Richtlinie zum „Software-Sprint“

Dystonse - Wahrscheinlichkeits-basierte ÖPNV-Routensuche

Schlussbericht

Zuwendungsempfänger:

Dystonse - Lena Schimmel und Kirstin Rohwer GbR

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01IS19S23 gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt beim Autor.

Kurze Darstellung der Aufgabenstellung und Motivation

Bisher bestehende Werkzeuge zur Routen-Planung im ÖPNV und Fernverkehr arbeiten primär auf Fahrplandaten und beziehen mögliche und tatsächliche Verspätungen und Ausfälle allenfalls nachrangig mit ein. Dadurch entsteht bei den Reisenden ein Gefühl der Unsicherheit - Wird der Anschlusszug erreicht? Was, wenn nicht? - und die Abwägung, ob eine vorgeschlagene Route ausreichend sicher ist, muss rein subjektiv erfolgen.

Im Projekt “Dystonse” sollte eine Routen-Suche entwickelt werden, die durchgängig mit Wahrscheinlichkeitsverteilungen arbeitet und somit nicht nur Ergebnisse für den Best-Case liefert, sondern eine realistische Angabe für die Ankunfts-Zeiten und -Wahrscheinlichkeiten. Die inhärente Unvorhersehbarkeit des öffentlichen Nah- und Fernverkehrs sollte damit einschätz- und handhabbar werden.

Die geplante Software sollte aus der Such-Komponente (Backend) sowie einer Website und/oder App bestehen, um die Suche zu bedienen (Frontend).

Im Backend sollten Echtzeitdaten von Verkehrsmitteln gesammelt und ausgewertet werden. In die Suche sollten sowohl aktuelle Daten eingehen, als auch statistische Modelle aus vergangenen Daten.

Das Frontend sollte den Reisenden ähnliche Grundfunktionen bieten wie bestehende Nah- und Fernverkehrs- Apps, jedoch mit dem grundlegenden Unterschied, dass nicht nur feste Zeitangaben auf einer definitiven Route angezeigt werden, sondern die relevanten Alternativen und Eventualitäten.

Im Ergebnis sollte das Vertrauen und Sicherheitsgefühl der Reisenden vor und während der Reise erhöht werden. Bestehende Nutzer_innen des öffentlichen Verkehrs sollten in ihrer Nutzung bestärkt werden, und Menschen, die wegen der Unsicherheiten bisher nicht den öffentlichen Verkehr nutzen, sollten dazu animiert werden.

Um dies umzusetzen, wurden folgende Meilensteine geplant:

1. Kontinuierliche Verspätungs-Datensammlung planen, aufbauen und betreiben
2. Statistische Analyse der gesammelten Echtzeit-Verspätungs-Daten
3. Prognose-Berechnung entwickeln
4. Routensuche neu implementieren
5. Web-Frontend anpassen
6. Userbefragung, UX- und UI-Design
7. Weitere Datenquellen/Formate und Testgebiete hinzufügen, inkl. neuer Analyseschritte
8. eigene API anbieten
9. eigene App entwickeln oder dystonse in andere Apps integrieren
10. (ggf. weitere Milestones, die sich aus User-Befragung etc. ergeben)

Beitrag des Projektes zu den Zielen der Förderinitative „Software-Sprint“

Reisende im öffentlichen Personenverkehr (ÖPV) sollen letztlich die Software für die Planung ihrer Reisen nutzen und profitieren von zuverlässigeren Verbindungen. Sie sparen damit Zeit und/oder Nerven, da weniger Unsicherheit bzgl. des Reiseverlaufs besteht.

Der ÖPV soll dadurch attraktiver werden und idealerweise seinen Anteil im Modal Split der Verkehrsträger erhöhen. Davon profitieren die Verkehrsbetriebe, und durch die verbesserte Klimabilanz praktisch alle.

Dadurch, dass die Software unter einer freien Lizenz (MIT-Lizenz) veröffentlich wird, profitieren letztlich auch diejenigen, die ähnliche Software entwickeln und dabei auf unseren Beiträgen aufbauen können. Dies betrifft ebenso private (Hobby-)Entwicklerinnen wie auch Unternehmen.

Historisch bedingt wurden Nahverkehrsdaten oft nur angeboten, um diese unverändert bereitzustellen. Das Projekt Dystonse soll praktisch aufzeigen, wie freie Daten unter offenen Lizenzen darüber hinausgehende Anwendungen ermöglichen, und damit hoffentlich dazu beitragen, dass weitere Beteiligte ihre Daten ebenfalls freigeben.

