puenktlichkeit.ch

Publikation von Pünktlichkeits-Auswertungen zum Öffentlichen Verkehr auf Basis von Open Data

Semesterarbeit

Studiengang:	CAS Business Intelligence, HS 16/17
Autor:	Andreas Gutweniger
Betreuer:	Arno Schmidhauser
Datum:	10. April 2017

Berner Fachhochschule | Haute école spécialisée bernoise | Bern University of Applied Sciences

Management Summary

Eine hohe Pünktlichkeit gehört zu den wichtigsten Erfolgsfaktoren des öffentlichen Verkehrs. Umso mehr erstaunt es, dass in der Schweiz bisher keine branchenweite und systematische Pünktlichkeitsstatistik veröffentlicht wird. Seit Dezember 2016 stellen jedoch zahlreiche Unternehmen der öV-Branche ihre Fahrpläne und Betriebsdaten als «Open Data» zur Verfügung.

Gegenstand der vorliegenden Arbeit ist es, auf Basis dieser Daten Pünktlichkeits-Auswertungen im Internet zu publizieren. Wichtige Ziele sind rasche Umsetzbarkeit, leichte Anpassbarkeit sowie niedrige Betriebskosten. Es wird gezeigt, wie ein solches System unter Verwendung von R Shiny und MySQL in der Cloud erstellt werden kann und welche Architektur dabei zielführend ist. Die Arbeit setzt sich mit den verfügbaren Quelldaten auseinander und geht ausführlich auf zu verwendende Datenmodelle und ETL-Prozesse ein. Exemplarisch werden mehrere Auswertungstypen konzipiert.

Der Grossteil der beschriebenen Konzepte, Modelle und Visualisierungen wurde in einem Prototypen implementiert, welcher im Web unter *puenktlichkeit.ch* abrufbar ist. Die dabei gewonnenen Erkenntnisse zu Open Data, Architektur, Datenmodellen, Prozessen, Technologien und Werkzeugen werden erörtert. Die Arbeit schliesst mit einem Ausblick auf sinnvolle weitere Features, den notwendigen Einbezug der Anwender und die Verwendung des Datenbestands für Erklärungs- und Prognosemodelle.

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Zu den wichtigsten Erfolgsfaktoren des öffentlichen Verkehrs gehört eine hohe Pünktlichkeit. Diverse Anspruchsgruppen haben folglich ein Interesse daran, die im täglichen Betrieb erzielten Pünktlichkeitswerte erfahren, vergleichen und analysieren zu können: Mitarbeiter der Verkehrsunternehmen, Fahrgäste, Behörden, Medienschaffende und Politiker. Umso erstaunlicher ist es, dass in der Schweiz bislang keine umfassende Pünktlichkeitsstatistik publiziert wird: Zwar veröffentlichen viele der ca. 250 Verkehrsunternehmen aggregierte Einzelwerte (z.B. „Jahrespünktlichkeit"), es existiert aber weder eine öffentliche landesweite Zusammenstellung, noch lassen sich die Werte zeitnah (z.B. Pünktlichkeit der letzten Woche) oder im Detail (z.B. Pünktlichkeit einer bestimmten Linie) einsehen.

Im Sinne der „Open Data Strategie" des Bundesrats hat das Bundesamt für Verkehr vor einiger Zeit den Auftrag erteilt, Daten der Schweizer Verkehrsunternehmen öffentlich bereit zu stellen. Diese Daten sollen es auch branchenfremden Akteuren ermöglichen, neue Systeme, Apps und Statistiken zum öffentlichen Verkehr zu entwickeln.[1] Erfahrungen im Ausland (z.B. in London) haben gezeigt, dass ein solcher Open Data-Ansatz hohe Innovationskraft freisetzen und zugleich die Verkehrsunternehmen von der Entwicklung eigener Informationssysteme entlasten kann.[2]

Seit Dezember 2016 steht die beauftragte Plattform *www.opentransportdata.swiss* bereit. In einer ersten Ausbaustufe sind dort u.a. die geplanten (= publizierter Fahrplan) und tatsächlichen Ankunfts- und Abfahrtszeiten (= Verkehrslage) diverser Verkehrsunternehmen abrufbar und dürfen frei genutzt werden. Damit ist die Basis für die Erstellung öffentlicher Pünktlichkeitsstatistiken gegeben.

1.2 Ziele

Unter Verwendung der Daten von *www.opentransportdata.swiss* sollen grafische und tabellarische Auswertungen zur Pünktlichkeit im öffentlichen Verkehr der Schweiz produziert und im Internet publiziert werden.

Im Rahmen der Semesterarbeit sind folgende Ergebnisse zu erstellen:

- Gesamtarchitektur der Lösung inklusive der grundlegenden Technologie-Entscheide,
- Datenmodell für das zu verwendende Data Warehouse und / oder den Data Mart,
- Beschreibung der wichtigsten ETL-Prozesse,
- Konzeption ausgewählter tabellarischer und grafischer Auswertungen,
- Prototypische Umsetzung,
- Diskussion der gesammelten Erfahrungen,
- Ausblick auf sinnvolle Weiterentwicklungen.

Die Ergebnisse sollen sich dabei an folgenden Zielsetzungen orientieren:

1. Die Arbeit soll zeigen, wie mit geringem Aufwand und in kurzer Zeit ein nutzenstiftendes System erstellt werden kann. Der Fokus liegt auf „Time-to-Market"; eine schlanke Lösung wird angestrebt.
2. Die Auswertungen sollen die Breite dessen aufzeigen, was mit den vorhandenen Daten und eingesetzten Technologien in begrenzter Zeit machbar ist. Die Exploration der Möglichkeiten

[1] Vgl. BAV (2016).

[2] Vgl. Gerny (2016).

steht im Vordergrund; weder Vollständigkeit, noch Anschlussfähigkeit / formale Konsistenz der Auswertungen untereinander ist erforderlich.

3. Die Auswertungs-/Analyse-Schicht soll leicht anpassbar und erweiterbar sein. Auf Seite der Datenquellen wird dagegen eine hohe Stabilität unterstellt.
4. Lizenz- und Betriebskosten sollen möglichst niedrig sein; es besteht eine Präferenz für den Einsatz frei verfügbarer Technologien.

1.3 Vorgehen

Die Arbeit deckt die gesamte Breite eines BI-Projekts von Beschaffung und Analyse der Daten, über Transformation und Bereitstellung bis zur Erstellung und Publikation der Auswertungen ab. Um den Rahmen einer Semesterarbeit einhalten zu können, sind verschiedene Einschränkungen erforderlich:

- Die Rohdaten werden nahezu unreflektiert übernommen: Zwar werden offensichtliche Datenfehler bereinigt, eine weitergehende Qualitätsprüfung und -sicherung erfolgt aber nicht. Auch erfolgt keine Auseinandersetzung mit methodischen Fragen der Datenentstehung, systematischen Verzerrungen etc.. Die Arbeit geht von der Annahme aus, dass die Daten „die Wahrheit" über die Pünktlichkeit im öffentlichen Verkehr uneingeschränkt wiedergeben.
- Alle Analysen sind auf die verfügbaren Daten beschränkt (insbesondere: betrachteter Zeitraum und betrachtete Verkehrsunternehmen).
- Es werden keine weiteren Datenquellen hinzugezogen.
- ETL-Prozesse werden nur soweit umgesetzt, wie dies für nachfolgende Verarbeitungsschritte zwingend notwendig ist. Die Dimensionen werden als stabil angesehen (= Slowly Changing Dimensions Typ 0, «SCD0»).
- Es erfolgt kein Einbezug von Stakeholdern und es gibt keine explizite Anforderungserhebung und -dokumentation. Design-Entscheide orientieren sich allein an „common sense" und dem Branchenwissen des Autors aus früheren beruflichen Tätigkeiten.
- Die Anzahl der Auswertungstypen ist beschränkt. Eine breite Abdeckung von Zielgruppen und Bedürfnissen wird nicht angestrebt.
- Die System-Performance orientiert sich an den Erwartungen eines Prototyp-/Showcase-Szenarios.

1.4 Bekannte Vorarbeiten

Eine systematische Literaturrecherche wird hier nicht geleistet. An dieser Stelle sollen aber die dem Verfasser bekannten Arbeiten zum Thema aufgeführt werden:

- Das im Umfeld einer Dissertation[3] entstandene System «Open Timetable» wird bei der SBB seit über 10 Jahren zur Analyse des Betriebsablaufs eingesetzt.
- Die Verkehrsbetriebe Zürich (VBZ) veröffentlichen bereits seit Ende 2015 ihre Pünktlichkeitsdaten als «Open Data» - und sind damit Vorreiter in der Schweiz. «Tagesanzeiger» und «Bund» haben eine Analyse dieser Daten mit grafischen Auswertungen veröffentlicht.[4]
- Im Rahmen eines Hackathons in Zürich am Open Data Day 2017 beschäftigten sich mehrere Projekte ebenfalls mit der Auswertung der VBZ-Daten.[5]
- Statistiken zu ausgewählten deutschen Zuglinien sind auf www.zugfinder.de zu finden. Die Daten werden mittels Web Scraping aus dem Online-Fahrplan der Deutschen Bahn gewonnen.[6]

[3] Vgl. Ullius (2005).

[4] Vgl. Tagesanzeiger (2016).

[5] Eine Übersicht der Projekte findet sich unter https://hack.opendata.ch/event/7 .

