Ausgang der Terminabstimmung zu informieren[5].
- Mitgeschickte Alarme, die zum Setzen der Abstimmungsfrist und der Erinnerung dienen, werden übersichtlich in der benachrichtigenden E-Mail aufgelistet.

[1] `https://tools.ietf.org/html/rfc5545`

[2] `https://tools.ietf.org/html/rfc5545`

[3] Die Implementierung weicht zur Zeit teilweise von der Spezifikation ab.

[4] `https://tools.ietf.org/html/rfc6047`

[5] Die derzeitige Implementierung ist nicht vollständig konform zur iTIP Spezifikation, da Evolution die Sequenzierung von iTIP Nachrichten nicht wirklich durchführt. Perspektivisch ist jedoch eine Überarbeitung der iCalendar bezogenen Teile des Quelltexts vorgesehen, um die Spezifikationskonformität zu erhöhen. Die derzeitige Implementierung sollte daher eher als eine Art Konzeptstudie gesehen werden.

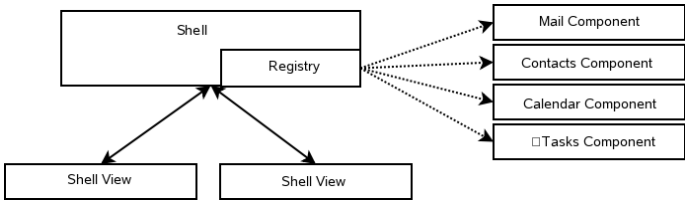

Abbildung 1: UML-Diagramm für Evolution[6]

Struktur des Gesamtsystems

Evolution ist in zwei Teile geteilt: Evolution, die Benutzeroberfläche und den Evolution Data Server, die per IPC über den Nachrichtenbus DBus Daten austauschen. Der Evolution Data Server speichert dabei E-Mails, Kontakt- und Termindaten und verarbeitet Suchanfragen auf diesen Daten. Kontakt- und Termindaten werden im vCard und iCalendar Format sowohl übertragen als auch verarbeitet.

Der Evolution Data Server kann durch Backends erweitert werden, die jeweils Funktionen wie E-Mail-Speicher und -Transport, Kalender- oder Kontaktspeicher übernehmen[7]. Der Evolution Data Server verarbeitet dabei im Wesentlichen die Anfrage der Benutzeroberfläche (Serialisierung, Deserialisierung, Normalisierung, Konvertierung, Hilfsfunktionen) und leitet die Anfrage an das jeweilige Backend weiter. Die Nutzung von standardisierten Datenmodellen[8] ist für die Anzahl und Uniformität der Backends und der Benutzeroberfläche wichtig.

Evolution greift dann über an dem Entwurfsmuster des Stellvertreters (Proxy) orientierte Objekte über die Benutzeroberfläche transparent und zu einem gewissen Grad nebenläufig auf die entsprechenden Objekte des Evolution Data Servers und letztendlich dessen Backends zu[9].

Evolution selbst besteht aus lose verbundenen Benutzeroberflächenkomponenten: E-Mail, Kalender, Memos und Aufgaben. Komponenten und Plugins können dabei eigene Menüs, Menüeinträge, Seitenleisten und Hauptfenster einbinden, so dass sich Evolution (sofern sinnvoll) um beliege Funktionen erweitern lässt. Zur Erweiterung werden zur Zeit ein älteres, direkt für Evolution entwickeltes Pluginsystem (EPlugin), sowie ein neueres auf dem Modulsystem von GObject aufsetzendes Erweiterungssystem genutzt[10].

Struktur der Pakete

Zur Umsetzung des Prototyps wurden musste das itip-formatter Plugin und die Benutzeroberfläche des Kalenders angepasst werden. Daher werden auch nur diese betrachtet.

[6] `https://live.gnome.org/Evolution/Evolution_Architecture?action=AttachFile&do=get&target=EvolutionArchitectureShell.png`

[7] Zu den Backends zählen unter anderem SMTP, NNTP, IMAP, POP 3, diverse Formate zur lokalen Speicherung von E-Mails, RSS, CalDav, GroupDav, HTTP, CouchDB, Microsoft Exchange, Scalix und Novell GroupWise.

[8] Backends können dabei nicht oder nur teilweise implementierte oder abbildbare Funktionalitäten einschränken oder auslassen.

[9] `http://burtonini.com/computing/taming-the-beast.pdf`

[10] So können via GObject auch beliebige Programmiersprachen mit GObject-Einbindung genutzt werden, was für das Projekt notwendig ist.

itip-formatter

Das itip-formatter Plugin ist als EPlugin umgesetzt, registriert sich (zur Laufzeit) hierüber als Darstellungskomponente für den MIME-Type `text/calendar` und fügt einen Konfigurationsreiter in das E-Mail Konfigurationsmenü ein.

Der itip-formatter ist in ein internes Daten- und Kontrollmodul und Darstellungsklasse (`ItipView`) unterteilt. Ersteres Modul liest die darzustellenden iCalendar Daten ein, verarbeitet diese und zeigt die Daten mit Hilfe der Darstellungsklasse an. Die Darstellungsklasse zeigt die iCalendar Daten dann anstatt einer E-Mail Nachricht samt aller Interaktionsmöglichkeiten an. Interaktionsereignisse der Benutzeroberfläche werden mittels des Signalmechanismus von GObject an das Daten- und Kontrollmodul übermittelt, dort verarbeitet. Im Vergleich zum bekannteren MVC-Entwurfsmuster sind Model und Controller verschmolzen, eine klare Trennung gibt es nicht.

Kalender

Der Kalender ist mittels des Modulsystems von GObject eingebunden und erweitert hauptsächlich die Benutzeroberfläche von Evolution (Shell). Er besteht aus der Evolution Data Server Anbindung und den Benutzeroberflächen für Kalender, Memos und Aufgaben und beinhaltet auch ein GNOME Kalender-Applet.

Da die Kalenderkomponente mit 90000 Zeilen Quelltext zu umfangreich für dieses Dokument ist und andere Teile für den Prototyp nicht verändert werden mussten, wird im Folgenden nur die Benuteroberfläche behandelt.

Gemäß dem GTK oft zu Grunde liegendem Ordnungsprinzip ist die Benutzeroberfläche in relativ eigenständige Fenster und Dialoge aufgeteilt, die sich untereinander aufrufen. Die Struktur der jeweiligen Klassen ist je nach Bearbeitungsstand und Entstehungsdatum verschieden. Einige Fenster und Dialoge werden imperativ durch Funktionsaufrufe, wie es früher bei GTK+ der Fall war, erzeugt, Andere wurden bereits vollständig oder teilweise auf eine deklarative Erzeugung mittels `GtkBuilder` und XML-Dateien umgestellt. Innerhalb des MVC-Entwurfsmusters verbinden solche Fenster View und Controller in einer Klasse, da innerhalb der Fenster jeweils auch gleich die durch Signale übermittelten Ereignisse verarbeitet und Aktionen auf dem mittels Evolution Data Server klar getrennten Daten durchgeführt werden.