Ausführliche Darstellung der Ergebnisse

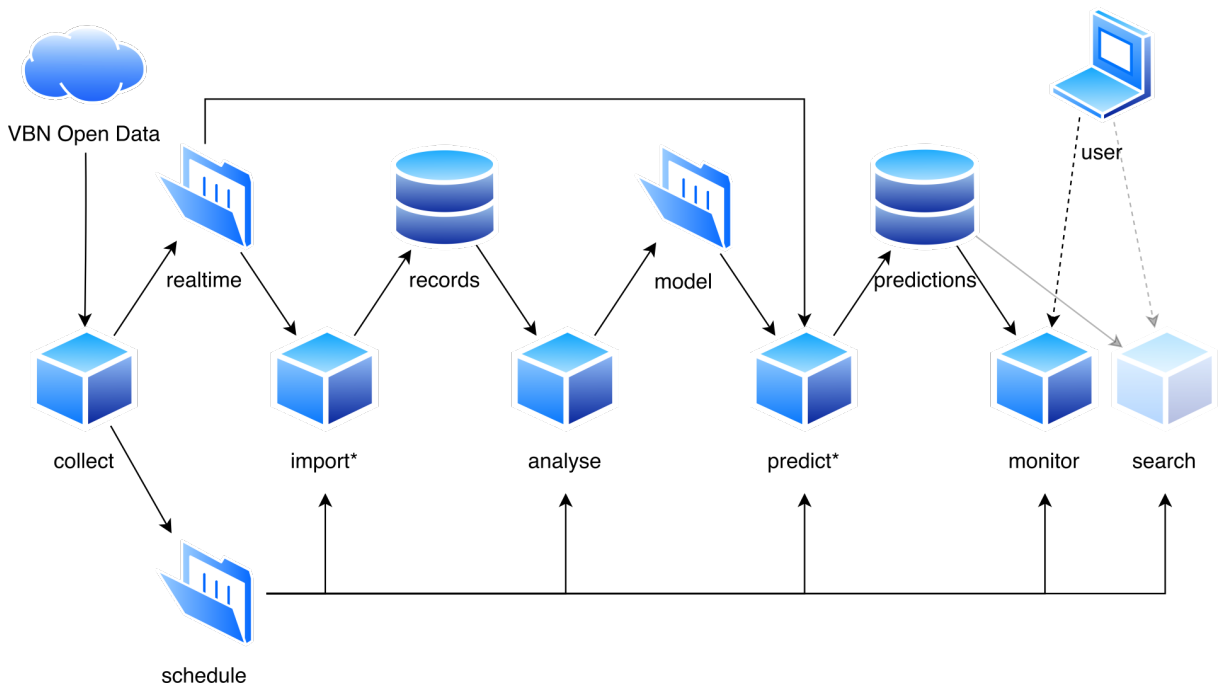
Angestrebt wurde die Entwicklung eine Backends und eines Frontends zur Routensuche. Zusammenfassend kann gesagt werden, dass das Backend erfolgreich umgesetzt wurde und seit Monaten im Dauerbetrieb funktioniert. Das Frontend wurde nicht in der ursprünglich geplanten Weise umgesetzt, sondern dessen Zweck und Funktionsweise angepasst. Im Rahmen dieser Planänderung ist auch das Frontend als erfolgreich umgesetzt anzusehen.

Die Gründe für die Neudefinition des Frontends sind im Abschnitt “Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung” erläutert, während die Funktion des Back- und Frontends weiter unten in diesem Abschnitt beschrieben sind.

Konkret konnten wir den folgenden Anteil der Meilensteine erreichen:

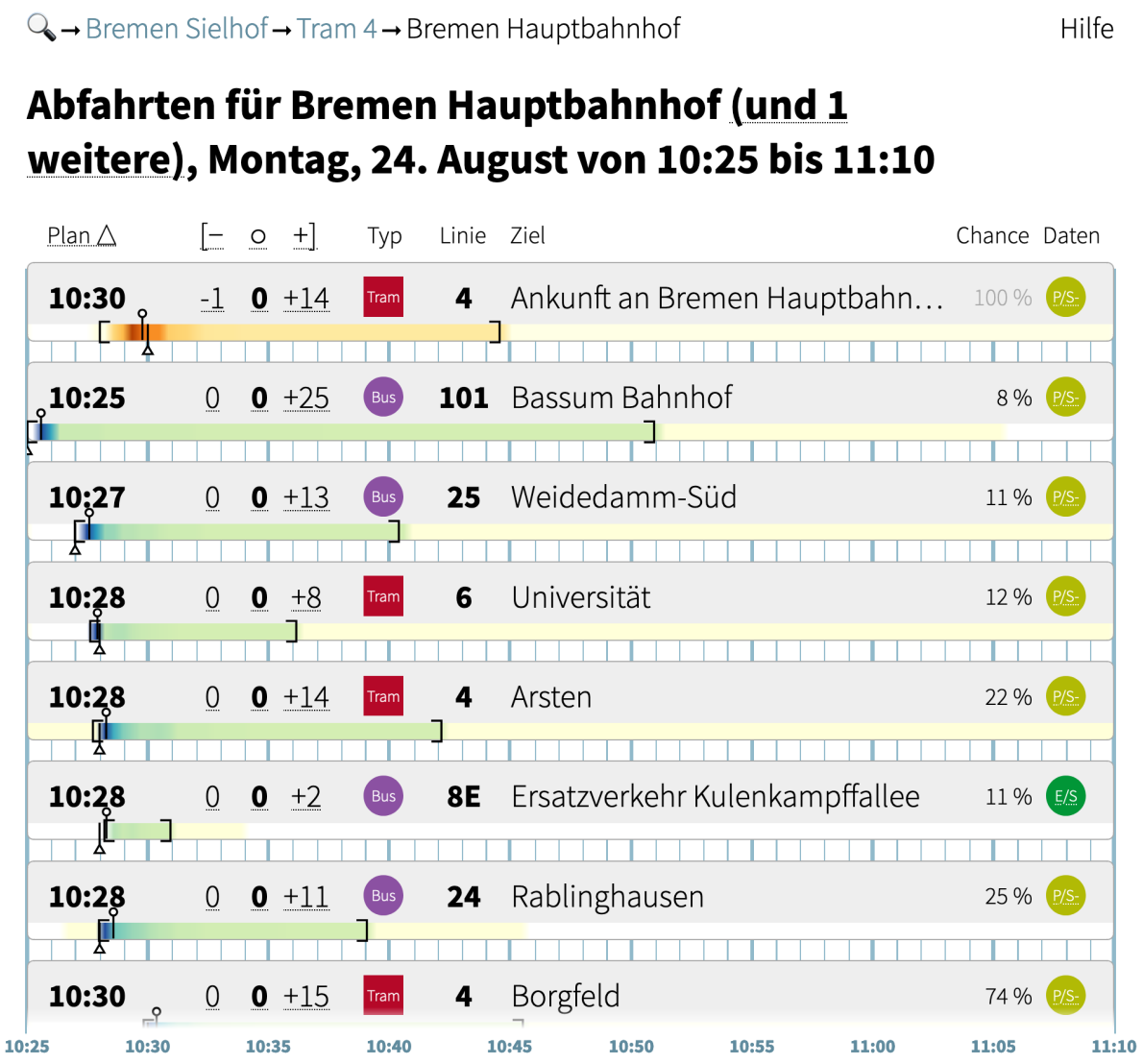
1. **erreicht:** Kontinuierliche Verspätungs-Datensammlung planen, aufbauen und betreiben
2. **erreicht:** Statistische Analyse der gesammelten Echtzeit-Verspätungs-Daten
3. **erreicht:** Prognose-Berechnung entwickeln
4. **nicht erreicht:** Routensuche neu implementieren
5. **teilweise erreicht:** Web-Frontend anpassen
6. **teilweise erreicht:** Userbefragung, UX- und UI-Design
7. **nicht erreicht:** Weitere Datenquellen/Formate und Testgebiete hinzufügen, inkl. neuer Analyseschritte
8. **teilweise erreicht:** eigene API anbieten
9. **nicht erreicht:** eigene App entwickeln oder dystonse in andere Apps integrieren
10. **nicht zutreffend:** (ggf. weitere Milestones, die sich aus User-Befragung etc. ergeben)

Über eine Vielzahl von Verarbeitungsschritten generiert das Backend fortlaufend Prognosen für Fahrzeuge des ÖPV, wobei für jedes künftige Abfahrts- oder Ankunftsereignis nicht nur eine konkrete Zeit berechnet wird, sondern die Wahrscheinlichkeitsverteilung über einen ausgedehnten Zeitraum. Dabei werden diese Verteilungen nicht auf statistische Grundparameter herunter gebrochen, sondern detailliert berechnet, gespeichert und übertragen, um damit im Nachgang komplexe Berechnungen und Visualisierungen zu ermöglichen.