2 Architektur der Lösung

Die in der Einleitung genannten Architekturziele (Time-to-market, Exploration von Möglichkeiten, Erweiterbarkeit, Kosten) bilden die Leitplanken für die Entwicklung des Prototypen.

Es hat sich bewährt, die Architektur von IT-Systemen multiperspektivisch, d.h. unter Einnahme unterschiedlicher Sichtweisen, zu beschreiben. Einen guten Rahmen dafür bietet das arc24-Framework[7]. Nachstehend wird eine Auswahl der dort aufgeführten Sichten verwendet.

2.1 Bausteinsicht: Komponenten der Lösung

Um Auswertungen zur Pünktlichkeit erstellen und im Web verfügbar machen zu können, werden folgende Komponenten benötigt:

- **Auswertungs-Applikation** zur Berechnung von Pünktlichkeitswerten und für deren grafische und tabellarische Aufbereitung,
- **Datenbanksystem** zur Sammlung und Bereitstellung der erforderlichen Daten,
- **Datenquelle** mit Angaben zu Soll-Fahrplan und aufgetretenen Verspätungen,
- **Scheduler** zum zeitgesteuerten Anstoss von Datenbezug und -transformation,
- **Applikations-Server und Plattform**, auf denen die Applikation betrieben werden kann,
- **Web-Server** zur Bereitstellung und zum interaktiven Abruf der Auswertungen im Internet.

Für Entwicklung, Pflege und Betrieb der Lösung sind zusätzlich erforderlich:

- **Entwicklungsumgebung** mit Editor, Compiler / Interpreter, Debugger etc.,
- Geeignete **Libraries**, um gängige Funktionsbausteine nicht selbst entwickeln zu müssen (z.B. Datentransformation, DB-Zugriff, Diagrammerstellung),
- **Code- und Versionsverwaltung**,
- **Administrations- und Monitoringwerkzeuge** für Server und Datenbanksystem,
- **Logging-Komponente** zwecks Kontrolle der Systemausführung.

Im Rahmen dieser Semesterarbeit erfolgt keine Anforderungserhebung mit potentiellen Systemnutzern. Für etwaige spätere Ausbauschritte ist diese aber zwingend erforderlich. Aus diesem Grund wurden von vornherein Möglichkeiten für die Beziehungspflege mit den Systemnutzern geschaffen:

- Einprägsame **Web-Adressen** (puenktlichkeit.de / pünktlichkeit.ch),
- **E-Mail-Kontaktmöglichkeit** (andreas.gutweniger@*puenktlichkeit.ch*),
- **Elektronischer Newsletter**,
- Möglichkeit zur **Zugriffsbeschränkung und -verwaltung** (z.B. um nutzerspezifische Funktionalität realisieren zu können),
- **Analysewerkzeuge** zur Bestimmung von Zahl, Profil und Verhaltensweisen der Anwender.

2.2 Anbieter- und Technologieauswahl

Die Auswahl von Technologien und Anbietern folgt dem Grundsatz «Open Source First, Cloud First»:

Da viele der zu lösenden Aufgaben für den Verfasser neu sind, bildet eine breite und gut funktionierende Support-Community das wichtigste Kriterium bei der Auswahl der Technologien. Die Wahl fiel

---

[6] Vgl. Schubert (2017).

[7] Vgl. Starke, Hruschka (2011).

auf R als Programmiersprache, RStudio als Entwicklungsumgebung, Shiny als Web-Applikationsserver, Ubuntu als Betriebssystem und MySQL für die Datenbank.

Um rasch starten zu können und Kosten niedrig und flexibel zu halten, werden fast ausschliesslich Cloud-Services verwendet. Nach kurzer Evaluation der bekannten Anbieter fiel die Wahl auf AWS (Amazon Web Services). Ausschlaggebend waren positive Erfahrungsberichte zum Betrieb von Shiny / R Server, europäisches Hosting (Frankfurt), das breite Service-Angebot und das im ersten Jahr (nahezu) kostenlose Einstiegs-Angebot.

Der Betrieb der Applikation auf shinyapps.io wurde verworfen, weil das Lizenzmodell im Hinblick auf einen späteren Ausbau nachteilig ist und nur wenig Funktionalität für Entwicklung und Administration angeboten wird. Zudem hätte der Datenbankserver geographisch und topologisch getrennt von der Applikation betrieben werden müssen - mit negativen Auswirkungen auf die Latenz. Stattdessen wurde ein eigener Shiny-Server auf einer AWS EC2-Instanz installiert[8], ebenso die Entwicklungsumgebung (R Studio Server), und der Proxy-Server (nginx). Der Scheduler (cron) ist Bestandteil des Betriebssystems. Der Datenbankserver (MySQL) befindet sich auf einem benachbarten Knoten (AWS RDS).

Bewusst wird auf den Einsatz eines separaten ETL-Tools verzichtet, weil mit SQL und R bereits leistungsfähige Werkzeuge für die Datenmanipulation zum Einsatz kommen. Der Zusatznutzen eines spezialisierten Produkts wird als sehr gering erachtetet gegenüber dem Komplexitätsanstieg und dem Zusatzaufwand für die Installation und Einarbeitung. Stattdessen ist die ETL-Steuerung in R-Skripten implementiert, inklusive der von dort aus abgesetzten SQL-Befehle.

Als Cloud Services bezogen werden weiterhin die Nutzeranalyse (Google Analytics), Newsletter (Mailchimp), Versionsverwaltung (Git auf Atlassian Bitbucket), E-Mail- und DNS-Adressen.

2.3 Verteilungssicht

Auf der nächsten Seite wird die Verteilung der zuvor beschriebenen Komponenten dargestellt. Jene Elemente, die ausschliesslich für Entwicklungs-, Test- und Administrationszwecke benötigt werden, sind farblich abgeschwächt gegenüber den Kernbestandteilen.

Der primäre Anwendungsfall besteht darin, dass ein Nutzer von einem Web Client aus die Adresse *puenktlichkeit.ch* aufruft. Diese führt auf eine AWS-Server-Instanz. Die Anfrage wird vom nginx-Proxy entgegengenommen und an die Applikation «puenktlichkeit» auf dem Shiny Server weitergeleitet. Diese ruft die benötigten Daten vom Schema «balu1» auf dem Datenbankserver ab. Zeitgesteuerte ETL-Skripte stellen sicher, dass die Datenbank regelmässig neue Daten vom externen Anbieter *opentransportdata.swiss* bezieht.

Das Gros der Entwicklungstätigkeit erfolgt ebenfalls Browser-basiert mit Zugriff auf R Studio Server. Alternativ können Teile des R Codes auch lokal entwickelt und getestet werden (inkl. Shiny auf localhost). Eine Synchronisation des Quellcodes erfolgt mittels Git über Atlassian Bitbucket. Für den Zugriff auf den Datenbankserver wird die MySQL-Workbench verwendet.

Die Verwendung von zwei getrennten Cloud Instanzen (Virtual Machines) für Datenbankserver einerseits und Web/Applikations-Server andererseits führt zu einer Verteilung der Last. Dies erhöht die Leistungsfähigkeit des Gesamtsystems und verhindert, dass Benutzerinteraktionen durch aufwendige DB-Operationen beeinträchtigt werden. Im Fall anwachsender Last erlaubt diese Aufteilung zudem eine effizientere Skalierung durch gezielte Redimensionierung der jeweiligen Engpassressource.

[8] Die Installation und Konfiguration folgte grossenteils den guten Anleitungen von Gritts (2016) und Banner (2016).

