

Praxis der Softwareentwicklung: Visualizing Trends. Was verrät uns Twitter?

Entwurf

Maximilian Awiszus Holger Ebhart Lidia Grigoriev
Paul Jungeblut Philipp Kern Matthias Schimek



WS 2014/15

Inhaltsverzeichnis

1	Einführung	3
2	Datenbank	4
2.1	Datenbankzugriff	4
2.2	ER-Modell	5
2.3	Datenflussdiagramm	5
3	Crawler	8
3.1	Aufbau	8
3.2	Start des Crawlers	9
3.3	Verarbeitung der Daten von Twitter	9
4	Kategorisierer	13
4.1	Aufbau	13
4.2	Start des Kategorisierers	13
5	GUI	15
5.1	Aufbau	15
5.2	Sequenzdiagramme	15
6	Glossar	19

1 Einführung

hier nochmal das Systemmodell mit einer detaillierteren Beschreibung dann evtl. noch Paketdiagramme vom Crawler/GUI

2 Datenbank

2.1 Datenbankzugriff

Die zentrale Komponente unseres Systems ist die Datenbank. In sie fügt der Crawler neue Datensätze ein und aktualisiert vorhandene. Der Kategorisierer ist dafür zuständig, dass die gefundenen Accounts nach der DMOZ.org Datenbank in Kategorien unterteilt werden. Die GUI wiederum ist die Komponente die die Daten aus der Datenbank ausliest und visualisiert. Gegebenenfalls kann sie auch Einträge verändern beziehungsweise vervollständigen.

Da alle unsere drei Systemkomponenten lesend, sowie schreibend auf die Datenbank zugreifen, haben wir uns entschlossen ein Paket für den Datenbankzugriff für alle Komponenten zur Verfügung stellen. Dieses sogenannte mysql-Package ist dann für den Auf- und Abbau der Verbindung zur Datenbank zuständig, sowie für das Schreiben und Lesen in beziehungsweise aus der Datenbank. Es stellt für jede der drei Komponenten ein eigenes Interface zur Verfügung, sodass jede Komponente nur die für sie erlaubten Änderungen an der Datenbank vornehmen kann. In Abbildung 2.1 ist der Aufbau des mysql-Packages zu sehen. Das darin eingeschlossene result-Package stellt Objekt und Methoden zu Verfügung um die Ergebnisse aus der Datenbank zu handeln.

AccessData Klasse zur Verwaltung der Zugriffsdaten für die Datenbank.

DBConnection Abstrakte Klasse die eine Verbindung zu einer Datenbank aufbaut und diese Verbindung auch wieder trennt.

DBIcrawler/DBIcategorizer/DBIGUI Interface's welche die Methoden spezifizieren die der Crawler, der Kategorisierer bzw. die GUI für den Datenbankzugriff benötigen.

DBcrawler Diese Klasse implementiert die Methoden des DBIcrawler Interfaces und stellt dem Crawler eine Datenbankverbindung zur Verfügung.

DBcategorizer Diese Klasse implementiert die Methoden des DBIcategorizer Interfaces und stellt dem Kategorisierer eine Datenbankverbindung zur Verfügung.

DBgui Diese Klasse implementiert die Methoden des DBIGUI Interfaces und stellt dem Client / der GUI eine Datenbankverbindung zur Verfügung.

Result Als abstrakte Klasse stellt Result eine Möglichkeit zum Speichern des Datenbankindexes von Datenbankeinträgen zur Verfügung.

Account In dieser Klasse werden einzelne Accounts verwaltet und gespeichert.

Retweets In dieser Klasse werden nach Orten (und eventuell nach Daten) gruppierte Retweets verwaltet und gespeichert.

Tweets In dieser Klasse werden nach Daten gruppierte Tweets verwaltet und gespeichert.

Location In dieser Klasse werden einzelne Orte verwaltet und gespeichert.

Category In dieser Klasse werden einzelne Kategorien verwaltet und gespeichert.