Das Backend und die davon erzeugzen Daten eignen sich dabei gleichermaßen für die (noch nicht umgesetzte) Routensuche, als auch für den (ersatzweise umgesetzten) erweiterten Abfahrtsmonitor.

Statt einer vollständigen Routensuche wurde ein erweiterter Abfahrtsmonitor entwickelt, der quasi die Vorstufe einer Routensuche darstellt. Für einen beliebigen, von den Nutzer_innen gewählten Haltepunkt werden die Abfahrten der nächsten halben Stunde angezeigt. Dabei wird, entsprechend der vom Backend berechneten Wahrscheinlichkeitsverteilungen, nicht nur ein einzelner Zeitpunkt angezeigt, sondern die gesamte Verteilung. Die Darstellung umfasst absolute Zeiten laut Fahrplan, dazu relative Angaben für die früheste, wahrscheinlichste und späteste zu erwartende Abfahrtszeit. Diese vier Zeitpunkte werden durch Symbole auf einer Zeitskala dargestellt, die durch Helligkeit und Farbton außerdem die detaillierte Wahrscheinlichkeitsverteilung über den gesamten Zeitraum anzeigt.



Zusätzlich zum bloßen Betrachten der Abfahrtszeiten können Reisende mit wenigen Klicks manuell eine Reisekette zusammen stellen. Für jede Abfahrt kann der weitere Fahrtverlauf gezeigt werden, wobei alle folgende Halte ebenfalls eine Verteilung möglichen der Ankunftszeiten zeigen. Mit einem weiteren Klick kann ein Ausstieg geplant werden, wobei alle dort verfügbaren Umstiege gelistet werden. Aus den Ankunfts- und Abfahrtswahrscheinlichkeiten wird automatisch die Chance errechnet, dass der jeweilige Anschluss erreicht wird. Bei längeren Reiseketten werden diese Chancen multipliziert, um die Gesamtchance zu zeigen, mit der die Reise planmäßig vollendet werden kann.

Für mögliche Umstiege werden auch Abfahrten von nahegelegenen Haltepunkten mit einbezogen. Die nötigen Fußwege werden in die Zeitplanung mit einberechnet. Auch hier wird statt einer Konstanten eine Wahrscheinlichkeitsverteilung von Reisezeiten errechnet, bei der z.B. verschiedene Geh-Geschwindigkeiten und Faktoren für nötige Umwege eingerechnet werden.

Das Monitor-Frontend basiert auf dynamisch generierten HTML-Seiten. Diese wurden im Hinblick auf Effizienz, Kompatibilität, Barrierefreiheit und Datenschutz komplett ohne Javascript, Sessions, Cookies, etc. umgesetzt.

Die gesamte Software wurde in der Programmiersprache Rust umgesetzt. Das Bauen und Betreiben der Software wurde mittels Docker und Docker-Compose automatisiert, so dass interessierte Personen mit möglichst geringem Aufwand eine eigene Instanz aufsetzen können.

Am Rande der Entwicklung sind kleinere Nebenergebnisse entstanden, wie z.B. eine Übersicht[1] über die Versorgungslage mit offenen ÖPV-Echtzeitdaten und ein Werkzeug[2] zur Sammlung der Angaben über Verkehrsbetriebe in Deutschland.

Zielgruppe, Nutzen und mögliche Weiterentwicklungen

Durch die Anpassung des Entwicklungsziels - Abfahrtsmonitor statt Routensuche - wurde sichergestellt, dass am Ende der Förderzeit bereits eine Software besteht, die für interessierte Endanwender_innen, hier Reisende im ÖPV, nutzbar und nützlich ist. Sie können den Monitor bei der Planung und Ausführung von Reisen im ÖPV nutzen, und darüber hinaus ihr “individuelles, intuitives Verspätungsmodell” mit den Statistiken abgleichen. Der Monitor kann auf Desktop- und Mobilgeräten genutzt werden.