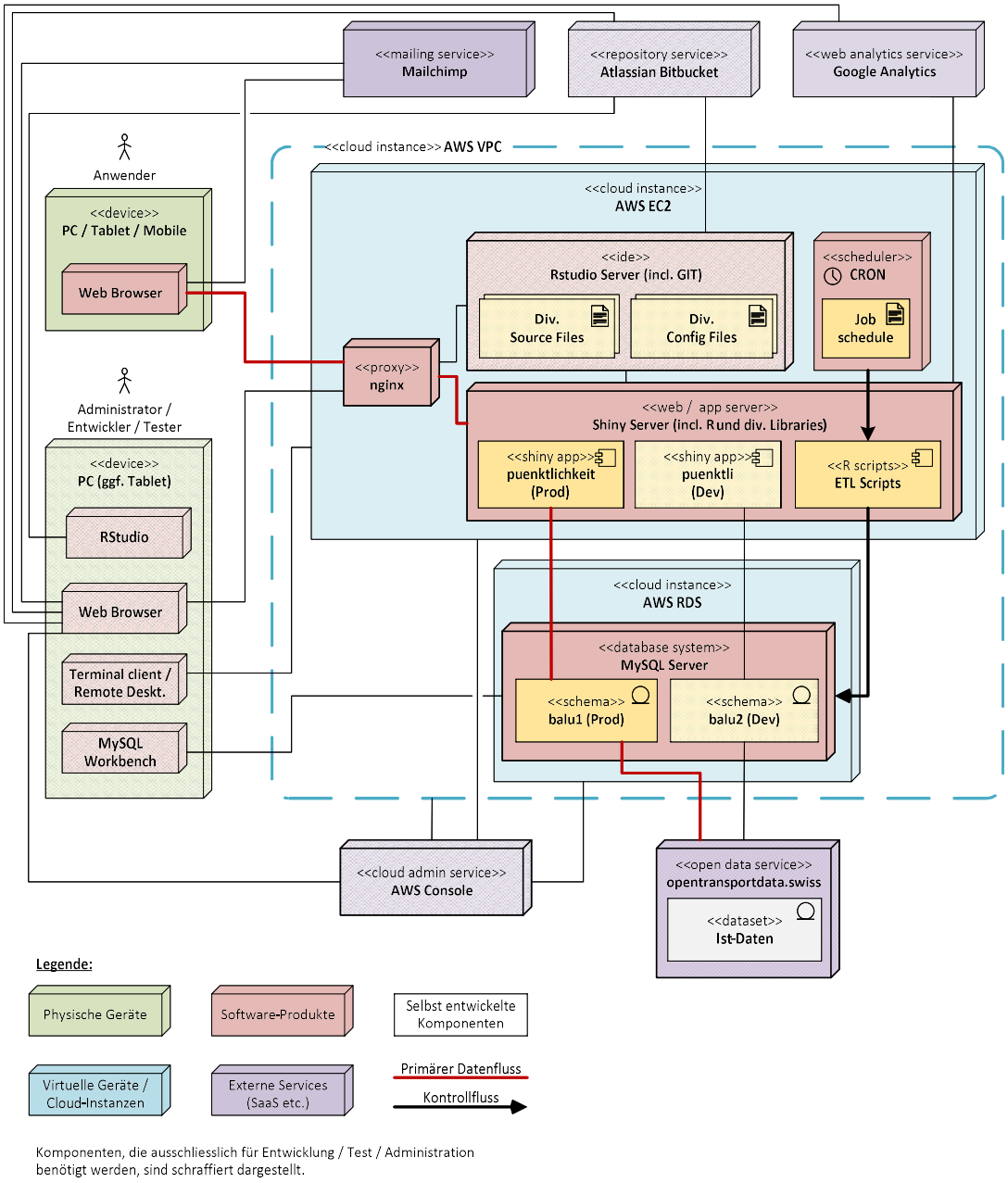

Abbildung 1: Architekturübersicht (Deployment-Diagramm)

2.4 Software-Design-Prinzipien

Besonderes Augenmerk erfordert die Wartbarkeit des Systems. Diese wird potentiell dadurch beeinträchtigt, dass die Programmlogik teilweise in R und teilweise in SQL implementiert wird und ein beachtlicher Teil der Aufgaben grundsätzlich in jeder der beiden Sprachen realisiert werden kann. Dies betrifft sowohl die Extract/Transform/Load-Prozesse (ETL) als auch die Benutzerfunktionalität. Erschwerend kommt hinzu, dass beide Sprachen bezüglich Wartbarkeit Defizite aufweisen (beide: Interpreter-Sprachen mit geringer Typstrenge, eingeschränkter Modularisierbarkeit, hoher Kompatkheit; R: grosser, oft inkonsistenter und redundanter Sprachumfang, starke Kontextabhängigkeit; SQL: fehlende Debugging-Funktionalität, zahlreiche Dialekte und Systemabhängigkeiten)[9].

[9] Für eine ausführliche Auseinandersetzung mit den diesbezüglichen Schwächen von R siehe Burns (2012), insbesondere Kapitel 8 («Believing it does as intended»).

Vor diesem Hintergrund ist eine disziplinierte und systemweit konsistente Verwendung der beiden Sprachen besondere wichtig. Folgende Prinzipien werden angewendet:

1. Aller Quellcode wird in einem gemeinsamen Repository verwaltet:
   - Alle SQL-Anweisungen werden innerhalb des R-Codes formuliert («Inline SQL»).[10]
   - Genereller Verzicht auf Stored Procedures und Datenbank-Views.
   - Datenbank-Struktur wird regelmässig als SQL-Skript ins Repository kopiert (Befehl «Server / Data Export / Dump Structure only» in der Workbench)
2. Für die Persistierung und den Abruf von Daten wird SQL verwendet:
   - Daten werden nicht über die aktuelle Verarbeitungsaufgabe hinaus in R gehalten.
   - Selektion, Projektion und Aggregation der benötigten Daten erfolgt in SQL.
3. Für Bulk-Transformationen (= viele Records) wird in der Regel SQL verwendet:
   - Ausnahmen möglich bei sehr komplexen Transformationen (z.B. zahlreiche Arbeitsschritte, viele Fallunterscheidungen, Transponieren von Zeilen / Spalten).
4. Für die Steuerung des Kontrollflusses wird R verwendet.
   - Genereller Verzicht auf SQL-Skripte.
5. Für die Aufbereitung von Tabellen und Grafiken wird R verwendet.
6. Für die Berechnung von Einzelwerten auf bereits geladenen Daten wird R verwendet (z.B. für die Bestimmung von Anschriften / Labels als Vorbereitungsschritt einer Visualisierung).
7. Bei Verwendung von R wird ein komprimierter Programmierstil vermieden.
   - Aufteilung komplexer Operationen in mehrere Anweisungen.
   - Strukturiert-imperativer Stil wird gegenüber funktionaler Programmierung bevorzugt.
8. Für wiederkehrende Aufgaben werden in R stets dieselben Befehle und Libraries verwendet (z.B. stringr für Textmanipulationen, data.table statt dataframe, ggplot2 für Grafiken).
9. Die Lesbarkeit des Codes wird in R und SQL durch bewährte Massnahmen gefördert (konsistente Benennung, Pretty Formatting, Inline-Kommentierung etc.).

2.5 Sicherheit

Die auf der Plattform verarbeiteten Daten sind naturgemäss «offen» und unterliegen keinen Vertraulichkeitsanforderungen. Dennoch ist der Sicherheit der Lösung Rechnung zu tragen. Schutzziele sind:

- Schutz vor Eindringen / «Hitch-Hiking» der verwendeten Plattform,
- Schutz allfälliger nutzerspezifischer Daten (in späteren Ausbauschritten),
- Schutz vor der Entwendung digitaler Identitäten (Login, Credentials),
- Schutz vor Manipulation oder Sabotage der Auswertungs-Funktionalität.

Erhöhte Risiken gibt es nicht, weshalb die Umsetzung allgemein etablierter Praktiken[11] als ausreichend angesehen wird, namentlich:

- Applikation und Datenbank werden in einer Virtual Private Cloud (AWS VPC) betrieben,
- Zugriff auf die Web-Applikation und Entwicklungsumgebung nur über den Proxy-Server,
- Kein direkter Anwender-Zugriff auf die Datenbank (sondern nur wie Applikation),
- Beschränkung des Administrations-Zugriff (Server und Datenbank) auf ausgewählte Geräte; Authentisierung über Key Files, soweit unterstützt,
- Rollenbasiertes Berechtigungskonzept; keine unnötige Verwendung des Root-Users.

[10] Für eine pointiert andere Bewertung von Inline SQL (mit anderen Argumenten und Prämissen) siehe Beech (2006).

[11] Eine Zusammenstellung findet sich in den «IAM Best Practices» von Amazon Web Services, vgl. Amazon (2017).

3 Modellierung der Daten

Ausgehend von einer Analyse der verfügbaren Quelldaten werden in diesem Kapitel die Daten-Architektur der Lösung und die verwendeten Datenmodelle beschrieben.

3.1 Analyse der Quelldaten

Die Open Data Plattform stellt derzeit 13 «Datensätze»[12] zur Verfügung, wovon lediglich derjenige zu den «Ist-Daten» verwendet wird: In diesem Datensatz werden für jeden Halt eines Verkehrsmittels die geplanten und tatsächlichen Zeitpunkte von Ankunft und Abfahrt an der Haltestelle aufgeführt. Die Ankunfts- und Abfahrtsverspätung lässt sich daraus einfach durch Differenzbildung ermitteln. Die Abfahrtsverspätung ist meist nicht relevant: den Fahrgast interessiert es wenig, ob sein Verkehrsmittel verspätet abgefahren ist, sofern er pünktlich am gewünschten Ort ankommt. Der Prototyp beschränkt sich daher auf die Ankunftsverspätung. Für die erste Haltestelle einer Fahrt liegt naturgemäss keine Ankunftszeit vor, folglich kann dort auch keine Ankunftsverspätung ermittelt zu werden.

Zu beachten ist, dass es sich bei den tatsächlichen Zeitpunkten zumeist um genäherte Werte handelt: Die Messpunkte der zu Grunde liegenden Leittechnik-Systeme befinden sich in der Regel vor dem effektiven Haltepunkt (z.B. bei der Bahnhofseinfahrt oder an einer Verkehrsampel). Durch Verwendung geeigneter Zu- und Abschläge wird daraus die tatsächliche Zeit geschätzt; entsprechend sind fast alle Einträge als «Prognose» oder «Geschätzt» gekennzeichnet.[13] Da es sich um die besten verfügbaren Daten handelt und die Unternehmen diese auch selbst für ihre Verspätungsprognosen, Kundeninformation und diverse Auswertungen verwenden, wird dieser Umstand toleriert. Es erscheint zudem plausibel, dass die Ungenauigkeiten relativ gering ausfallen.

Die Daten werden in «flacher», nicht normalisierter Struktur geliefert: Angaben zu Datum, Fahrt, Unternehmen («Betreiber») etc. werden für jeden Halt einer Fahrt wiederholt aufgeführt. Insgesamt werden pro Verkehrsmittel-Halt 21 Attribute geliefert. Von Bedeutung sind vor allem:[14]

- **Betriebstag:** Kalenderdatum, dem die Fahrt zugeordnet ist. Zu beachten ist, dass Fahrten mitternachtsüberschreitend sein können, d.h. eine Fahrt am Betriebstag X weist möglicherweise Ankunftszeiten am Betriebstag X+1 auf.
- **Fahrt-Bezeichner:** Eine Zeichenfolge, mit der die Zusammengehörigkeit mehrerer Halte zur gleichen Fahrt ausgedrückt wird. Das Format variiert zwischen den einzelnen Betreibern; codiert sind neben Länder- und Unternehmenscode teilweise noch Zugsnummern, Umlaufnummern und weitere Betreiber-abhängige Identifier.

