C.U.O.N.

Jürgen Hamel

C.U.O.N.
von Jürgen Hamel

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1. C.U.O.N. - eine kfm. Entwicklungsumgebung

Mit C.U.O.N. sollten zwei Fliegen mit einer Klappe geschlagen werden:

Zum ersten soll es eine kaufmännische Anwendung werden, die es einer Firma ermöglicht, den normalen Geschäftsbetrieb unter LINUX ablaufen zu lassen, das heißt also : Adressen erfassen, Artikel bearbeiten, Aufträge mit Lieferschein und Rechnung drucken, Termine wahrnehmen, etc.

Aber mit C.U.O.N. soll man auch einfach neue Module einbinden können, und es soll ein Webinterface vorhanden sein

Schnell wurde klar, das mit PYTHON, GTK, GLADE, ZOPE, XML und POSTGRES die richtigen Partner zur Verfügung standen.

I. Bedienungsanleitung

Kapitel 2. Erste Schritte

2.1. Neuinstallation/Update

Nach einer Neuinstallation oder einem Update starten sie noch vor dem Einloggen den Menüpunkt:

Werkzeuge --- Erneuern

Hiermit werden die entsprechenden neuen GUI-Elemente, reports, etc. auf das Home-Verzeichnis des Clients abgelegt.

2.2. Benutzereinstellungen

Nach dem Anmelden gehen Sie zu folgenden Menüpunkt:

Werkzeuge --- Einstellungen

Es öffnet sich ein Fenster, in dem Sie jetzt Ihre persönlichen Einstellungen vornehmen.

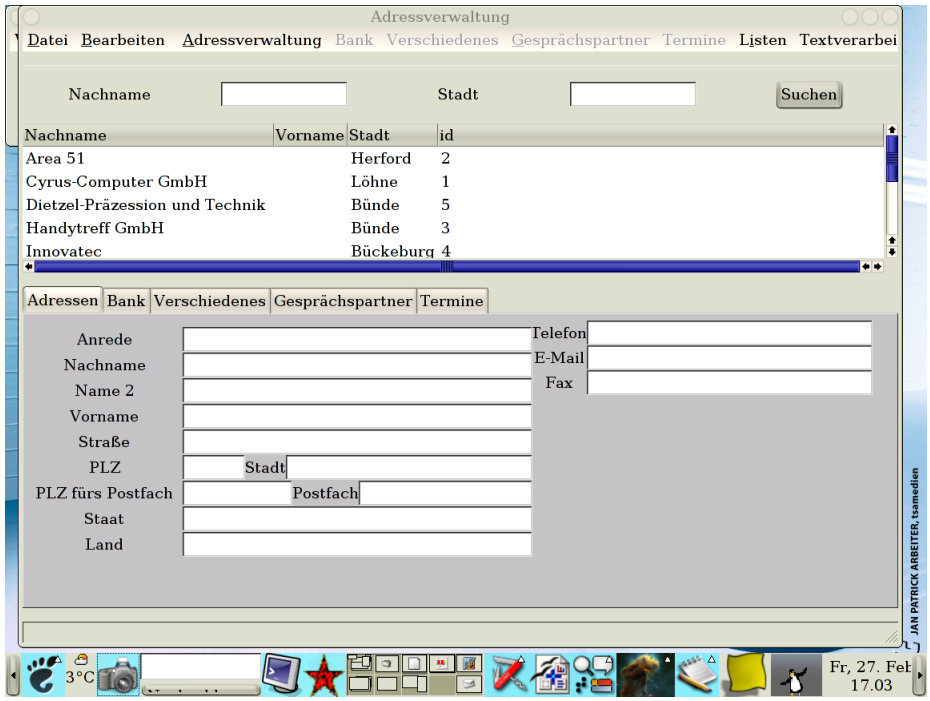
Die Pfade zu den Reports und den Dokumenten sind in der Regel firmenspecifisch und entsprechend nachzufragen, ansonsten wird es im Home-Verzeichnis unter cuon abgelegt.