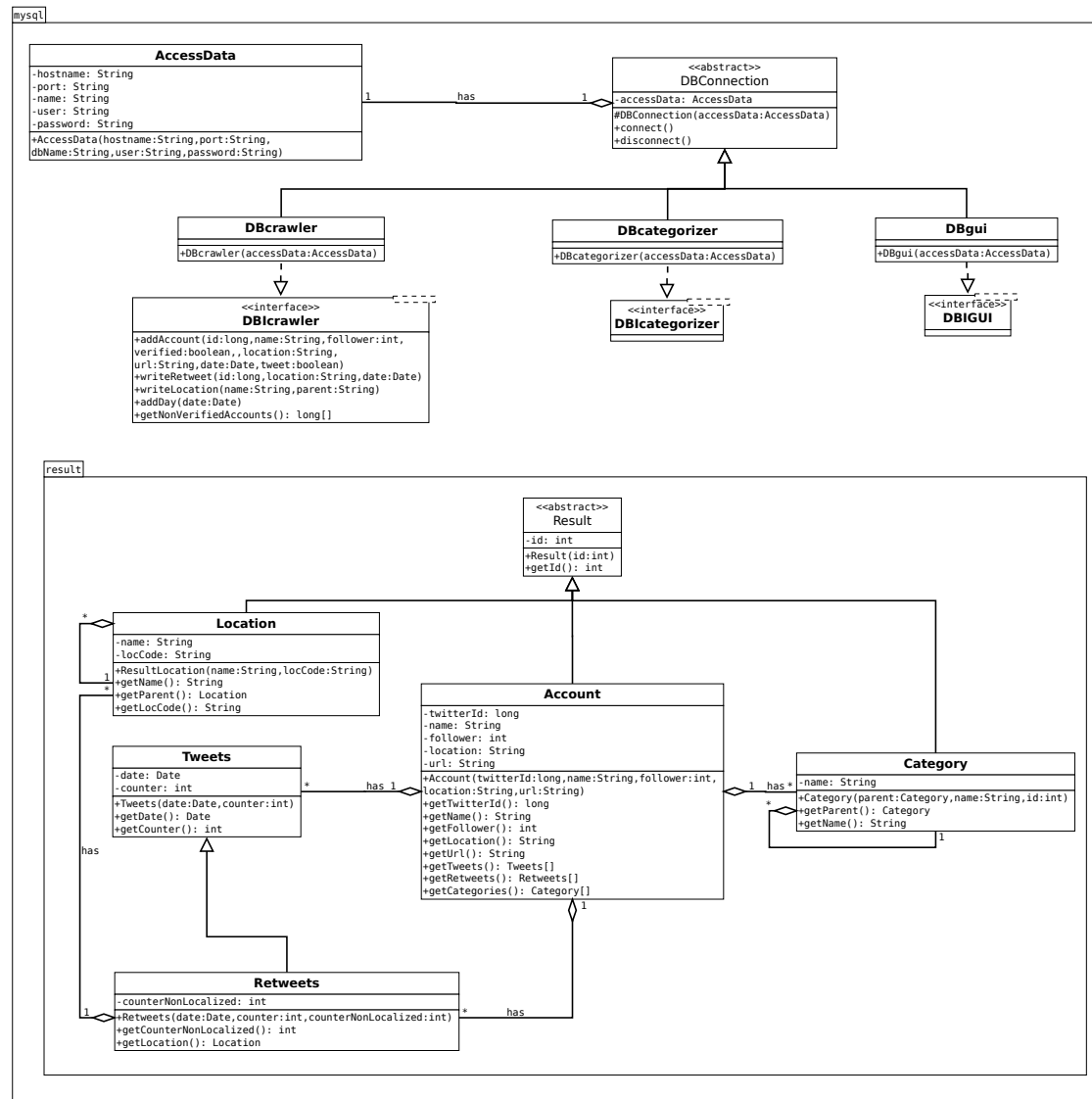


Abbildung 2.1: UML-Klassendiagramm des mysql-Packages

2.2 ER-Modell

2.3 Datenflussdiagramm

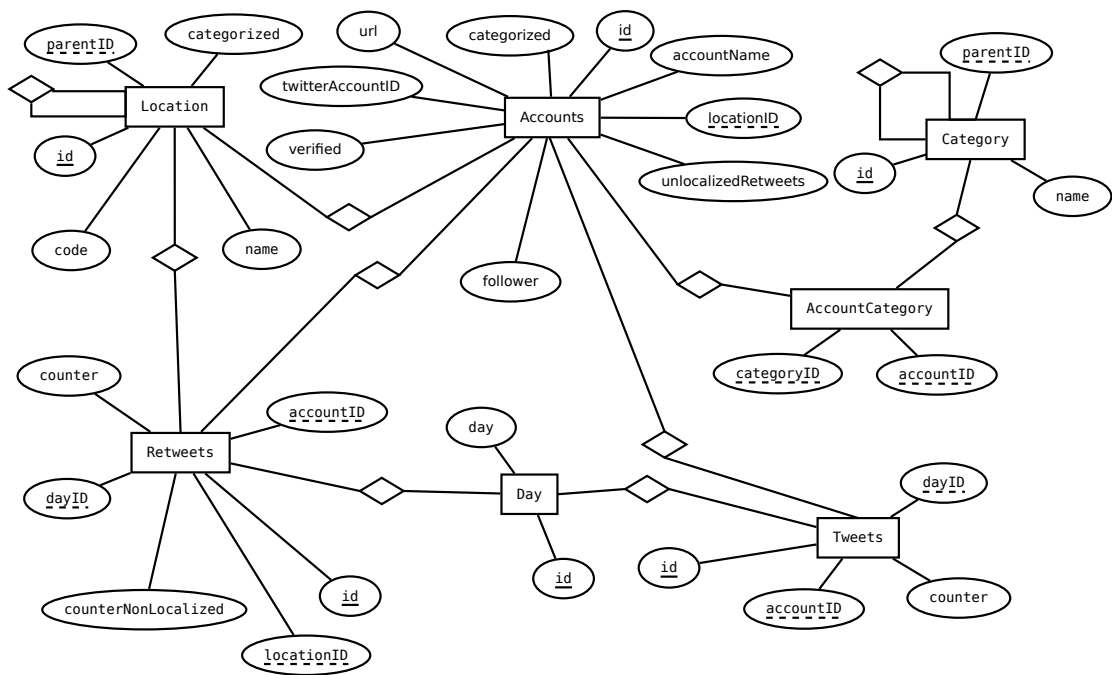


Abbildung 2.2: ER-Modell der MySQL-Datenbank

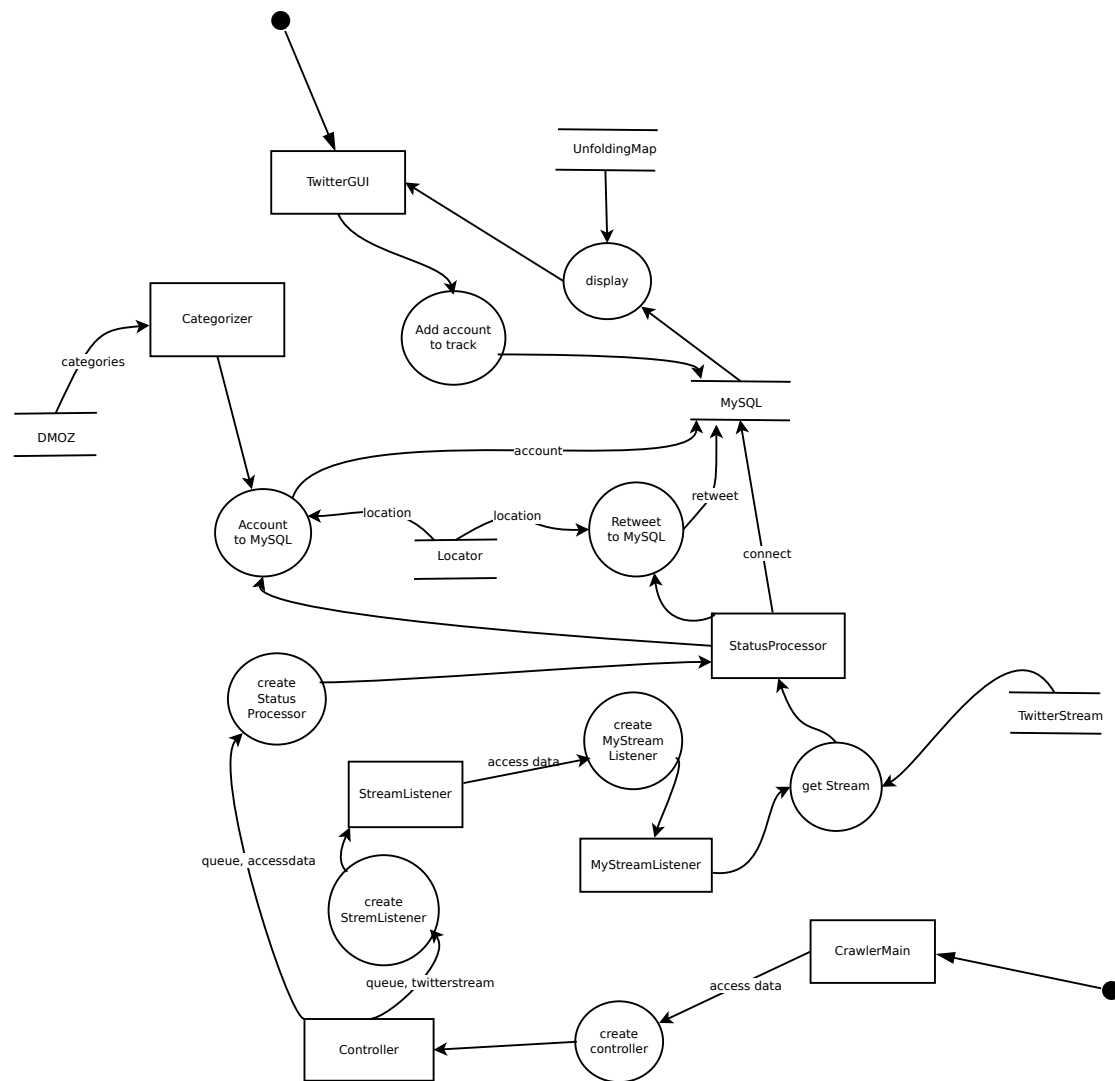


Abbildung 2.3: Datenflussdiagramm

3 Crawler

3.1 Aufbau

Zum Sammeln von Daten von Twitter verwenden wir einen Crawler, welcher über die Twitter-API Daten sammelt. Dazu ist es nötig diese Daten zu empfangen, dann zu puffern und schlussendlich in die Datenbank zu Schreiben. Allerdings müssen die Daten noch gefiltert werden, da wir nur Daten von verifizierten Twitter-Accounts (und manuell hinzugefügten) speichern. Sind die Daten gefiltert, werden sie vom Crawler noch lokalisiert. Das heißt, dass jedem Account beziehungsweise jedem Retweet eine Geoposition/Land zugeordnet wird. Ist dies erfolgt so werden die Daten in die Datenbank geschrieben.

In Abbildung 3.1 ist der Aufbau des Crawlers anhand eines UML-Klassendiagramms spezifiziert.

CrawlerMain Klasse dient als Einstieg ins Programm. Sie überprüft die Eingabe für die Datenbankverbindung und startet einen Controller. Danach sieht sie dem Benutzer über die Konsole zur Verfügung um das Programm zu überwachen.

