Datenreise

Idee

Um unsere Visibility in der Öffentlichkeit zu erhöhen entstand heute die Idee, eine Datenreise durch die gesamte Stadt zu realisieren. Dazu ist der Plan an verschiedenen Stellen in Münster kleine Stationen (Schaufenster, etc) aufzustellen, die auf möglichst unterhaltsame und anschauliche Weise Daten übertragen. Zum Beispiel könnte man dann ein Foto vom Hawerkamp, vom Hafen oder sonst was übertragen und zwar Bit für Bit, oder Byte für Byte. Ganz geil wäre noch, wenn die aktuellen Daten auf einem Bildschirm angezeigt werden.

Infrastruktur

Die Daten werden von eine zentralen Instanz bereitgestellt. Die Stationen holen sich die Daten übers Inet von einem Server, übertragen sie auf die lustige und anschauliche Art und schieben sie wieder auf den Server zurück. Dadurch kann man alle Stationen monitoren und man kann das System dynamisch erweitern oder Stationen wieder entfernen. Sogar eine Erweiterung in eine andere Stadt ist denkbar.

Zentrale Instanz

Auf dem Warpzone Server gibt es eine VM (datenreise.warpzone.ms), die Stand heute noch nix kann. Ich will aber mal beschreiben was sie können soll:

Frontend

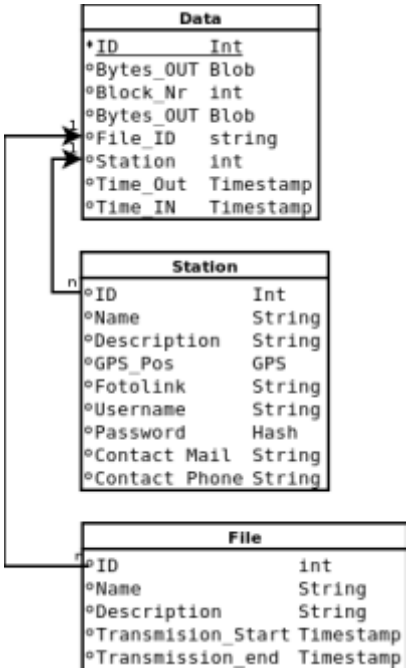
Das Frontend soll möglichst einfach sein, damit die Einstieghürde für potentielle Mitmacher möglichst gering ist. Die Schnittstelle soll möglichst vielseitig sein, aber auch ein Mindestmaß an Sicherheit bieten. Daher kommen für die Übertragung der Daten nur Protokolle in Frage, die eine Authentifizierung ermöglichen. Eine mögliche Schnittstelle wäre http mit nem PHP oder CGI im Hintergrund. Da http selbst zu proggen nicht ultraeingängig ist, könnte man noch ein MiniFTP entwickeln, der mit den Befehlen `auth`,`get` und `put` arbeitet. Der MiniFTPdemon könnte die Daten auch direkt aus der Datenbank holen. Mit dieser Vorgehensweise hat man folgende Probleme erschlagen:

- Dateinamen : Dateinamen sind immer eine Fehlerquelle bei Falschbenennung oder auch bei bereits vorhandenen Dateinamen. Dadurch kann es zu zahlreichen unvorhergesehenen Fehlern kommen und das Interface sollte ja einfach sein, gell?
- Timestamps: Für unsere statistische Auswertung müssen wir wissen, welche Station wie lange gebraucht hat. Einfach nur eine Datei hochladen, die dann von einem Prozess eingesammelt wird, der nur alle 15 Minuten läuft ist etwas ungenau. Dann könnte der Dateiname der Uploaddatei einen Timestamp enthalten. Der müsste aber korrekt im Layout und synchron zur zentralen Uhr sein. Zudem hat nicht jede Station überhaupt eine Uhrzeit da.
- Threadsicherheit: Die Threadsicherheit ist notwendig, da immer 2 Prozesse auf die Daten zugreifen. Erstens der Upload/Download der Station und zweitens Auswertungsprozesse auf der

zentralen Einheit. Wenn beide Prozesse gleichzeitig auf eine Datei zugreifen, bekommt der eine nur die Hälfte der Daten und die andere Hälfte landet im Nirvana. Konstrukte mit .lock Dateien oder ähnlichem sind für dieses Szenario ungeeignet bzw. zu komplex umzusetzen. Beim Schreiben in eine Datenbank achtet die Datenbank drauf, dass sich 2 verschiedene Prozesse ins Gehege kommen.