Kapitel 3. Adressverwaltung

3.1. Adressen

In dieser Maske werden die Addressen eingegeben:

Abbildung 3-1. Adressverwaltung - Adressen eingeben



Dies sollte normalerweise der Firmenname oder Familienname sein. Evtl. Gesprächspartner können beliebig viele in der Make Gesprächspartner erfaßt werden.

Kapitel 4. Webinterface

4.1. Einleitung

Wenn Sie in einem beliebigen Browser (empfohlen wird mozilla) die Addresse Ihres Zope-Servers (im Normalfall http://server:9673/Cuon, SSL = https://server:8443/Cuon aufrufen

II. Development

Kapitel 5. Erste Schritte bei der Entwicklung von C.U.O.N.

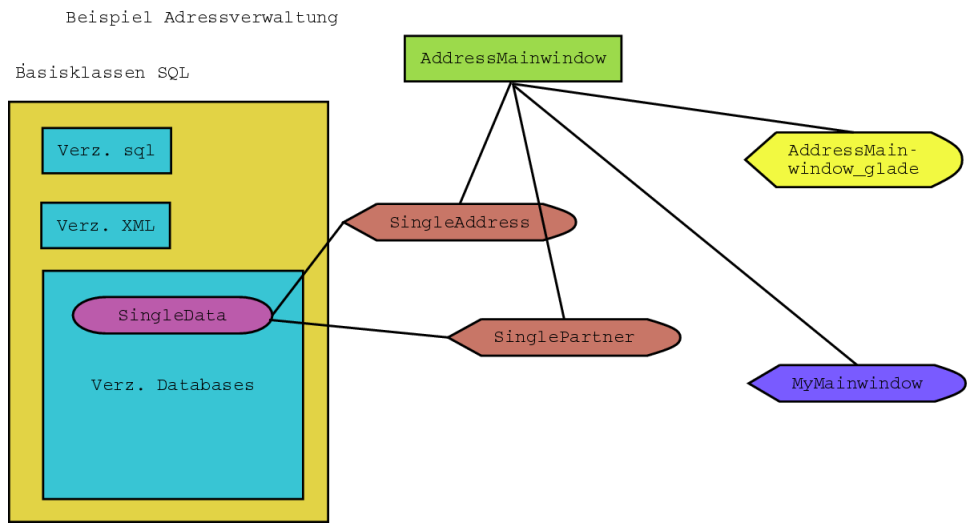
Nach einigen Diskussionen wurde also beschlossen, ein Open-Source-Projekt zu entwickeln, das es uns und anderen Entwicklern ermöglicht, unsere Arbeit auf ein flesibles Fundament aufzubauen. Dazu sollte das Projekt C.U.O.N. folgende Voraussetzungen erfüllen:

1. einfaches Erweitern für eigene Projekte
2. einfaches Pflegen und Verändern der Module
3. Standard sollte SQL und XML sein
4. Python, C++ und C als Programmiersprache
5. ZOPE als Aplication-Server

Dazu waren allerdings noch einige zusätzliche Vorüberlegungen notwendig. Bei kaufmännischen Anwendungen fällt auf, das ein Großteil der Datenoperationen ähnlich sind (Adressen, Artikel, Aufträge, etc., alle werden neu angelegt, verändert und gelöscht. Dies heißt, das die Verarbeitung der Daten recht ähnlich ist. Das Aussehen der Eingabemasken wird allerdings mehr von der Eigenart der Daten bestimmt. Aus diesen Überlegungen bot sich folgende Vorgehensweise an. Die Fenster werden mit einem GUI-Builder, in unserem Fall GLADE, erzeugt. Das Aussehen kann frei definiert werden. Die Daten werden durch eine abgeleitete Klasse von SingleData, einem Modul, das für das Anlegen, Updaten und Löschen der Daten zuständig ist, repräsentiert.