RunnableListener Interface welches Runnable erweitert und zusätzlich noch eine exit-Methode fordert um Threads zu beenden.

Controller Diese Klasse koordiniert alle Aktionen die nötig sind um Daten bei Twitter abzuholen und in die Datenbank zu Schreiben. Dazu startet sie einen StreamListener, ein AccountUpdate und mehrere StatusProcessor's jeweils als Thread. Außerdem kontrolliert sie den Puffer und sorgt für ein sauberes Beenden des Programms indem alle Verbindungen ordnungsgemäß geschlossen und die Threads beendet werden.

StreamListener Stellt eine Verbindung zur Twitter-Streaming-API her und initialisiert einen MyStatusListener.

AccountUpdate Diese Klasse stellt eine Methode zur Verfügung um in der Datenbank periodisch nach Accounts zu suchen, welche manuell hinzugefügt wurden, aber auch wie Verifizierte behandelt werden sollen.

StatusProcessor Diese Klasse stellt die Funktionalität zur Filterung der Daten von Twitter zur Verfügung, welche sie aus dem Puffer nimmt. Außerdem bietet sie die Möglichkeit diese Daten in die Datenbank zu schreiben.

MyStatusListener Diese Klasse nimmt die Daten von Twitter entgegen und schreibt diese in einen Puffer.

MyRateLimitStatusListener Diese Klasse nimmt Meldungen von Twitter bezüglich Rate-Limits entgegen.

Locator Der Locator lokalisiert Accounts und Retweets mithilfe eines Webdienstes.

3.2 Start des Crawlers

Beim Starten des Crawlers werden sämtliche notwendigen Komponenten der Reihe nach gestartet. Dadurch wird garantiert, dass jede Komponente eine Umgebung vorfindet in der sie laufen kann und alle Ressourcen bereits zur Verfügung stehen. In Abbildung 3.2 ist der Start des Crawlers beispielhaft mit 2 StatusProcessor's dargestellt.

3.3 Verarbeitung der Daten von Twitter

Um zu verdeutlichen wie die Daten von Twitter innerhalb des Crawlers verarbeitet werden, ist in Abbildung 3.3 der Datenfluss durch den Crawler exemplarisch dargestellt. Dabei werden die Daten von Twitter abgeholt, gepuffert, dann gefiltert und schlussendlich in die Datenbank geschrieben.

