Universität Leipzig – Softwaretechnik Praktikum 2014 - xodx

Entwurfsbeschreibung

Entwurfsbeschreibung für den „Prototyp“ für das Projekt Xodx

Henrik Hillebrand
26.3.2014

Inhaltsverzeichnis

Allgemein

Innerhalb der Softwarestudie soll der bereits existierende Xodx-Knoten des DSSN um eine zusätzliche Funktion erweitert werden. Mit dieser neuen Unfriending-Funktion soll es dem Nutzer möglich sein, bereits geknüpfte Freundschaften aufzuheben. Die Nutzung dieser Funktion würde bewirken, dass von der zuvor abonnierten Person keine neuen Feeds mehr bezogen werden und die bereits vorhandenen gelöscht werden.

Produktübersicht

Das Ergebnis der Softwarestudie, kann bei laufendem Server - oder wahlweise eine lokal laufende Virtuelle Serverumgebung - über gängige Webbrowser aufgerufen und verwendet werden. Hierzu muss der Nutzer sich in seinen Xodx-Account einloggen, einen zuvor eingerichteten Test-Account nutzen oder einen neuen erstellen.

Nach erfolgreichem Login kann der Nutzer nun auf seinem eigenen Profil, in der Freundesliste, durch Klicken auf den "Unfriend"-Button einen Freund löschen, oder aber er besucht das Profil eben jener Person, wo er ebenfalls die Möglichkeit hat durch ein Klicken auf den "Unfriend"-Button, die Freundschaft zu beenden.

Die Funktion einen „Freund zu entfernen“ löscht diesen jedoch nicht nur aus der Freundesliste, sondern führt zu weiteren Abläufen die sich im Hintergrund ereignen. Denn das Aufrufen der Funktion führt dazu, dass die bisher von der Person erhaltenen Activities, wie zum Beispiel geteilte Bilder oder Feeds, aus der eigenen Activity-Liste entfernt werden und auch zukünftig keine neuen von eben jenem abonniert werden.

Grundsätzliche Struktur- und Entwurfsprinzipien

Da unser gewähltes Vorprojekt auf dem bereits erwähnten Xodx-Knoten aufbaut und für diesen als Erweiterung dient, kann als grundsätzliche Struktur der von unserer Gruppe entwickelten Softwarestudie das Strukturprinzip des Xodx als gegeben empfunden werden. Dies gilt auch aus dem Grund, da in Xodx bereits die Funktion zum Hinzufügen von Freunden implementiert ist und die Lösch-Funktion selbiger eine Abwandlung dieser darstellt.

Grundsätzlich baut Xodx wiederum auf dem DSSN auf, welches intern in 3 verschiedene Ebenen (layer) unterteilt ist: Die Daten-, Protokoll-, Anwendungs- und Service-Ebene.