BETREIBER_NAME	FAHRT_BEZEICHNER
Automobil Rottal AG	85:819:488933-07025-1
Automobildienst SZU	85:807:218417-11135-1
Automobildienste Aare Seeland mobil	85:870:51001
BDWM Transport (BD)	85:31:411:000
Berner Oberland-Bahnen	85:35:118:000
BLS AG (BLS)	85:33:14620:001
Chemins de fer du Jura	85:43:103:000
DB Regio AG	80:06 :17010:000

Abbildung 2: Ausgewählte Betreiber mit Beispielen ihrer Fahrt-Bezeichner

[12] «Datensatz» hier im Begriffsverständnis der Statistik, d.h. als Gesamtheit der Beobachtungen zu einem Thema (englisch «dataset»), nicht dagegen wie in der Datenbanktechnik üblich als einzelne Beobachtung. Zur Unterscheidung wird für Einzelbeobachtungen in dieser Arbeit stets der englische Begriff «record» verwendet.

[13] Für ausführlichere Erläuterungen zur Erhebungsmethodik und den in der Praxis auftretenden Konstellationen siehe Open Data Plattform öV Schweiz (2016b), Abschnitt «Spezielle Effekte und ihre Ursachen».

[14] Für eine vollständige Beschreibung aller gelieferten Attribute siehe Open Data Plattform öV Schweiz (2016b), Abschnitt «Struktur der Daten».

Zu beachten ist, dass Fahrt-Bezeichner nur innerhalb desselben Betriebstags eindeutig sind: Regelmässig verkehrende Fahrten haben zwar meist an aufeinanderfolgenden Tagen denselben Fahrt-Bezeichner, dies gilt aber nicht immer (Ausnahmen z.B. bei unterjährigen Anpassungen des Produktionsplans; oder wenn für einen Ausfall ein Ersatz bereitgestellt wird, der aus Kundensicht dasselbe Angebot darstellt, aus betrieblicher Sicht jedoch nicht). Umgekehrt kann der Bezeichner von tageweise verkehrenden Fahrten (z.B. Ersatzzüge, vorübergehende Verstärkungen) an anderen Tagen für gänzlich andere Fahrten «wiederverwendet» werden.

- Die Felder zu **Linien, Umlauf und Verkehrsmittel** werden von den Betreibern ebenfalls uneinheitlich verwendet. Die dort hinterlegten Informationen repräsentieren unterschiedliche fachliche Konzepte. So wird z.B. im Feld LINIEN_ID teilweise die Zugsnummer (innerhalb eines Tages nicht wiederholend), teilweise eine Gruppierungsinformation (z.B. S-Bahn-Linie 3; mehrere Fahrten innerhalb eines Tages) geliefert. Eine Verwendung dieser Daten würde eine fachliche Analyse unter Einbezug der Unternehmen voraussetzen, die hier nicht geleistet werden kann.
- **Zugsausfälle** sind mit einem Flag markiert. Diese Records werden nicht verwendet.
- Haltestellen werden als **Betriebspunkte**[15] bezeichnet und über einen international eindeutigen Code (BPUIC) identifiziert. Zusätzlich werden der Haltestellen-Name und im Bahnverkehr ein eindeutiges Kürzel (z.B. «BN» für Bahnhof Bern) geliefert.
- Die vier **Zeitpunktangaben** zu geplanter und tatsächlicher Ankunft und Abfahrt sind als Datums/Zeit-Werte codiert. Bei der Transformation in eine Uhrzeit ist zu beachten, dass sich aufgrund von Mitternachtsüberschreitungen Werte grösser als 24:00 ergeben können (Beispiel: 25:08 Uhr entspricht 1:08 Uhr des Folgestages). Angaben zur tatsächlichen Ankunfts- / Abfahrtszeit werden mit Sekunden geliefert, solche zu den Planwerten ohne Sekunden.
- Angaben zu **Durchfahrten** und **Zusatzfahrten** werden vom Prototypen nicht verwendet.

Die Daten wurden erstmals für den 18. November 2016 publiziert. In den ersten Tagen erfolgten noch kleinere Änderungen am Format, seither ist die Struktur stabil. Für den Prototypen werden die Daten seit Fahrplanwechsel am 11. Dezember 2016 verwendet. Korrekturlieferungen sind in den betrachteten 4 Monaten niemals aufgetreten. In jeder Lieferung sind die Daten von ca. 50 Verkehrsunternehmen enthalten. Ausgeschlossen werden jedoch:

- Ausländische Unternehmen (hierzu liegen nur Daten im Schweizerischen Teil der Fahrt vor),
- Unternehmen, die nur unregelmässig oder sehr wenige Fahrten aufweisen,
- Unternehmen, die keine Ist-Zeiten liefern oder deren Zeitangaben nicht plausibel sind[16],
- Unternehmen, zu denen nicht über den gesamten betrachteten Zeitraum Daten vorliegen.

Bei den verbleibenden 33 Unternehmen handelt es sich um Betreiber von Zug-, Tram- und Buslinien. Die bisher grössten Datenlieferanten sind (in absteigender Reihenfolge): SVB (Bernmobil), Verkehrsbetriebe Luzern, SBB, BLS, Auto AG Rothenburg und THURBO. Für Werktage werden bisher ca. 340'000 Datensätze pro Tag geliefert, für Sonntage sind es ca. 240'000.[17] Die Tageslieferungen werden als CSV-Dateien mit einer Grösse von 50 bis 70 MB bezogen[18]. Bei Abschluss dieser Arbeit liegen bereits ca. 38 Mio Datensätze zu 3 Mio Fahrten an 120 Betriebstagen vor.

[15] Fachlich korrekt sind Betriebspunkte eine Obermenge der Haltestellen. Betriebspunkte, die nicht Haltestellen sind, sind hier aber nicht von Relevanz.

[16] Eine detaillierte Analyse erfolgte nicht. Es gibt aber Fälle, in denen die Zeitangaben derart «regelmässig» sind, dass es sich ganz offensichtlich nicht um empirisch erhobene Werte handelt.

[17] Hier noch nicht enthalten sind jene Unternehmen, die per 6. April 2017 hinzugenommen wurden. Dazu gehören mit VBZ, VZO, SBW und Postauto mehrere sehr grosse Datenlieferanten. Die pro Tag publizierte Datenmenge ist damit neuerdings doppelt so gross. Im Prototyp werden diese Unternehmen noch nicht berücksichtigt.

[18] Die Verwendung eines dateibasierten Datenbezugs wird in Kapitel 4.1 begründet.

3.2 Aufbau des Data Warehouse

Das Data Warehouse[19] ist in einer 2-Schicht-Architektur umgesetzt, bestehend aus Staging-Bereich und Data Mart. Auf einen Core-Layer wird aus mehreren Gründen verzichtet:

- Ein wichtiges Ziel liegt darin, anhand des Prototypen aufzuzeigen, was in kurzer Zeit möglich ist. Der Aufbau eines zusätzlichen, langfristig motivierten Layers ist dabei nicht zielführend.
- Betrachtet wird eine eng abgegrenzte Fachdomäne mit nur einer Datenquelle und wenigen Entitäten. In diesem Szenario ist der Nutzen eines normalisierten Core-Modells gering – die notwendigen Transformationsschritte können unmittelbar bei der Überführung in die Auswertungs-Datenbank vorgenommen werden, ohne dass dies zu übermässiger Komplexität führt.
- Mit dem Bezug der Daten im Dateiformat sind einige typische Aufgaben einer DWH-Stage bereits erfüllt: Entkopplung vom Quellsystem, Nachvollziehbarkeit durch Aufbewahrung der Quelldaten, jederzeitige Möglichkeit eines erneuten Loads (partiell oder vollständig). Beim Laden der Stage können daher bereits in grösserem Ausmass Transformationen durchgeführt werden als dies bei einem eng gekoppelten Quellsystem empfehlenswert wäre. Dies wiederum vermindert den zusätzlichen Nutzen eines Core-Layers.
- Ein Historisierungs-Schritt ist nicht notwendig: Die Fakten werden bereits mit klarem Zeitbezug (Zuordnung zu Betriebstag) geliefert, für die Dimensionen wird Stabilität unterstellt[20]. Damit entfällt eine weitere typische Aufgabe des Core-Layers.

Zusätzlich zu den importierten Daten (Stage) und dem dimensionalen Modell (Data Mart) wird noch eine Metadaten-Tabelle vorgehalten, in der die Import-Läufe protokolliert werden.

Für die Tabellen im Data Warehouse gelten folgende Namenskonventionen:

- stg_* : Für Staging-Tabellen.
- meta_*: Für Meta-Daten.
- etl_* : Für Hilfstabellen, die im Rahmen eines ETL-Prozesses benötigt werden.
- dim_* : Für Dimensions-Tabellen im dimensionalen Modell (Data Mart).
- fct_* : Für Fakten-Tabellen im dimensionalen Modell (Data Mart).