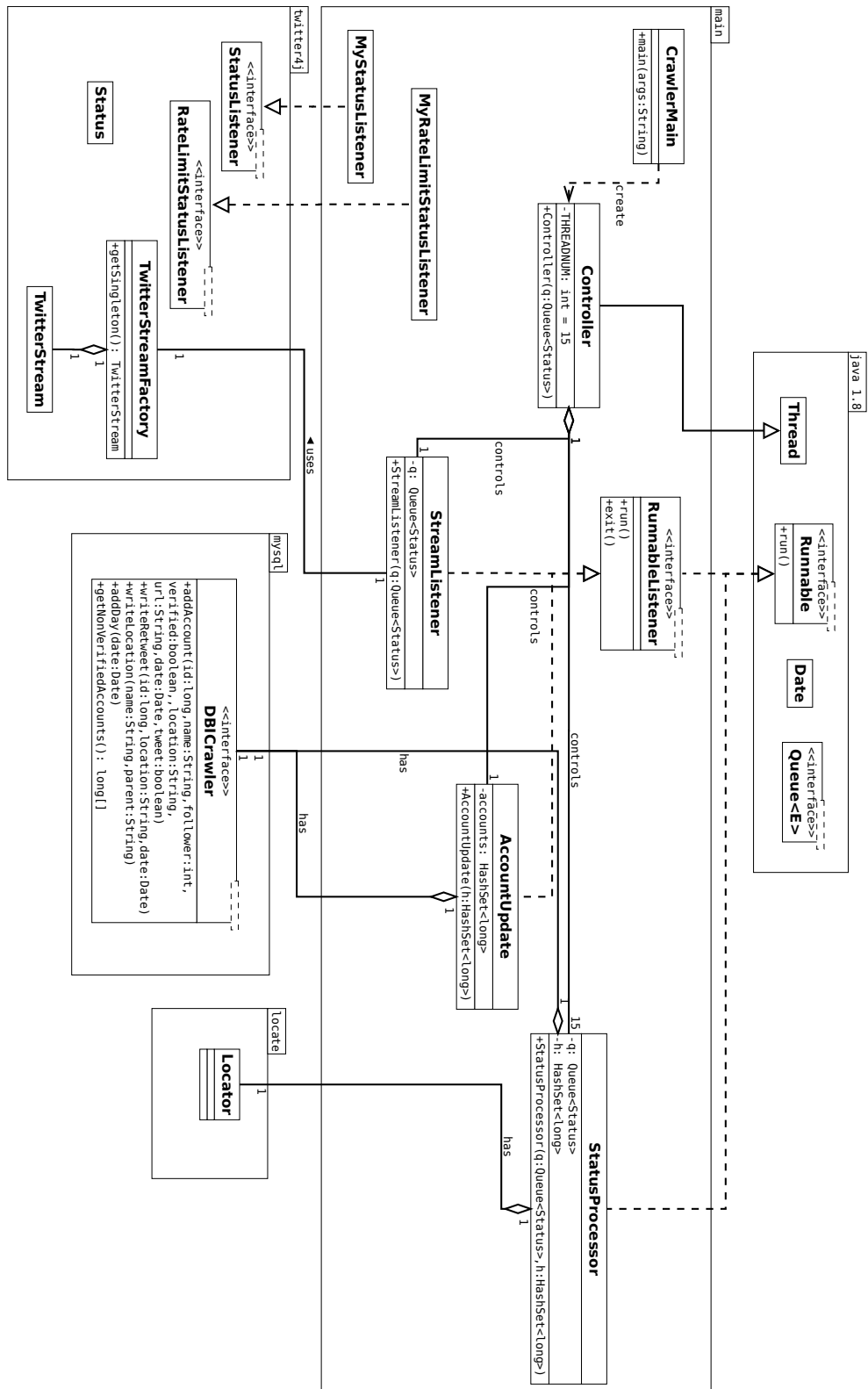


Abbildung 3.1: UML-Klassendiagramm des Crawlers

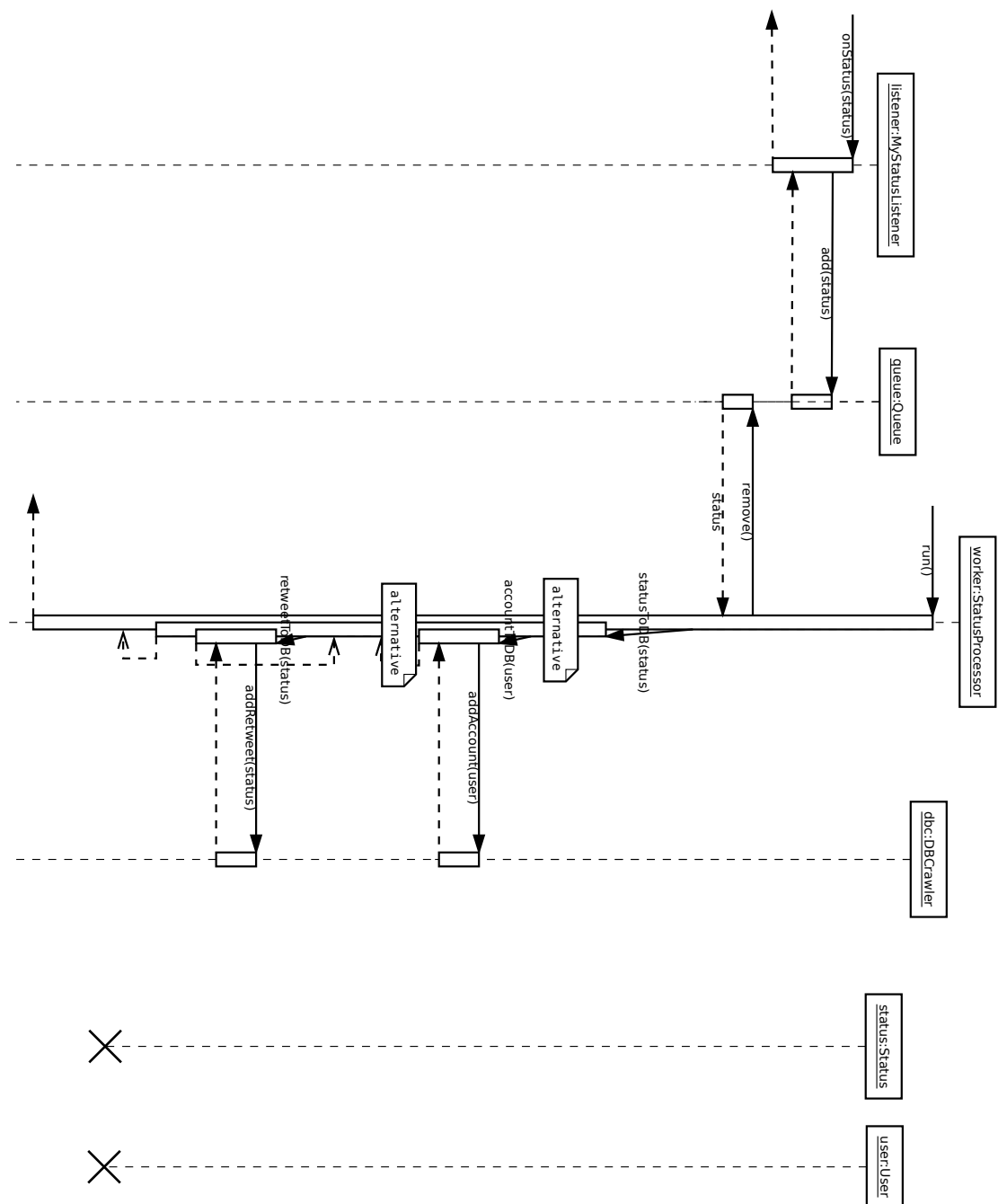


Abbildung 3.3: Sequenzdiagramm der Verarbeitung der Daten von Twitter

4 Kategorisierer

4.1 Aufbau

Der Kategorisierer wird in regelmäßigen Abständen vom Betriebssystem gestartet und verbindet sich mit der Datenbank. Über die Datenbankschnittstelle ließt er die bislang unkategorisierten Twitteraccounts aus Accountstabelle aus und sucht in der DMOZ-Datenbank nach passenden Kategorien. Diese werden dann in die Kreuztabelle AccountCategory eingetragen.