Für andere Software-Entwickler_innen bietet unsere Software mehrere Anknüpfungspunkte. Andere Projekte, die ebenfalls aufgezeichnete Verspätungsdaten, daraus abgeleitere Verspätungsmodelle und/oder Prognosen benötigen, können eine eigene Instanz von Dystonse betreiben und z.B. auf Datei- oder Datenbank-Ebene auf die Ergebnisse zugreifen. Die ständig aktualisierten Prognosen sollen zudem in Kürze als GTFS-Realtime-Feed bereitgestellt werden. Damit können diese über das Internet abgerufen werden, ohne eine eigene Instanz von Dystonse zu betreiben.

Wir sind nach wie vor daran interessiert, die ursprünglich geplante Routensuche umzusetzen. Wir versuchen derzeit, einen organisatorischen und finanziellen Rahmen dafür zu finden, idealerweise durch eine Beauftragung durch oder Kooperation mit einem Verkehrsbetrieb.

Für das Backend und den Abfahrtsmonitor gibt es eine Reihe von Ideen zur Weiterentwicklung:

- Bewertungsfunktion für Routen, die mit anderer Software ermittelt wurden, und Integration dieser Bewertungen in vorhandene Apps und Websites
- Werkzeug zur Fahrplanoptimierung, welches Hotspots für häufige Betriebsverzögerungen findet und die realen Auswirkungen auf Umstiege bemisst
- Anbindung weiterer Formate und APIs, Interoperabilität zwischen Datensätzen (ggf. als eigenständiges Paket/Projekt)

---

[1] https://github.com/dystonse/dystonse/blob/master/project-status/Datenquellen.md

[2] https://github.com/dystonse/dystonse-tools/tree/master/agencies

- Einbindung von OpenStreetMap und Pfädle[3], um die konkreten Fahrtverläufe darstellen zu können
- Fußweg-Routing und exaktere Umsteige-Zeiten (basierend auf OSM und/oder dem Graphenmodell aus GTFS[4])

Für uns persönlich und als Gruppe hat das Projekt ebenfalls positive Auswirkungen:

Zu Umsetzung von Dystonse haben wir die Programmiersprache Rust neu erlernt, die sich als exzellente Wahl für dieses Projekt heraus gestellt hat und sicher auch bei künftigen Projekten zum Einsatz kommen wird.

Wir konnten unsere Verbindungen in die Open-Source-, Open-Data- und ÖPNV-Communities ausbauen. Durch unsere eigene Recherche und den Austausch mit eben diesen Communities haben wir nun einen besseren Überblick darüber, welche Probleme und Lösungen es in diesen Gebieten gibt und was aktuelle Entwicklungen sind.

Auf der persönlichen Ebene konnten wir außergewöhnlich positive Erfahrung mit der Teamarbeit im gemeinsamen Home Office machen und unsere Produktivität und Organisations-Skills deutlich verbessern, trotz der bedrückenden globalen Lage in Zeiten der Pandemie.

Kurze Darstellung der Arbeiten, die zu keiner Lösung geführt haben

Anfangs haben wir versucht, die Fahrplandaten, die uns im GTFS-Format geliefert werden, in eine relationale Datenbank zu importieren. Vorhandene Lösungen waren zu unperformant, und anstatt eine eigene Implementierung zu entwickeln, haben wir stattdessen eine Library ausgewählt, die Fahrpläne aus GTFS-Dateien in den RAM lädt und in Rust-Programmen effizienter und direkter nutzbar macht.

Fahrplan-Daten werden von den Verkehrsbetrieben bereitgestellt und häufig verändert. Unsere Aufzeichnungen ergeben, dass mindestens täglich Änderungen an den Daten vorgenommen werden. Für unsere Arbeit erschien es wichtig, eine Vorstellung davon zu bekommen, welcher Art und Umfang diese täglichen Änderungen sind. Wir haben Versuche unternommen, um GTFS-Daten in git-Repositories zu versionieren. Neben der Analysierbarkeit mit gängigen git-Werkzeugen erhofften wir uns außerdem Einsparungen bei der Archivierung der Daten. Es stellte sich heraus, dass Fahrplandaten schon für einen einzigen Verkehrsverbund zu umfangreich sind, d.h. dass git nicht in der Lage ist, csv-Dateien mit der dort zu erwartenden Menge an Zeilen absturzfrei zu verarbeiten.

Ein weiterer fruchtloser Versuch war der Betrieb der Dystonse Software auf CPUs mit ARM-Architektur. Zwar gelten alle Komponenten von Dystonse als portabel und auf ARM einsetzbar, doch die Integration der Komponenten mittels Docker scheitert an Programmfehlern. Die Fehler liegen im Bereich von Docker und/oder dem Linux-Kernel und sind dort bekannt und dokumentiert.

