FACHHOCHSCHULE VORARLBERG GMBH

Projekt Dokumentation

Roadrunner

ausgeführt von

Franziskus Domig, Bsc;

Stefan Gassner, BSc;

Wolfgang Halbeisen, BSc;

Matthias Schmid, BSc

Bearbeitung: Dornbirn, im Frühjahr 2011

Betreuer: XXX

1 Vorwort

TODO

2 Spezifikation

Roadrunner ist ein System zur Überwachung von Arzneimittel-Transporten. Es verwaltet Waren in einem Backend-System, welches ein verteiltes Datenbanksystem verwendet. Mit mobilen Android-Geräten wird der Transport sowie die Lagerung von Zustellern sowie Lagermitarbeitern überwacht.

Transport von medizinischen Produkten (temperatursensitiv, Transportzeit, überwachbar, etc.).

2.1 Technologie

- Backend-/Application-Server
- Annahmestelle für Waren (Item)
- verteilte Datenbank (CouchDB)
- Android für mobile Überwachung der Sensoren/Position/etc.
- Sensoren
  - WLAN: Zugriff mit bspw. `http://192.168.47.11:1234/`
  - Bluetooth: TODO Matze
- Barcode-Lesegerät mit Android Devices
  - Barcode-Scanner App

    ```
    Activity.forResult(com.google.zxing.client.android.SCAN);
    ```

2.2 Fragestellungen

CouchDB für diese Anwendung sinnvoll, machbar? Ja.

Barcode mit Android-Device lesbar? Schnell genug? Ist ein Framework verfügbar?

Sensoren mit Bluetooth auslesbar? PAN? Alternativen (evt. WLAN/HTTP)?

3 Usecases

- Wareneingang

  registriert Pakete im System

  klebt QR-Code auf Pakete

- Logister plant und erstellt Lieferungen (neue Auftragsnummer wird generiert)

  wählt Pakete aus (können mit Sensoren bestückt sein)

  wählt Fahrer aus

  wählt Transportmittel (können mit Sensoren bestückt sein) für Lieferungen aus

  trägt Zielort und Auftraggeber ein

- Transporteur

  holt oder hat Device mit Roadrunner App

  loggt sich im Roadrunner System ein

  mit Benutzerdaten wird sein/e aktuelle/r Auftrag/Lieferung aufs Device synchronisiert ODER

  scannt Pakete und lädt sie in das vom Logistiker ausgewählte Transportmittel

Daten-Synchronisierung Vorbedingungen:

- Transporteur hat sich in der System-App eingeloggt

Die Daten-Synchronisierung oder Replizierung wird durch das Einloggen im System angestoßen. Das mobile Gerät erhält folgende Information:

- Adressen der Sensoren, die das Gerät überwachen sollte
- alle Produkte, Pakete der aktuellen Lieferung, sowie Zielort, etc.
- Überwachungs-Thresholds der Pakete

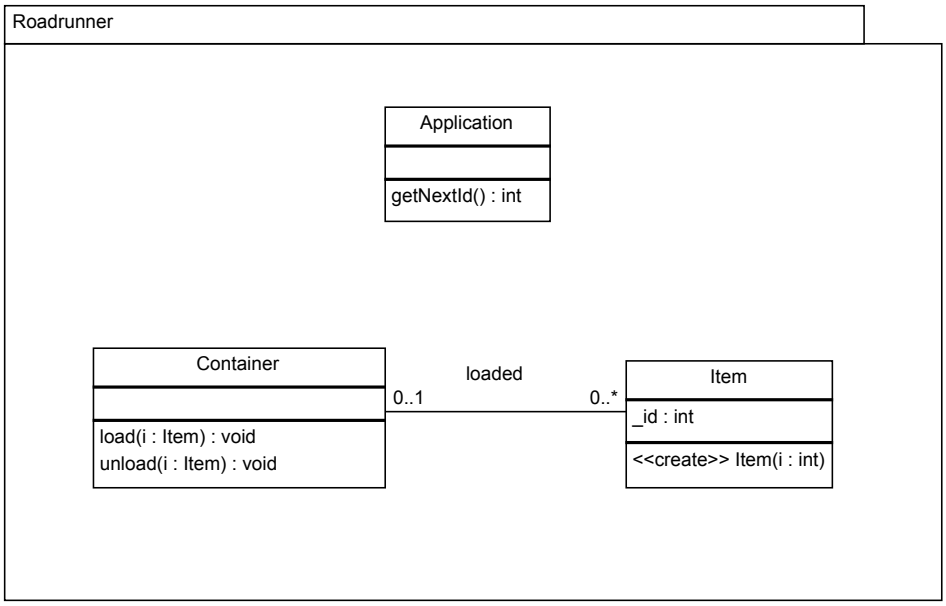

Abbildung 4.1: Iteration 1

4 Iteration 1

4.1 Ziele

- Produkte können erzeugt werden.
- Produkte können eingelagert werden.
- Produkte können ausgelagert werden.