Im Diagramm 4.1 ist der grundlegende Aufbau des Kategorisierers dargestellt:

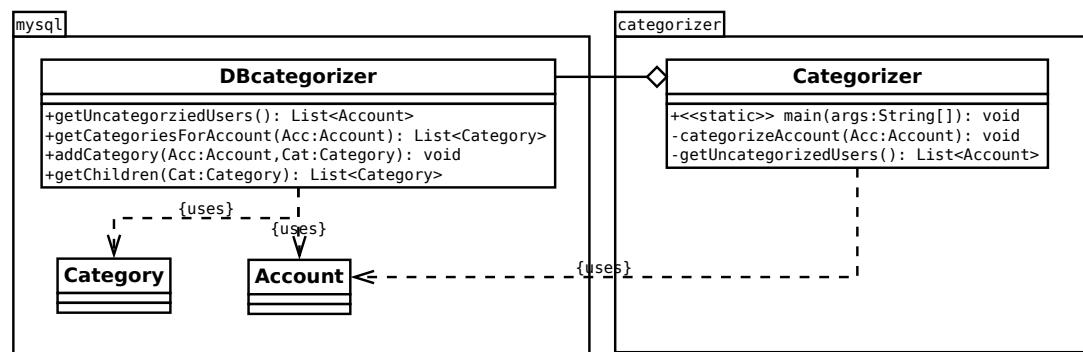


Abbildung 4.1: Klassendiagramm des Kategorisierers

Account siehe Abschnitt 2.1

Category siehe Abschnitt 2.1

DBcategorizer Über diese Klasse kommuniziert der Kategorisierer mit der Datenbank. Sie enthält Methoden zum Holen der unkategorisierten Accounts, zum Finden von Kategorien, zum Eintragen einer Kategorie und zum Auffinden aller Subkategorien einer Kategorie.

Categorizer Dies ist die Haupt-Klasse des Kategorisierers. Sie nutzt die Methoden von **DBcategorizer**, um unkategorisierte Accounts zu suchen und gefundene Kategorien einzutragen.

4.2 Start des Kategorisierers

Der Kategorisierer soll in regelmäßigen Abständen vom Betriebssystem gestartet werden und daraufhin die neu gefundenen Accounts kategorisieren.

Der Ablauf des Kategorisierers ist im Sequenzdiagramm 4.2 zu sehen.

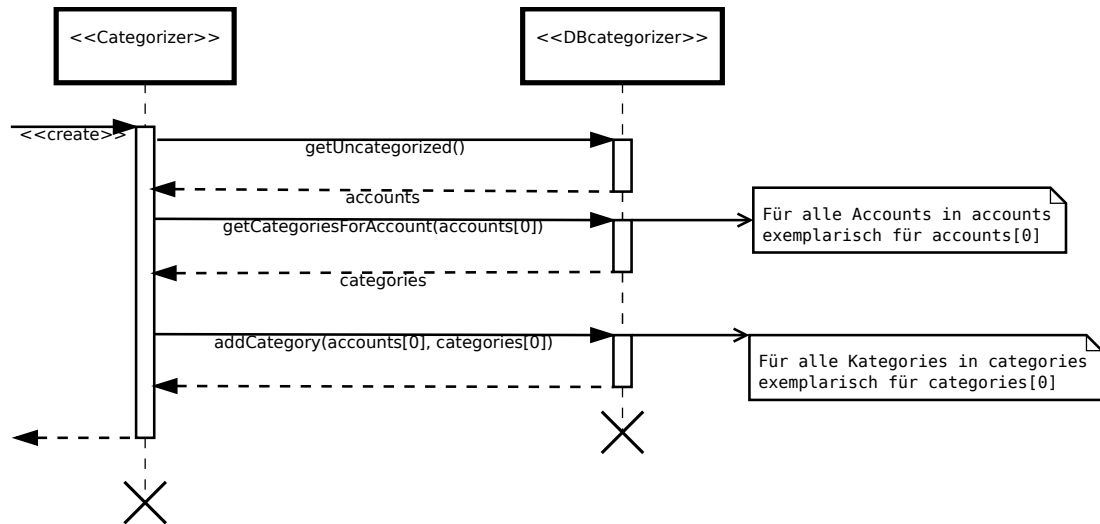


Abbildung 4.2: Sequenzdiagramm für einen Durchlauf des Kategorisierers. Dabei sind exemplarisch nur jeweils der erste unkategorisierte Account und die erste gefundene Kategorie aufgeführt.

Im ersten Schritt wird also eine Liste von unkategorisierten Accounts ausgelesen. Für jeden dieser Accounts wird eine Liste passender Kategorien ermittelt, die dann nach und nach in die Datenbank geschrieben werden.

5 GUI

5.1 Aufbau

Die GUI ermöglicht die Interaktion des Benutzers mit der Anwendung und stellt die über den Crawler gesammelten Daten grafisch aufbereitet dar.