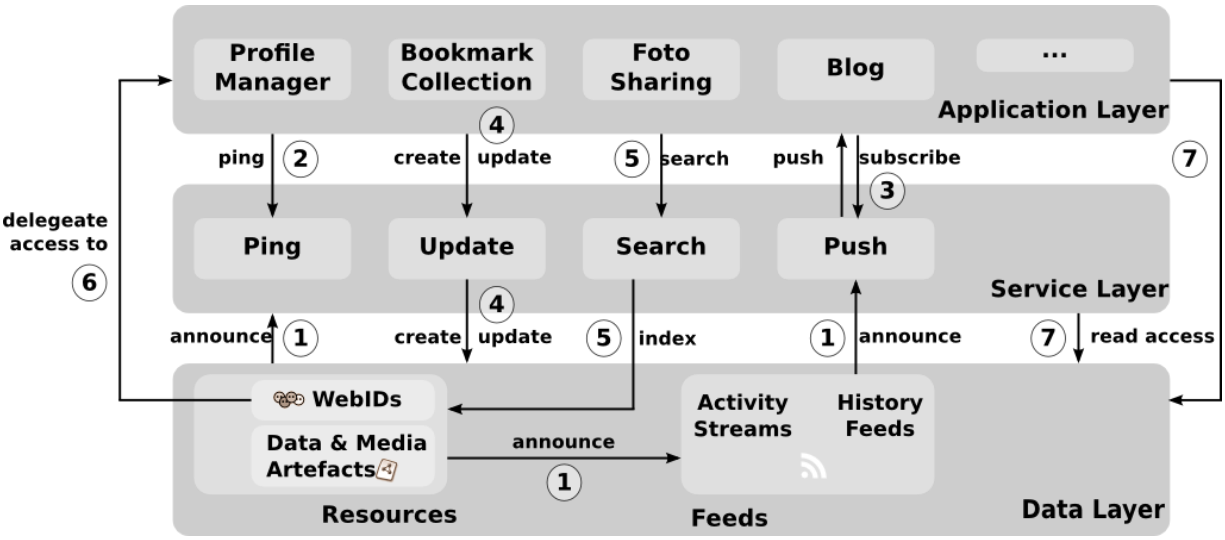

Abbildung 1: Ebenen-Modell des DSSN

Die Daten-Ebene beinhaltet die RDF-Ressourcen, Media-Daten wie Bilder und letztlich die Activity-Feeds. Diese Daten werden über einen URI eindeutig benannt und können per Linked Data abgerufen und benutzt werden. Zudem besteht noch die Möglichkeit jede Ressource zu kommentieren oder andere Relationen zwischen ihnen zu erzeugen (Freundschaften zwischen Personen-Ressourcen o.ä), welche auch in der Daten-Ebene gespeichert werden.

Die Protokoll-Ebene ermöglicht die Kommunikation der einzelnen Dienste des DSSN untereinander. Da es sich bei Xodx um eine Webapplikation handelt, ist dieser Ebene zum einen das HTTP zuzuordnen, allerdings auch nötige Protokolle wie das zuvor angesprochene Linked Data, das Semantic Pingback und PubSubHubbub.

Die Service- und Anwendungsebene stellt die Dienste oder Anwendungen bereit - mit anderen Worten die Oberfläche -, die gebraucht werden um innerhalb des Knotens auf Konten registrierter Xodx-Nutzer zuzugreifen, aber auch für beliebige andere Nutzer des DSSN. Die Ebene verfügt daher bspw. über den Profilmanager, das Teilen von Media-Daten und Benachrichtigungen.

Zusätzlich zu erwähnen sei, dass Xodx selbst, und damit auch unsere Erweiterung, auf bereits bestehende Bibliotheken und Frameworks zurückgreift, die die Arbeit mit den angesprochenen Protokollen und RDF-Ressourcen erheblich vereinfacht. Dieses vermeidet Dopplungen im Code und erhöht die Qualität des Quellcodes wenn es um Struktur und Übersichtlichkeit geht.
So nutzt Xodx das Erfurt-Framework - welches dem Programmierer den Zugriff auf Triple Stores und deren Modelle vereinfacht und mittels SPARQL die Nutzung dieser effizient ermöglicht -, das Saft-Framework - welches als Grundgerüst für Xodx dient und das MVC-Entwurfsmuster nutzt - und letztlich das lib-dssn-php - welches die Verarbeitung von Aktivitystreams erleichtert.

Da durch das SAFT-Framework das MVC-Entwurfsmuster verwendet wird, liegen als zusätzliche Strukturelle Ebene die Model-, View-, und Controllerebene über den angesprochenen DSSN-Ebenen, welche durch ein Layout-System auf der View-Ebene und der Controller "ResourceController" und zusätzliche actions in der Controller-Ebene die den Zugriff für den Programmierer noch weiter vereinfachen.

Struktur- und Entwurfsprinzipien einzelner Pakete

Wie zuvor erwähnt dient das Saft-Framework dem Zweck dem Entwurfsmuster MVC konkret in Xodx folge leisten zu können. Daher ist im nachfolgenden die Struktur der einzelnen Pakete in die Ebenen eben jenes Entwurfsprinzips unterteilt und in diesem näher erläutert.