4.2 UML

5 Sicherheit

5.1 Zeitsynchronisierung

In diesem Abschnitt werden Probleme besprochen, die durch fehlerhafte respektive mangelhaft durchdachte Zeitsynchronisierung oder Verbindungsabbruch entstehen können.

Problem durch falsche Zeitstempel bei Logeinträgen: Betrachtet wird das Szenario "Umladevorgang eines Produktes". Das mobile Gerät der Transporteinheit wird benutzt um den Ausladevorgang aus einem Container im System zu verarbeiten. Mit dem scannen des Produkts wird auf dem mobilen Gerät der Transporteinheit ein Logeintrag in dessen lokale Datenbank erstellt. Genauso wird beim darauffolgenden Ladevorgang der Umladestation ein Logeintrag auf dessen Gerät erstellt. Wenn das System mit absoluter Zeit arbeitet und die Uhrzeit des Geräts der Transporteinheit vor jener der Umladestation ist, dann würde im System der Übernahmevorgang der Umladestation vor dem Ausladevorgang der Transporteinheit stattfinden.

Lösungsansatz: Um dieses Problem zu lösen muss relative Zeit eingeführt und synchronisiert werden. Für die Zeitsynchronisierung können bekannte Algorithmen für verteilte Systeme eingeführt werden. Mögliche Algorithmen sind

TODO: UPV distributed Clocks .. algorithmen herausfinden und oben einfügen

Christian's Algorithm, Berkley Algorithm, $http://en.wikipedia.org/wiki/Clock_synchronization$

Grundsätzlich müssen diese Probleme berücksichtigt werden. In unserem Projekt werden die erwähnten Lösungen aus zeitlichen Gründen und anderer Zielsetzung nicht implementiert.

6 Sensorik

6.1 Was für Sensoren werden verwendet?

6.2 Wann und wie kommt der Sensor ins System?

6.3 Wie wird er überwacht und erreicht?

6.4 Wie werden dem Mobilen Device Sensoren zugeordnet?

6.5 Temperatursensoren

Hygrosens TLOG20-BLUE Das ist *NICHT* unsere Lösung. hygrosens.com/TLOG20-BLUE, Zugriff am 16.04.2011

Ampedrf BT11 Das ist *NICHT* unsere Lösung. BT11 Class1, Zugriff am 16.04.2011 BT11 Datasheet

$149 Programmable Universal Key Fob Sensor Wir haben uns für das BlueRadios BR-FOB-SEN-LE4.0 Device entschieden, weil es eine komplette und etablierte Lösung für Temperatur, Beschleunigungs- und Licht-Messung ist. Blueradios BR-FOB-SEN-LE4, Zugriff am 16.04.2011

Blueradios BR-FOB-SEN-LE4

6.6 Simulation

Wir haben uns in diesem Projekt entschieden, die Thematik Sensor vollständig zu simulieren. Einerseits aus zeitlichen Aspekten und andererseits, um unseren Hauptfokus intensiver ausarbeiten zu können.

6.7 Wie sieht unsere Sensorsimulation aus?

Alle benötigten Sensoren, Zeitsensoren sowie Temperatursensoren werden mit *nodejs* simuliert.

nodejs ist eine Server-seitige Javascript Umgebung, welche ein asynchrones Event-basiertes-Modell verwendet github.com/nodejs, Zugriff am 09.04.2011.

6.8 Warum nodejs?

Der Vorteil von *nodejs* liegt darin, dass sehr schnell, mit wenig Zeilen Code und Aufwand ein Http-Server programmiert werden kann.

6.9 Benötigte Packete

- nodejs (für Unix)
- npm (um Bilbliothek optimist komfortabel zu installieren)
- optimist (nodejs Library für Console-Parameter übergabe)