GuiController Diese Klasse enthält alle GUI-Elemente. Damit kann über diese Klasse jedes einzelne Element angesprochen und damit gesteuert werden. Außerdem speichert sie die jeweils aktuellen Resultate der Datenbankabfragen zentral. Jede Erweiterung muss sich im GuiController als Observer eintragen, um über Änderungen der Daten informiert zu werden.

GuiElement Interface, das jedes GUI-Element implementieren muss.

SelectionOfQuery Dieses Paket enthält Darstellung und Anwendungslogik für die Auswahl einer Suchanfrage (Auswahl von Kategorie, Land, usw.)

databaseOptions Dieses Paket enthält die Darstellung und Anwendungslogik für Änderungen an der Datenbank, wie das Hinzufügen eines bisher nicht mitgetrackten Accounts.

standardMap Das Paket enthält Anwendungslogik und Darstellung für die Standardkarte, welche die Länder nach dem jeweiligen Tweet-Retweet-Aufkommen einfärbt.

table Paket, welches Anwendungslogik und Darstellung für die Erstellungen und Anzeige des Datenblattes zur aktuellen Anfrage enthält.

timeSliderMap Paket, welches Anwendungslogik und Darstellung für Erstellung und Anzeige des Tweet-Retweet-Aufkommens in Abhängigkeit des gewählten Zeitraums anzeigt.

myUnfoldingMap Diese Klasse kapselt die eigentliche Darstellung sämtlicher Kartenanzeigen. Sie ist die Schnittstelle für Unfolding-Library, welche für die Anzeige der Weltkarte verwendet wird.