Model

Unsere Funktion greift im Bereich der Model-Ebene, welche für die Speicherung der Daten und Anbindung an die Datenbank zuständig ist, auf die in dem Triple Store gespeicherten Einträge zu und löscht die Relation zwischen der eigenen URI und der des ehemaligen Freundes. Diese Relation ist geknüpft über das Prädikat "http://xmlns.com/foaf/0.1/knows"

Im weitesten Sinne gehören daher auch die RDF-Ressourcen kurz erwähnt, welche über den Triple Store in Relation zueinander gesetzt werden. Diese Verbindung wird von uns, falls eine Löschung per "Unfriending" eingeleitet wurde, entfernt, was allerdings die RDF-Ressource selbst in keiner Weise ändert.

View

Auf der View-Ebene wurde das Template des eigenen Profils und das anderer Profile so weit verändert, dass in der dynamisch erzeugten Freundesliste nun Unfriend-Button hinzugefügt wurden, welche das Löschen von Freunden auf dem eigenen Profil ermöglicht und der "Add as Friend"-Button

nach dem "Friending" in einen "Unfriend"-Button umgewandelt wird, welcher ebenfalls zum löschen des jeweiligen Freundes dient. Diese Veränderungen betreffen jedoch nur die Oberfläche, also die zwei genannten Templates, aber nicht das Layout-System des Xodx.

Controller

Innerhalb der Controller-Ebene fanden einige Erweiterungen statt, die sich innerhalb der folgenden Controller wiederfinden:

- UserController
- PersonController
- PushController

So wurden innerhalb des UserControllers die zwei Funktionen "unsubscribeFromRressource" und "unsubscribeFromFeed" implementiert. Die erste der beiden Funktionen ermöglicht das Entfernen der Subskription ohne explizite Übergabe einer Zieladresse, wohingegen die zweite selbe Funktionalität bietet, hier jedoch mit erforderlicher Übergabe. Dies ist notwendig, da beim "Unfrienden" über das eigene Profil eine Zieladresse vom Nutzer mitgegeben wird, wohingegen das "Unfrienden" auf einem Freunde-Profil die Zieladresse über das Objekt des Freundes geliefert wird.

Der PersonController wurde um die Funktion "deleteFriend" erweitert, welche dafür sorgt, dass die einseitige Freundschafts-Verbindung gelöscht wird. Dies bedeutet auch, dass innerhalb dieser Funktion auf die beiden oben erläuterten Funktion zur Entfernung der Subskription zugegriffen wird.

Letztlich wurde im Pushcontroller die Funktion "unsubscribe" hinzugefügt, welche die Benachrichtigungen über neue Feed-Updates abbestellt und die Verbindung seitens PubSubHubbub trennt.