Und schließlich wurde die Routensuche vorerst verworfen, um schneller zu einer benutzbaren Software für Reisende zu kommen.

---

[3] https://github.com/ad-freiburg/pfaedle

[4] https://developers.google.com/transit/gtfs/reference#pathwaystxt

Kurze Angabe von Präsentationsmöglichkeiten für mögliche Nutzer

Der erweiterte Abfahrtsmonitor ist unter https://monitor.dystonse.org erreichbar und funktioniert sowohl auf Desktop- als auch auf Mobilbrowsern. Eine ausführliche Dokumentation für Nutzer_innen ist über den Link Hilfe[5] ebenso erreichbar.

Das Projekt wurde, wie alle Projekte der 7. Prototype-Fund-Förderrunde, durch einen Artikel[6] auf der Website des Prototype Fund vorgestellt.

Technische Hintergründe und Berichte aus der Entwicklungszeit haben wir außerdem in unserem eigenen Blog[7] beschrieben. Die wöchentlichen Statusberichte, die im Rahmen der Förderung entstanden sind, sind ebenfalls online verfügbar[8] und werden vorerst über den Förderzeitraum hinaus weiter gepflegt.

Informationen zum Bauen und Betreiben einer eigenen Software-Instanz finden sich in den Readme-Dateien unserer GitHub-Repositories, insbesondere in dystonse-docker[9].

Kurze Erläuterung zur Einhaltung der Arbeits- und Kostenplanung

Die ersten Schritte der Entwicklung, vor allem die Suche nach geeigneten Datenquellen und die Implementierung der Datensammlung in Verbindung mit einer Datenbank, stellten sich als zeitaufwändiger heraus, als vorher geplant.

Zusätzlich hat sich die angespannte Lage durch die Coronavirus-Pandemie im April und Mai deutlich auf unsere generelle Produktivität ausgewirkt, so dass wir in den ersten Monaten deutlich langsamer voran kamen, als ursprünglich geplant.

Als absehbar war, dass die Abschlussveranstaltung (Demo Day) nicht vor Ort stattfinden kann, wurde stattdessen das Konzept “Demo Week” als Online-Format geplant – das brachte unter anderem mit sich, dass viele Ergebnisse schon deutlich früher als üblich fertig sein sollten, um in der Präsentation an das neue online-Format angepasst zu werden. Daher haben wir uns dann entschieden, zunächst eine andere Frontend-Variante vorzuziehen, da diese besser planbar war als die eigentliche Routensuche. So entstand der interaktive, erweiterte Abfahrtsmonitor, der alle Elemente der Wahrscheinlichkeitsbasierten Routensuche visualisiert, aber die Suche nicht automatisch durchführt, sondern den Nutzenden die Möglichkeit gibt, sich selbst eine Reisekette zusammenzustellen. Die Entwickung der voll-automatischen Routensuche haben wir stattdessen auf einen späteren Zeitpunkt verschoben.

Die Entwicklung des erweiterten Abfahrtsmonitors ging erfreulicherweise schneller als zunächst erwartet, so dass wir innerhalb eines Monats sogar viele optionale Features einbauen konnten.

---

5 https://monitor.dystonse.org/help

6 https://demoweek.prototypefund.de/projects/02-dystonse.html

7 https://blog.dystonse.org/

8 https://github.com/dystonse/dystonse/tree/master/project-status/Montagsupdates.md

9 https://github.com/dystonse/dystonse-docker

Kurze Darstellung von etwaigen Ergebnissen bei anderen Stellen

Gegen Ende der Förderzeit wurden wir auf die Dissertation[10] von Dr. Mohammad Hossein Keyhani mit dem Titel “Computing Highly Reliable Train Journeys” aufmerksam gemacht. Darin wird ein Ansatz beschrieben, der dem von Dystonse sehr ähnlich ist. Der Ansatz wurde als Erweiterung von MOTIS[11] entwickelt. Zwar ist MOTIS seit April 2020 unter der MIT-Lizenz verfügbar, das betrifft jedoch nicht die Teile aus der Arbeit von Keyhani.

Die wahrscheinlichkeitsbasierte Routensuche in Dystonse umzusetzen bleibt damit weiterhin relevant. Bevor dies geschieht, werden wir die genannte Quelle genau prüfen, um nützliche Ansätze zu übernehmen.

[10] https://tuprints.ulb.tu-darmstadt.de/6227/

[11] https://github.com/motis-project/motis