5.2 Sequenzdiagramme

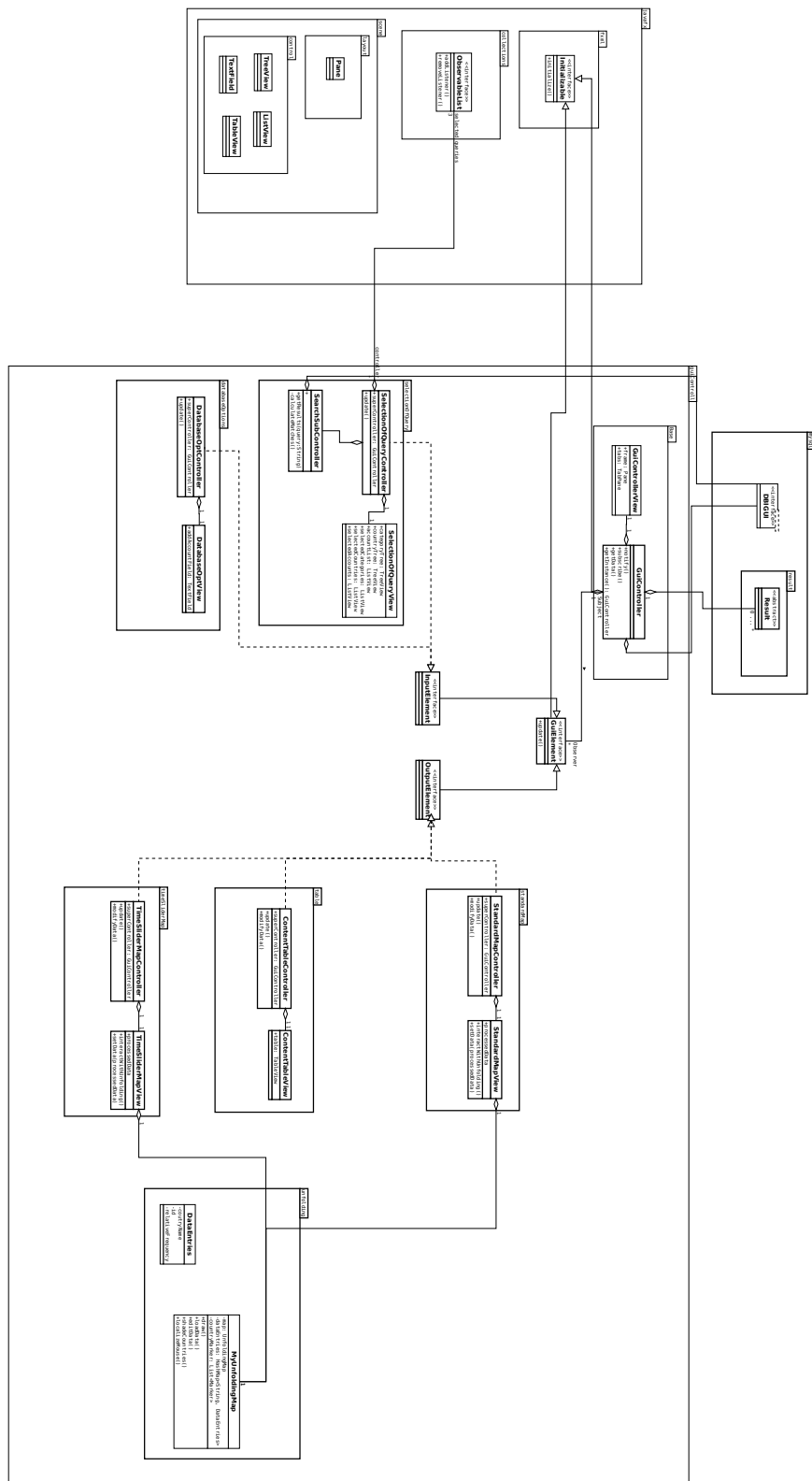


Abbildung 5.1: Klassendiagramm der GUI

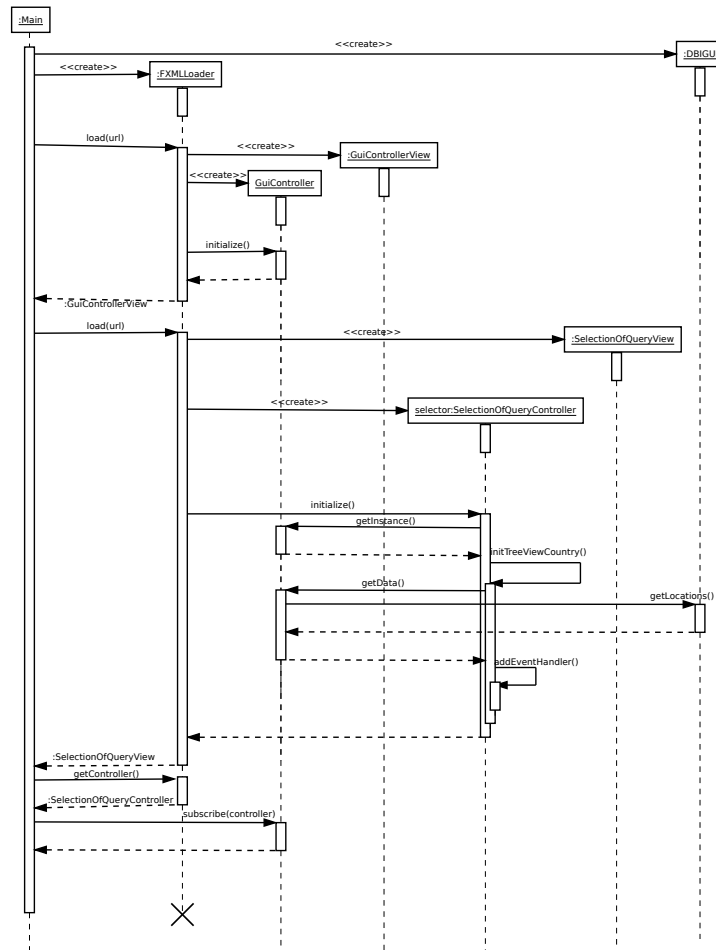


Abbildung 5.2: Sequenzdiagramm der Initialisierung der GUI.

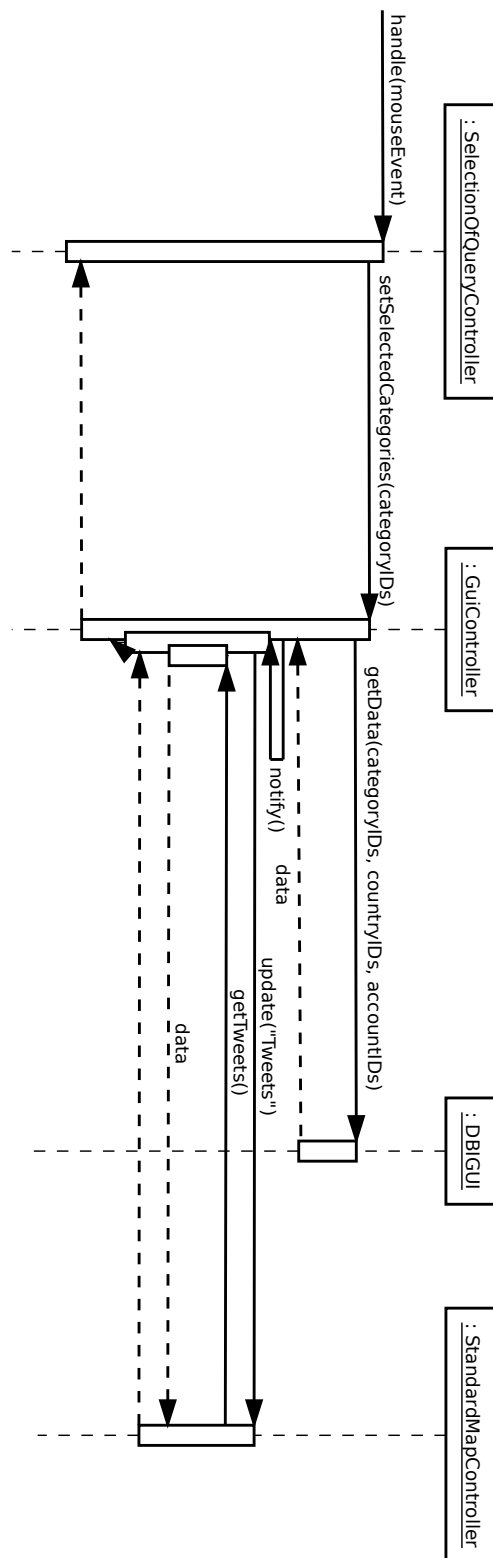


Abbildung 5.3: Sequenzdiagramm für Auswahl einer neuen Kategorie in der GUI.

6 Glossar