3.3 Import-Bereich (Stage)

Zentrales Element des Import-Bereichs ist die Tabelle stg_istdaten, in welche die täglich von der Open Data-Plattform bezogenen Dateien geladen werden. Dabei wird wie folgt verfahren:

- Es werden alle Attribute importiert, auch jene, die im Prototypen nicht verwendet werden (z.B. Angaben zu Umläufen, Durchfahrten, Zusatzfahrten, Abfahrtszeiten). Import-Prozess und Datenstruktur können somit auf Produktions- und Entwicklungs-Instanz auch dann identisch gehalten werden, wenn neue Features und Attribut-Verwendungen in Entwicklung stehen.
- Aus den gleichen Überlegungen werden sämtliche Records importiert, also auch jene von Betreibern, die in den Auswertungen (noch) nicht berücksichtigt werden.
- Es findet bereits eine Typ-Konversion statt. Dies ist möglich, weil sich einerseits das Quellformat als stabil und verlässlich erwiesen hat und andererseits im Fehlerfall die archivierten CSV-Dateien zur Rekonstruktion der Ursprungsdaten verwendet werden können.

[19] Ein Data Warehouse wird hier als Gesamtheit aller beteiligten Schichten aufgefasst und nicht - wie bei einigen Autoren üblich - reduziert auf die Core- oder Auswertungsschicht.

[20] Stabilität in den Dimensionen (SCD0) ist eine vereinfachende Annahme für die Zwecke dieser Arbeit. Vergleiche dazu die Abgrenzungen in 1.3 sowie die Ausführungen zum dimensionalen Modell in 3.4.

- Es werden folgende abgeleitete Werte hinzugefügt:
    - Planmässige Ankunfts- und Abfahrtszeit als Uhrzeit (d.h. nach Entfernung des Datums und unter Berücksichtigung von Mitternachtsüberschreitungen),
    - effektive Ankunfts- und Abfahrtszeit als Uhrzeit (dito),
    - Ankunfs- und Abfahrtsverspätung als Sekunden-Differenz zwischen Plan- und Ist-Wert,
    - Markierung der ersten und letzten Haltestelle jeder Fahrt mit einem Flag.
- Es wird eine Import-ID hinzugefügt, mit deren Hilfe in Log Files und Metadaten (Tabelle meta_istdaten_imports) der Ladeprozess nachvollzogen werden kann.

Die Daten in stg_istdaten brauchen nicht langfristig aufbewahrt zu werden. Auf der Produktions-Instanz empfiehlt sich sogar eine gelegentliche Bereinigung, weil mit steigender Tabellengrösse nachfolgende ETL-Schritte verlangsamt werden. Auf der Entwicklungs-Instanz ist es sinnvoll, eine ausreichende Auswahl von Staging-Records für Testzwecke zur Verfügung zu haben.

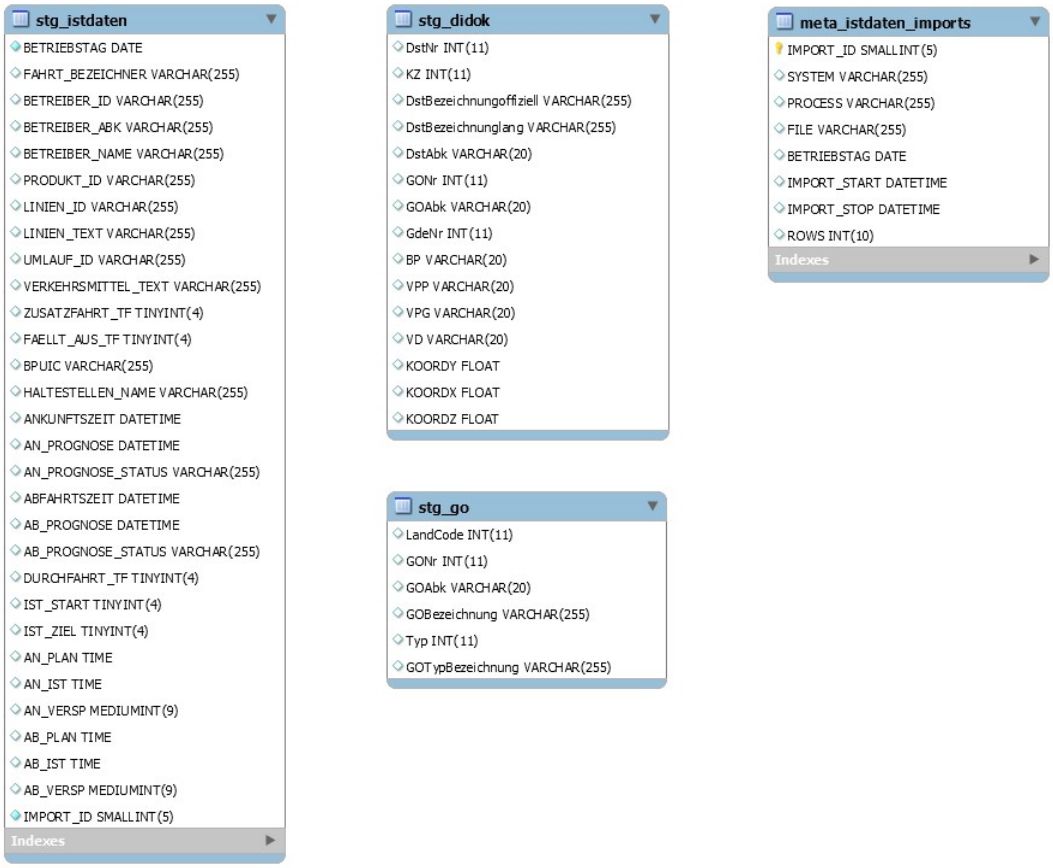

Abbildung 3: Staging- und Metadaten-Tabellen

Weiterhin umfasst der Import-Bereich zwei Tabellen, die einmalig vom Open Data-Portal bezogen wurden: Aus dem Datensatz **«Geschäftsorganisationen»** (Tabelle stg_go) wurde die Klassifikation der Unternehmen übernommen («Strasse» für die Betreiber von Bus- und Tramlinien, «Bahn», «Schiff»). Der Datensatz **«Dienststellen-Dokumentation»** (Tabelle stg_didok) umfasst ein Verzeichnis aller Haltestellen. Hieraus werden die Geokoordinaten übernommen, was für die spätere Erstellung von Karten-bezogenen Auswertungen nützlich sein kann (bisher nicht umgesetzt).

3.4 Auswertungs-Datenbank (Data Mart)

Die Auswertungs-Datenbank besteht aus insgesamt sieben Dimensionen sowie aus drei Fakten-Tabellen auf unterschiedlichem Aggregationsniveau.

3.4.1 Feingranulares Modell

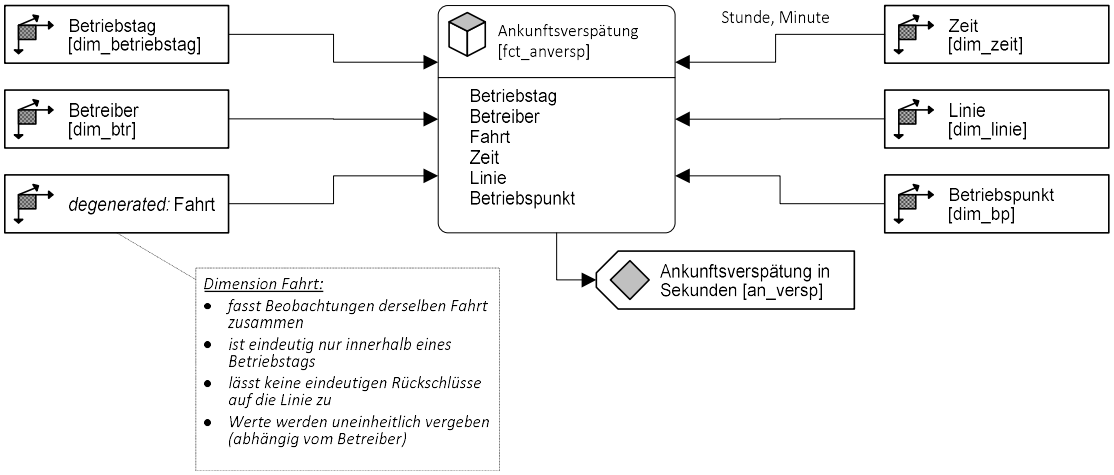

Abbildung 4: Die feingranulare Ebene des dimensionalen Modells

Abbildung 4 gibt einen Überblick über die feinste Ebene der Auswertungs-Schicht in (leicht modifizierter[21]) ADAPT-Notation. Jeder Quelldaten-Record der ausgewählten Betreiber-Unternehmen (d.h. jeder Verkehrsmittel-Halt) wird zu einem Eintrag in der Faktentabelle und kann anhand von 6 Dimensionen klassifiziert werden. Alle Dimensions-Referenzen (Fremdschlüssel) der Faktentabelle werden indexiert.

Bislang einziges Fakten-Attribut ist die **Ankunftsverspätung in Sekunden**. Eine sinnvolle Erweiterung wäre ein Attribut, welches die Verspätungs-Änderung auf dem vorausgegangenen Fahrtabschnitt ausweist. Dies würde es ermöglichen, Analysen über den Auf- und Abbau von Verspätungen im Fahrtverlauf zu erstellen. Auch Angaben zu Ausfällen, ausserordentlichen Durchfahrten, Ersatzfahrten oder Abgangsverspätungen könnten in die Faktentabelle aufgenommen werden.