- Authentifizierung: Für einen Dienst, der im Internet hängt und dessen öffentlich Schnittstellen dokumentiert sind, braucht man eine Authentifizierung. Sie muss nicht besonders komplex sein, aber man soll es ja auch nicht zu einfach machen.
- Datenintegrität: Damit uns nicht irgendwelche Skriptkiddies Pornobilder in die Daten schieben, sollten wir eine Überprüfung der Daten vornehmen. Z.b. mit der Levenshtein-Distanz. Wenn diese Distanz zu groß ist, ist vielleicht Schluderei oder eine defekte Station im Spiel. Hier sollte man überlegen, ob man der Station nochmal dieselben Daten schickt, oder die Daten einfach benutzt? Wenn eine Station offensichtlich defekt ist, weil sie nur noch Quatsch schickt, könnte man sie auch aus dem Prozess rausnehmen und dem Admin Bescheid geben. Damit die Stationen nicht nur als Datensenke fungieren, sollte es neue Daten immer erst geben, wenn die letzten bereits wieder hochgeladen wurden.

Habe mal ein Datenbankschema gefummelt:



Webseite

Um das Projekt auch überregional bekannt zu machen brauchen wir natürlich eine Webseite. Dort wäre eine Karte nett, auf der alle Stationen verzeichnet sind. Bei Klick auf die Stationen kriegt man Infos über die Station:

- Foto
- Entstehungsgeschichte
- Funktionsweise
- aktuelle Bitfehler (Vergleich von ausgehenden und eingehenden Daten)
- aktueller Durchsatz
- ggf. Webcam

Zivilistenintegration

Natürlich wäre es cool, wenn die Nichtzonies auch was zum Projekt beitragen könnten. So könnten sie z.B. Daten zur Übertragung (Fotos, Texte) bereitstellen, oder eigene Stationen basteln, die dann ins Projekt integriert werden.

Brainstorm für Stationen

Digitale Übermittlung

Laser

2 Laser übertragen die Daten. Einer gibt den Takt auf einen Fotosensor und der andere wird angeschaltet, je nachdem ob 0 oder 1. Das coole ist hier, dass über 2 Laserpointer theoretisch Daten injectet werden können.

Murmelbahn

Im getakteten Rythmus rollt eine Murmel vorbei, oder halt nicht. Keine Kugel Bit 0 / Kugel Bit 1.

Farberkennung

Murmeln einer bestimmten Farbe rollen in eine Kammer und werden mit einer RGB LED und einem Fotosensor gescannt. Jede Farbe steht für ein Tupel.

Slow Ethernet

Die Daten werden nach dem Ethernetstandard übertragen, aber mit leuchtenden „Kabeln“ und halt nur mit 0.00001MBit, so dass man sieht, was passiert.

Fahnenschwenker

Eine Figur mit 2 schwenkbaren Armen steht auf einem Dach und bewegt je nach darzustellendem Byte die Arme ein. Eine Kamera auf einem anderen Hausdach erkennt die Armstellung und speist das Byte wieder ein.

Analoge Übermittlung

Wasserwaage

Je nach Bytewert wird eine bestimmte Menge Wasser in ein Wasserglas geschüttet und gewogen. Das Wiegeergebnis ist das übertragene Byte (bzw. 100% Füllstand = vollster Farbwert, 0%=schwarz). Danach muss das Wasser wieder zurückgepumpt werden.

Entfernung

Ein Hindernis wird auf einen Ultraschallsensor zu bewegt oder entfernt. Wenn es zum Stillstand kommt, ist die Entfernung zur Messeinheit das zu übertragende Byte. (Bzw. Volle Entfernung = 100% Farbwert, niedrigste 0%)

Alternative Übermittlung

Crypt/Decrypt

Die Daten werden verschlüsselt (XOR(Data,„Warpzone“), oder so) und schickt es an die nächste Station. Dort werden die Daten nur verschlüsselt dargestellt und übertragen. Die Station danach entschlüsselt wieder und die Daten sind wieder lesbar.