Datenmodell

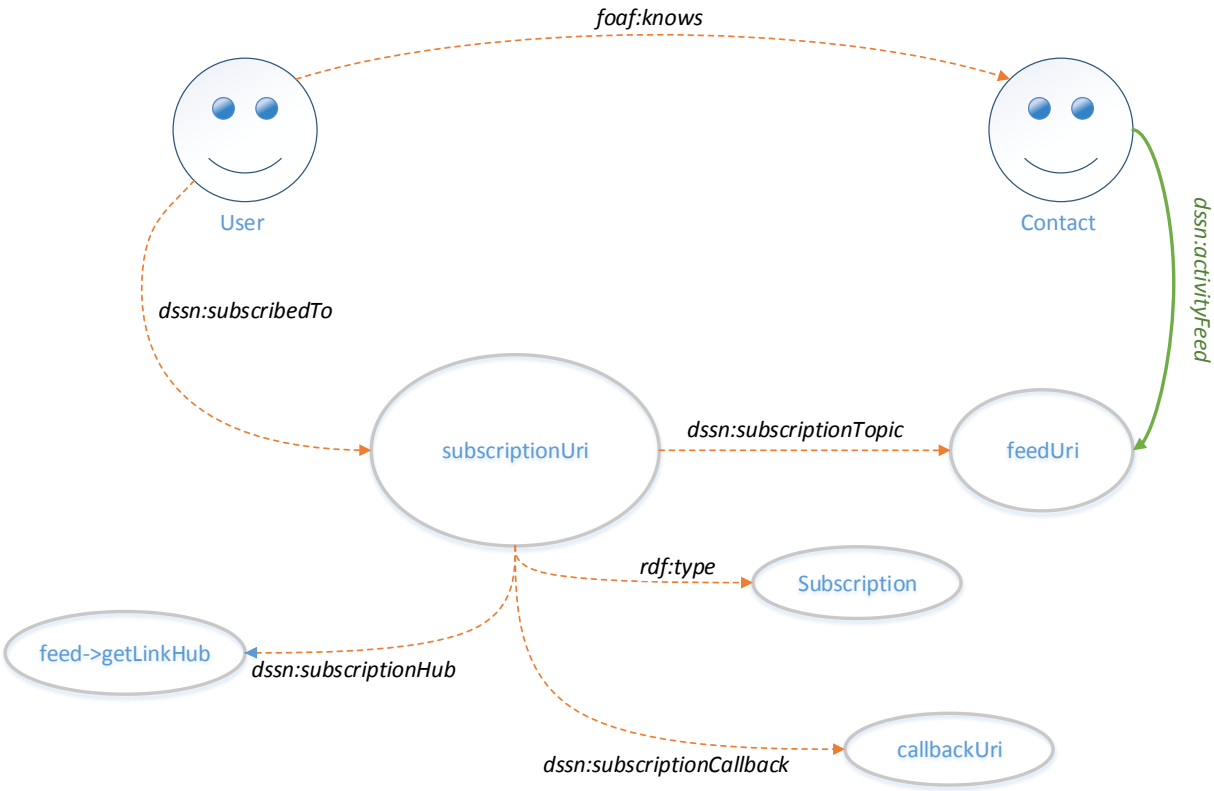

Abbildung 2: Datenmodell „Freunde-löschen-Prozess“

Testkonzept

Da die größte Schwierigkeit im Vorprojekt vielmehr die Einarbeitung in das System selbst, die für uns bisher unbekannten RDF-Ressourcen, die ActivtyStreams, das Semantic Pingback und das Triple Store System war und weniger die eigentliche Implementierung, wurde vorerst auf automatisierte Testverfahren mittels PHPUnit o.Ä. verzichtet.

Alle zu implementierenden Funktionalitäten wurden durch den Verantwortlichen für Test und die Programmierer selbst getestet und als funktionsfähig erachtet.

Glossar

Triple Store

Ein Tripel ist die Einheit bestehend aus drei Elementen Subjekt, Prädikat und Objekt. Eine Ressource als Subjekt wird mit einer weiteren Ressource als Prädikat zusammen mit einer dritten Ressource als Objekt näher beschrieben. Das Tripel besteht aus den beiden Enden des Links und der Art des Links und ist somit eine logische Aussage.
Die Link-Struktur des Tripels bildet einen gerichteten benannten Graph, welcher die logische Aussage darstellt. Die Graphdatenbank benutzt diese um vernetzte Informationen darzustellen und zu speichern.

Linked Data

Bezeichnet im WWW Daten, welche frei verfügbar sind und über einen zugehörigen URI direkt via HTTP abgerufen werden können. Zusätzlich besteht die Möglichkeit, auf andere Daten zu verweisen, welche auch über einen URI identifiziert werden.

ActivitySteam

Ein Activity Stream ist eine Liste von Aktivitäten, die ein Individuum (z.B. Person oder Gruppe), getätigt hat.

Semantic Pingback

Semantic Pingback ist eine, auf die Veröffentlichung von RDF-Ressourcen mit Linked Data angepasste Methode, die es Web-Autoren erlaubt Benachrichtigungen anzufordern, falls jemand eine Ressource verlinkt.

DSSN

Das DSSN ist ein dezentralisiertes semantisches Soziales Netzwerk der AKSW

Quellen

Literatur

- "An Architecture of a Distributed Semantic Social Network" von Sebastian Tramp (28.03.14)
- Natanael Arndt (2013): Xodx, Konzeption und Implementierung eines Distributed Semantic Social Network Knotens (Masterarbeit)

Abbildungsverzeichnis