Bei der **Fahrt** handelt es sich um eine degenerierte Dimension: Ihre derzeit einzige Aufgabe besteht darin, die Zusammengehörigkeit von Fakten der gleichen Fahrt zum Ausdruck zu bringen. Die Dimension trägt bislang keine eigenen Attribute und kann auch nicht als Selektions- oder Drill-Down-Kriterium verwendet werden, da der Fahrtbezeichner nicht notwendig stabil über mehrere Tage ist. Denkbar wäre die Erweiterung zu einer Dimension «Kurs», welche äquivalente Fahrten unterschiedlicher Betriebstage zusammenfasst. In diesem Fall müsste statt des Fahrtbezeichners ein fachliches Kriterium zu Grunde gelegt werden (z.B. die Haltestellenabfolge inkl. Fahrplan-Zeiten).

[21] In eckigen Klammern sind allfällig abweichende Namen der zugehörigen DB-Tabellen und -Attribute angegeben.

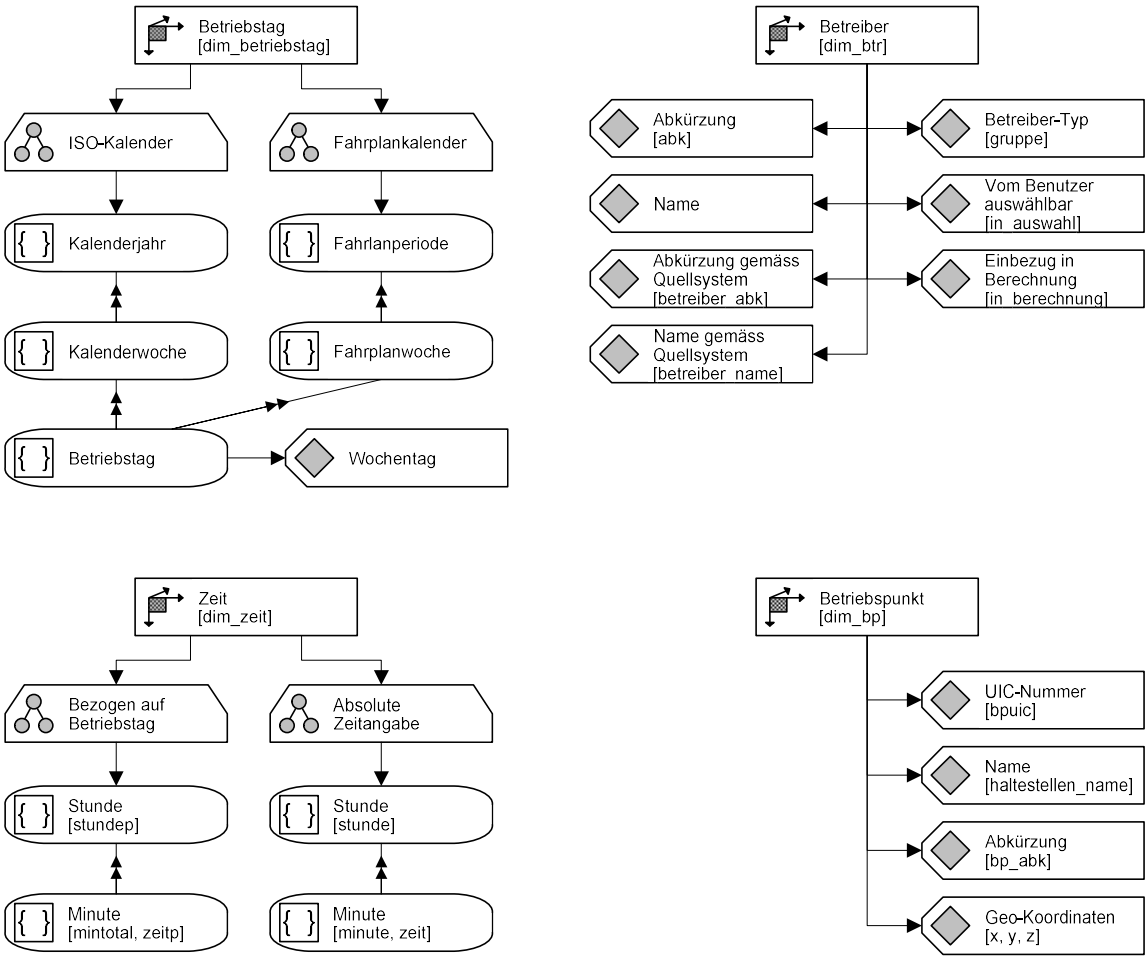

Abbildung 5: Die Dimensionen Betriebstag, Betreiber, Zeit und Betriebspunkt.

Die Dimension **Betriebstag** klassifiziert die Fakten nach Datum. Grössere Änderungen am Fahrplan erfolgen jeweils am 2. Sonntag im Dezember, die dazwischenliegende Zeit wird als Fahrplanperiode bezeichnet. Die Wochen eines Fahrplans beginnen ebenfalls an einem Sonntag. Neben dem bekannten Jahreskalender ergibt sich also eine zweite Hierarchie. Die Granularität beträgt 1 Tag. Der Wertebereich reicht derzeit vom 11.12.2016 bis 9.12.2017 und muss zu gegebener Zeit erweitert werden.

Die Dimension **Betreiber** umfasst zusätzlich zu den aus den Quelldaten bezogenen Angaben noch einige applikationsspezifische Attribute (z.B. abweichende Bezeichnungen und eine Gruppierung). Der Wertebereich war 4 Monate lang stabil und muss nun vermutlich erweitert werden (vgl. Fussnote 17). Eine automatische Hinzunahme aus den Quelldaten (d.h. > SCD0) scheint jedoch nicht ratsam, da Auswertungen über zusätzliche Unternehmen erst angeboten werden sollten, nachdem deren Datenqualität über mehrere Wochen begutachtet werden konnte.

Die Dimension **Zeit** weist zwei Hierarchien auf, die sich darin unterscheiden, ob Zeitangaben am Folgetag in das 24-Stunden-Raster transformiert werden (absolute Zeitangabe, z.B. 1:08 Uhr) oder nicht (bezogen auf Betriebstag, z.B. 25:08 Uhr). Die Granularität beträgt 1 Minute, der Wertebereich reicht von 0:00 bis 30:00 und ist stabil.

Die Dimension **Betriebspunkt** umfasst alle auftretenden Haltestellen. Änderungen am Wertebereich (z.B. Umbenennung von Haltestellen, Eröffnung oder Schliessung von Haltestellen) sind selten und werden von den Betreibern nach Möglichkeit auf das Datum eines Fahrplanwechsels gelegt. Aus diesem Grund erscheint auch hier der SCD0-Ansatz vertretbar.

Unter dem Begriff **Linie** werden üblicherweise Fahrten zusammengefasst, die eine identische oder mindestens ähnliche Haltestellen-Abfolge aufweisen. Für die Erstellung von Auswertungen ist es sinnvoll, zwischen den Fahrtrichtungen zu unterscheiden, d.h. die Abfolgen A-B-C-D-E und E-D-C-B-A sollten als getrennte Linien[22] modelliert werden. Wie bereits in 3.1 thematisiert wurde, sind auswertbare Linieninformationen nicht für alle Betreiber vorhanden. Vor allem im Bahnverkehr existieren häufig nur Zugskategorien oder die Linienangaben sind nicht eindeutig (z.B. schweizweit mehrere S1-Linien oder Mischbezeichnungen wie S3/S31).

Auch die automatisierte Herleitung von Linien aus den vorhandenen Fahrtverläufen erweist sich als schwierig. Dies soll anhand einiger Beispiele erläutert werden:

- Sofern die Haltestellen-Abfolge aller Fahrten einer Linie identisch und eindeutig bestimmbar ist, können diese problemlos zusammengefasst werden.[23]
- Häufig treten jedoch Konstellationen auf, bei denen einzelne Fahrten nur einen Teil der Linie abdecken (z.B. C-D-E), einzelne Haltestellen auslassen (z.B. A-C-D-E) oder zusätzliche beinhalten (z.B. A-B-C-D-X-E). Oft sollen solche Fahrten aber dennoch der Linie zugeschrieben werden.
- Diese Fälle sind jedoch zu unterscheiden von jenen, wo zwei unterschiedliche Linien streckenweise parallel verlaufen und danach verzweigen (z.B. Linie 1: A-B-C-D-E und Linie 2: A-B-C-K-L-M) oder wo Fahrten einer Strecke mit unterschiedlicher Haltepolitik als separate Linien angesehen werden (z.B. S-Bahn: A-B-C-D-E, Intercity: A-C-E).
- Unklar ist auch der Umgang mit alternierenden Streckenverläufen, die gemeinsam einen «integralen Takt» bilden (z.B. zur geraden Stunde: A-B-C-D-E und X-Y-C-K-L, zur ungeraden Stunde: A-B-C-K-L und X-Y-C-D-E).
- Schliesslich ist die Abfolge der Haltestellen aus den Quelldaten nicht immer eindeutig zu ermitteln: Es wird keine Sequenz-Nummer geliefert und die Fahrplan-Zeiten (Granularität 1 Minute) können bei nahe aufeinanderfolgenden Haltestellen identisch sein, so dass sie sich nicht als Sortierkriterium eignen. Folglich kann mitunter nicht zwischen A-B-C-D-E und A-B-D-C-E unterschieden werden.