Um dies zu erreichen, hat ein Modul in C.U.O.N. folgende Grundstruktur (Beispiel Adressverwaltung):

Abbildung 5-1. Beispiel Adressverwaltung



Kapitel 6. Die Praxis bei der Entwicklung neuer Module

Um nun ein neues Modul, z.B. eine KFZ-Verwaltung einzubinden, sind folgende Schritte nötig:

Neues Verzeichnis anlegen
mit GLADE neue Bildschirmmaske erzeugen
mit XML-Editor neue Datentabellen in tables.xml, ext*.xml anlegen
Menüpunkte und Buttons mit CallbackRoutinen verbinden

Wir legen also als erstes ein neues Verzeichnis in cuon/src/ mit dem Namen "Car" an. Für den nächsten Schritt nutzen wir (Frau/Man macht sich sowenig Arbeit wie nötig) ein vorhandenes Script. Im Verzeichnis cuon/src/Skeleton ist ein Gerüst für ein entsprechendes Programmodul vorbereitet. Mit "sh createModul.sh Car " werden die entsprechenden Dateien in das neuangelegte Verzeichnis kopiert und überarbeitet. Nun wird im Verzeichnis cuon/src/Car mit glade "car.glade" der GUI-Builder aufgerufen und das entsprechende Bildschirm-Design erarbeitet. Um immer gleiche, englischsprachige Menüs und Buttons zu haben, rufe ich immer "export LANG=;glade" auf. Das erleichtert das Design und die Portierung in verschiedenen sprachen, z.B. Deutsch.

Die Datenbanktabellen werden durch 5 externe Dateien im XML-format gesteuert. Dabei gilt folgendes Regelwerk: Die Datei tables.xml ist für die Kerntabellen von C.U.O.N. zuständig, ext1.dbd für C.U.O.N. -Konfigurationstabellen sowie ext2.dbd für sonstiges. Die Dateien ext3.dbd und ext4.dbd sind für Ihre Projekt-eigenen Tabellen zuständig. Diese klare Trennung sorgt dafür, das sie mit C.U.O.N. nach Herzenslust Programmieren können, ohne das spätere Versionen Ihre Daten überschreibt. Mit dem xml-Editor Ihrer Wahl (wenn Sie noch keinen Lieblingseditor haben oder unsicher sind, schauen Sie in der Linux-Enterprise 07.2001 Seite 57ff nach. dort finden Sie eine reichliche auswahl), z.B. XEMACS, öffnen sie die Datei cuon/ext3.xml oder, falls Ihr Programm der Allgemeinheit zu Gute kommen soll, cuon/ext2.xml. Geplant ist auch hier die Nutzung eines ensprechenden tools, z.B. dbdesigner oder mergeant . Hier bin ich allerdings noch am Testen. Für Vorschläge bin ich immer offen.

Kapitel 7. Installation

7.1. Bezugsquellen

C.U.O.N. kann von Berlios.de (http://developer.berlios.de/projects/cuon) herunterladen werden. Alternativ gibt es auf www.cyrus.de (http://www.cyrus-computer.de/download) jeweils die neuste Version zum download.

7.2. Server Installation

7.2.1. Zope

Auf der Serverseite muß ZOPE laufen. Da ich nicht weiß, wie die ZOPE-Module auf anderen Systemen heißen, hier kurz die Debianliste:

libroxen-zopeg 1.6-3 Zope relay module for the Roxen Challenger w
zope 2.6.1-10 An Open Source Web Application Server
zope-book 20030918-1 Zope Open Content Book
zope-btreefold 0.5.0-2 Zope folder that can contain many more objec
zope-cmf 1.3.2-1 Zope Content Management Framework (CMF)
zope-cmfcalend 1.3.2-1 Zope CMF Calendar
zope-cmfcore 1.3.2-1 Zope CMF Core services
zope-cmfdefaul 1.3.2-1 Zope CMF Default (basic) content
zope-cmftopic 1.3.2-1 Zope CMF Topic
zope-devguide 20011206-4 Zope Developer's Guide
zope-parsedxml 1.3.1-3 ParsedXML Zope Product
zope-psycopgda 1.1.5.1-1 Zope Database Adapter based on python-psycop
zope-tinytable 0b2-8 Present tabular data in Zope
zope-znavigato 2.02-6 Zope product for creating navigation bars
zope-zpatterns 0.4.3p2-17 Database Independence and Framework Integrat
zope-zwiki 0.18.0-1 WikiWikiWeb on Zope
zopeedit 0.7-1 Helper Application for Zope External Editor

Unter ZOPE müssen jetzt die ZOPE-Module von C.U.O.N. importiert werden, sowie die externen Methoden aktualisiert werden

7.2.2. Postgres SQL

Postgres installieren, Database anlegen (im Zweifelsfall cuon nennen), grants.xml anppassen. Dieses XML-File von C.U.O.N. verwaltet die USER und deren Rechte.

7.3. Umgebungsvariablen

Setzen der Umgebungsvariable (Beispiele) :

1. Linux:

in der .bashrc setzen sie bitte die Variablen (Hier mit Beispiel)

nur Server "export CUON_INI=/etc/cuon/cuon.ini "

Client und Server "export CUON_HOME=~/cuon "

Client und Server "export CUON_SERVER=https://192.168.17.251:8443/Cuon, Hier gilt also IP_Adresse:Port/Verzeichnis (ohne SSL ist der Port im Regelfall 9673)

nur Client CUON_OOEXEC=/opt/OpenOffice.org1.1.0/program/python, dies ist das eingebaute Python von Openoffice für die uno-bridge

Wichtig ist hierbei, das sich der Wert Server auf den Rechner bezieht, von dem aus das Program cuon_server gestartet wird. Dies kann durchaus ein Computer sein, der auch als client definiert ist oder eine normale Arbeitsstation ist. (entsprechend muß die ip in cuon.ini angepaßt werden.

7.4. Verzeichnisse

Im Home-Verzeichnis (siehe CUON_HOME) müssen und sollten folgende Verzeichnisse existieren:

Reports

Docs

Am einfachsten werden diese und andere Verzeichnise mit der Datei /usr/lib/cuon/createUserDirs.sh angelegt

7.5. Konfiguration

7.5.1. cuon_server

Wenn Sie cuon_server.py gestartet haben, klicken Sie folgenden Menüpunkte an:

Tools --- Database --- SQL-Server --- db-check

Hiermit werden die Postgres-Tabellen unter cuon erzeugt

bevor Sie mit

Tools --- Database --- SQL-Server --- grant

die verschiedenen User und Gruppen anlegen

Anschließend führen Sie noch

Tools --- Database --- XML-Tools --- load Defaults

aus. Dies speichert die verschiedenen XML-Dateien für die z.B. GUI in eine objektorientierte Database auf den ZOPE-Server.

Kapitel 8. Benutzer

Zuerst muß ein Benutzer angelegt werden.

Ab Version 0.24 geschieht dies über die Datei grants.xml, später ist eine seperate Datei vorgesehen. Editeren sie die XML-Datei mit einem beliebige Editor und legen sie Ihren Benutzer an.

Ab Version 0.25 können Sie Benutzer und Gruppen über ein Webinterface anlegen.

Ordnen sie den Benutzer im zweifelsfalle der Gruppe cuon_all zu. Damit kann er so ziemlich alles in C.U.O.N. machen.

Eine vernünfitige Gruppenplanung sollte aber auf jeden Fall eingeführt werden.

III. Impressum

Kapitel 9. Entwickler

Jürgen Hamel

email: jh@cyrus.de
country: Germany