Alle genannten Fälle treten im öV Schweiz in bedeutender Häufigkeit auf und sind entsprechend oft in den Quelldaten zu finden. Während es sich beim letzten Punkt um ein eher technisches Problem handelt, verlangen die anderen Konstellationen nach einer fachlichen Beurteilung, da sie mit formalen Methoden nicht eindeutig entscheidbar sind und höchstens näherungsweise gelöst werden können.[24]

Die Dimension Linie ist aus diesen Gründen bisher nicht im Prototypen implementiert. Abbildung 6 zeigt einen fachlichen Entwurf: Eine Linie wird dabei durch eine Folge von Haltestellen (= Referenzen auf Einträge der Dimension Betriebspunkt) charakterisiert. Diese Haltestellen müssen in der angegebenen Reihenfolge in einer Fahrt enthalten sein, damit die Fahrt der Linie zugeschrieben wird. Bei mehreren Linien-Kandidaten wird nach Priorität entschieden. Sinnvoll erscheint weiterhin eine Kategorisierung (z.B. IC, RE, S-Bahn) und ein Verweis auf das Linien-Element der Gegenrichtung.

[22] Alternativ könnte man auch von einer gemeinsamen Linie mit zwei separaten «Halblinien» sprechen.

[23] In MySQL kann dies leicht realisiert werden, in dem die Identifier aller Haltestellen einer Fahrt mit der Aggregationsfunktion group_concat() zu einer Zeichenfolge verkettet werden, die sich dann mit andern Fahrten vergleichen lässt.

[24] In 4.5 wird eine mögliche Heuristik vorgestellt.

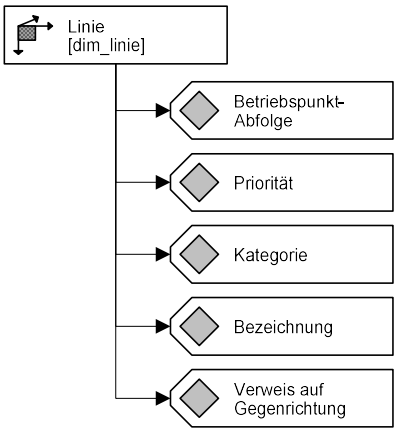

Abbildung 6: Entwurf eines fachlichen Modells für die Dimension Linie.

Wenn man unterstellt, dass alle Fahrten einer Linie vom gleichen Betreiber ausgeführt werden, könnte die Linie alternativ auch als Verfeinerung der bestehenden Betreiber-Dimension aufgefasst werden. Die Elemente aus Abbildung 6 sind dann in diese Dimension zu integrieren.

3.4.2 Aggregierte Modelle

Um trotz der grossen Datenmenge niedrige Antwortzeiten zu erreichen, empfiehlt es sich, Auswertungen nicht in jedem Fall auf der feinsten Granularitätsstufe durchzuführen, sondern soweit möglich auf vorberechnete, aggregierte Daten zuzugreifen. Um z.B. die Pünktlichkeitswerte eines Betreibers im Jahresverlauf zu berechnen, ist in den Fakten keine Differenzierung zwischen Betriebspunkten, Tageszeiten, Linien und Fahrten notwendig – wünschenswert ist in diesem Fall also eine geeignete Zusammenfassung aller Vekehrsmittel-Halte desselben Betreibers pro Tag.

Dies wirft die Frage auf, wie das Fakten-Attribut «Ankunftsverspätung in Sekunden» bei Zusammenfassung mehrerer Beobachtungen verdichtet werden soll. Die naheliegenden Ideen, entweder eine Summe (Anzahl Verspätungssekunden über alle Beobachtungen) oder einen Durchschnitt (Mittel der Verspätungssekunden über alle Beobachtungen) zu verwenden, eignen sich nicht:

- Als Masszahl für die Pünktlichkeit ist die Angabe von Anteilswerten üblich, z.B. «88.7% aller Ankünfte waren weniger als 3 Minuten verspätet.». Dies kann weder aus der Summe noch aus dem Mittelwert herausgelesen werden.
- Eine weitere sinnvolle Beschreibung von Pünktlichkeiten basiert auf der Verwendung von Perzentilen, z.B. «95% der Züge waren weniger als 4:30 Minuten verspätet» oder «die Hälfte der Züge kam mehr als 1:15 Minuten verspätet an.». Auch dies kann nicht aus Summen- oder Durchschnittswerten hergeleitet werden.
- Die empirische Verteilung der Verspätungssekunden ist sehr gespreizt und stark rechtsschief (vgl. Abbildung 7). Summen- und Mittelwertangaben sind daher weder anschaulich noch repräsentativ.

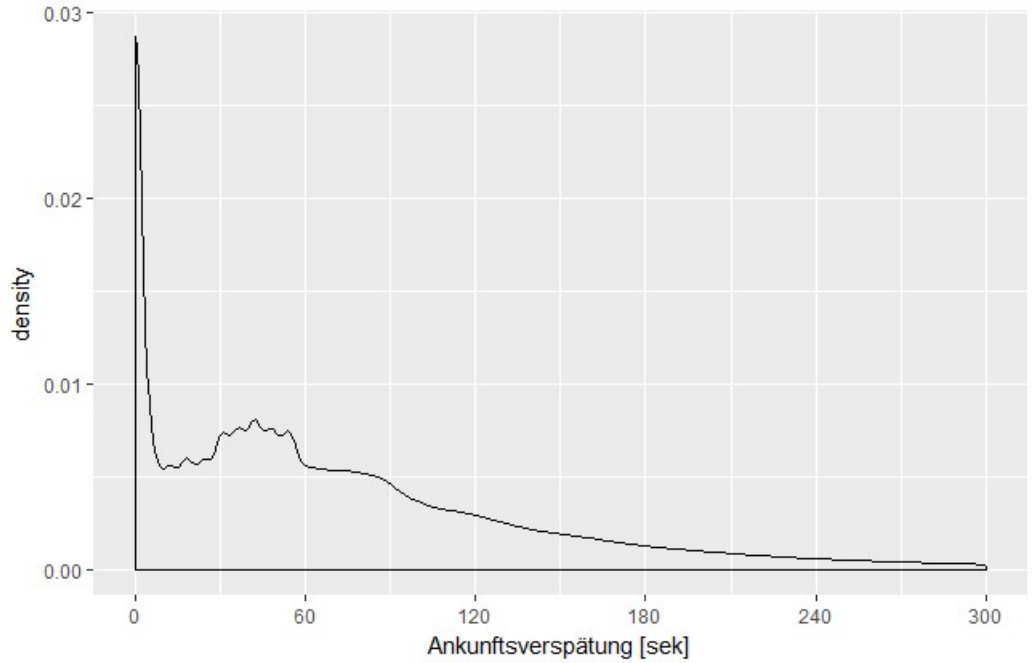

Abbildung 7: Verteilung der Ankunftsverspätungen alle betrachteten Betreiber im März 2017.[25]

Die Aggregationsmethode sollte also dem Umstand Rechnung tragen, dass es sich bei Verspätungen um Verteilungsinformationen handelt, die durch eine Kombination von Ausmass (Sekunden) und Häufigkeit (Anzahl oder Anteil) charakterisiert werden.

Die Lösung liegt in der Einführung einer Dimension «Verspätungsniveau»: jede Ausprägung dieser Dimension repräsentiert ein Intervall von Verspätungswerten und in der Faktentabelle wird ausgewiesen, wie viele Beobachtungen in dieses Intervall fallen (vgl. Abbildung 8). Das Verfahren ist vergleichbar mit der Erstellung von Histogrammen und der dort praktizierten Bildung von «bins».

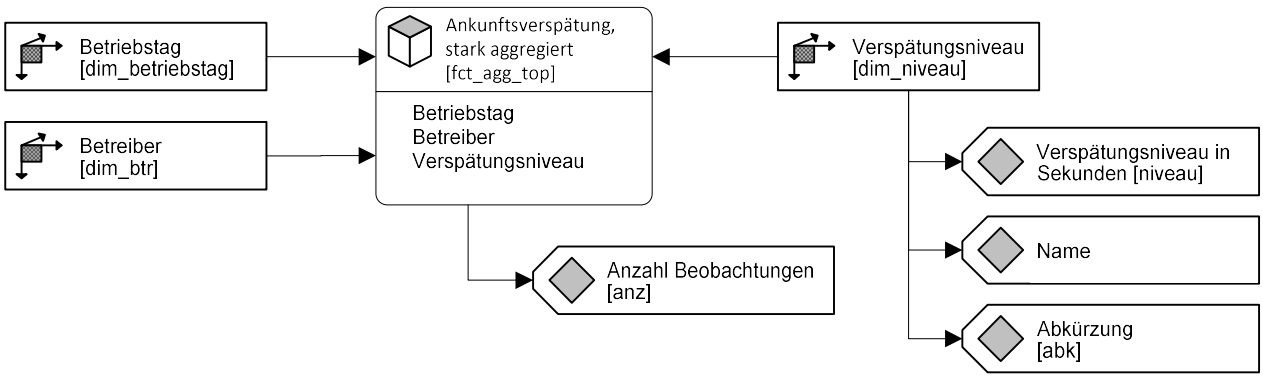

Abbildung 8: Aggregiertes dimensionales Modell mit Verwendung von «Verspätungsniveaus».

Aufgrund der starken Schiefe der Verteilung, ist die Schrittweite der Intervallgrenzen nicht konstant: Bis zu einer Verspätung von 3 Minuten beträgt sie 15 Sek, später dann 30 sek, 1 min, 5 min etc.. Ein «Rest»-Intervall fasst alle Verspätungen >2 Stunden zusammen.

Anders als bei Histogrammen sind die Intervalle nicht disjunkt, sondern bilden eine Inklusionskette:

$$\text{Verspätungen} > 60\ \text{sek} \subset \text{Verspätungen} > 45\ \text{sek} \subset \text{Verspätungen} > 30\ \text{sek} \subset \ldots$$

[25] Die kleinen «Wellen» im linken Teil der Dichtefunktion resultieren vermutlich daher, dass einige Quellsysteme die Verspätungsinformation in 6 Sekunden-Schritten auflösen – im Zusammenhang mit Fahrplänen ist die Verwendung von Zehntelminuten verbreitet.

Die in der Faktentabelle auftretenden Häufigkeiten sind also kumuliert (siehe Beispiel in Abbildung 9).

NIVEAU	NAME	ANZ
0	alle	13503
15	> 15 sek	10699
30	> 30 sek	9721
45	> 45 sek	8583
60	> 60 sek	7326
75	> 75 sek	6043
90	> 90 sek	4927
105	> 105 sek	3998
120	> 120 sek	3288
135	> 135 sek	2685

NIVEAU	NAME	ANZ
150	> 150 sek	2198
165	> 165 sek	1808
180	> 180 sek	1517
210	> 210 sek	1023
240	> 240 sek	718
270	> 270 sek	481
300	> 300 sek	323
360	> 360 sek	155
420	> 420 sek	76
480	> 480 sek	39

NIVEAU	NAME	ANZ
540	> 540 sek	27
600	> 10 min	23
660	> 11 min	19
720	> 12 min	16
780	> 13 min	14
840	> 14 min	9
900	> 15 min	4
960	> 16 min	1

Abbildung 9: Häufigkeit der Verspätungsniveaus bei den Fahrten der BLS am 6. April 2017.

Die Verwendung kumulierter Werte hat folgende Vorteile:

- Bei der Berechnung der Faktentabelle muss nur mit der Untergrenze des Intervalls verglichen werden und nicht mit 2 Werten. Dies ist in der Berechnung effizienter.
- Das «0-Niveau» gibt zugleich die Anzahl aller Beobachtungen an. Wo Verhältniswerte benötigt werden, muss der Nenner somit nicht separat bestimmt werden.
  Beispiel: Anteil > 3 min = Anzahl > 3 min : Anzahl 0-Niveau = 1517 : 13503 = 11.2%.
- Viele der im Prototypen implementierten Auswertungen benötigen kumulative Werte; die Notwendigkeit des Aufsummierens entfällt dort.
- Perzentile können aus der Tabelle abgelesen werden.
  Beispiel Median: gesuchte Anzahl = Anzahl 0-Niveau : 2 = 6751.5
  → Median liegt zwischen 60 und 75 sek (je nach Berechnungsmethode).
- Wo nur einzelne Werte von Interesse sind (z.B. 3 Minuten-Pünktlichkeit) müssen auch nur diese Intervalle aus der Datenbank abgefragt werden.

Das in Abbildung 8 dargestellte Modell bildet das höchste im Prototypen verwendete Aggregationsniveau. Eine weitere Verdichtung ist nicht erforderlich (aktuell umfasst die Tabelle fct_agg_top «nur» ca. 100'000 Records).

Das Modell eignet sich nicht, um die Verspätungsentwicklung im Tagesverlauf zu analysieren, da die Zeit-Dimension fehlt. Das feingranulare Modell (Abbildung 4: Die feingranulare Ebene des dimensionalen Modells) ist andererseits zu wenig performant. Im Prototypen ist daher noch eine mittlere Aggregationsebene implementiert, bei der eine Aufschlüsselung nach Stunde (= zweite Hierarchie-Ebene der Dimension Zeit) erfolgt (vgl. Abbildung 10). Nach diesem Vorbild könnten bei Bedarf noch weitere «mittlere» Modelle implementiert werden (z.B. für Haltestellen und Linien).

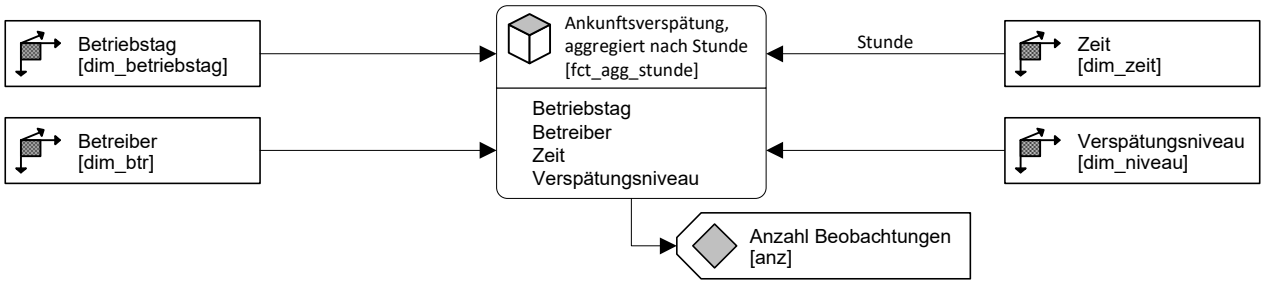

Abbildung 10: Mittlere Granularitätsebene des dimensionalien Modells: Aggregation nach Stunden.

4 ETL-Prozesse und Scheduling

Bezug und Verarbeitung der Quelldaten von opentransportdata.swiss gliedern sich in 3 Phasen mit einer gemeinsamen Ablaufsteuerung. Für die Linienbildung existiert ein heuristisches Verfahren, das aber noch nicht in den Prototypen integriert wurde.

4.1 Bezug der Quelldaten (Phase 1)

Die Open Data-Plattform des öV Schweiz basiert auf dem Open Source-Produkt CKAN[26]. Es werden 2 Varianten für den Datenbezug angeboten: Einerseits können Records über ein REST-API im JSON-Format bezogen und direkt in der Client-Applikation weiterverarbeitet werden. Andererseits besteht die Möglichkeit, pro Betriebstag eine CSV-Datei mit allen Daten dieses Tages herunterzuladen. Beide Möglichkeiten werden durch eine R-Library («ckanr») unterstützt. Die Verwendung des REST-APIs erwies sich in Versuchen als umständlich, zumal die verfügbare Dokumentation[27] lückenhaft ist. Nachteilig ist vor allem, dass nur eine begrenzte Zahl von Records pro Aufruf geladen werden kann, so dass ein Paging-Mechanismus (inkl. Überwachung und Fehlerbehandlung) implementiert werden muss. Die Wahl fiel daher auf den CSV-Download, was sich als einfache und robuste Lösung herausstellte. Ein zusätzlicher Vorteil liegt darin, dass die bezogenen Daten in Dateiform vorliegen, was Nachvollziehbarkeit, Backup und Archivierung erleichtert.

Der Ablauf des Datenbezugs ist wie folgt:

1. Anfrage der verfügbaren «Ist-Daten»-Lieferungen beim CKAN-API der Open Data-Plattform.
2. Abgleich mit den bereits im Prototypen vorhandenen CSV-Dateien.
3. Anforderung der noch nicht vorhandenen Lieferungen (Betriebstage) über die zugehörige URL.
4. Speicherung der bezogenen Dateien im Dateisystem der Cloud-Instanz. Dabei wird der jeweilige Betriebstag im Dateinamen codiert (z.B. 2017-02-04istdaten.csv).
5. Hinzufügen von Datei-Kopien zu einem ZIP-Archiv im Dateisystem der Cloud-Instanz. Dieses dient als erste Sicherungskopie und erleichtert die Archivierung und den Transfer auf andere Rechner. Gegenüber dem CSV-Format weist es eine sehr hohe Kompression auf.

4.2 Laden der Quelldaten in den Import-Bereich (Phase 2)

Die Daten aus dem Dateisystem werden in die Stage-Tabellen der Datenbank überführt. Wie in 3.2 begründet wurde, handelt es sich dabei nicht um 1:1-Kopien, sondern es finden erste Transformationen statt. Folgende Schritte werden durchgeführt:

1. Betriebstag aus Dateinamen ableiten.
2. Beenden, falls Daten dieses Betriebstags bereits in der Stage vorhanden sind.

Andernfalls:

3. Neue Import-ID bestimmen mit High-Water-Mark-Verfahren.
4. Anlegen eines Metadaten-Eintrags zur importierten Datei, inklusive Startzeit des Imports.
5. Einlesen der Datei nach stg_istdaten. Dabei:
   - Zuordnung der Import-ID zu allen Records zwecks Nachvollziehbarkeit,

[26] CKAN steht für «Comprehensive Knowledge Archive Network» und wird von zahlreichen Open Data-Portalen weltweit eingesetzt. Für ausführliche Informationen zum Produkt siehe ckan.org.

[27] Vgl. Open Data Plattform öV Schweiz (2016) und CKAN (2013).

- Typ-Konversion,
- Extraktion von Uhrzeiten aus den kombinierten Datums-/Zeit-Feldern,
- Berechnung der Ankunfts- und Abgangsverspätungen als Differenz aus Ist- und Plan-Zeit,
- Markierung der ersten und letzten Haltestelle einer Fahrt mit einem Flag.

6. Protokollieren der Abschlusszeit des Imports in der Metadaten-Tabelle.

4.3 Transformation der Stage-Daten in die Dimensionalen Modelle (Phase 3)

Nach Laden des Import-Bereichs werden die Fakten-Tabellen des dimensionalen Modells um die neuen Einträge ergänzt. Zur Unterstützung der Nachvollziehbarkeit wird weiterhin die Import-ID aus Phase 2 verwendet. Der Ablauf ist folgendermassen:

1. Prüfe, ob Daten des Imports bereits ins feingranulare Modell geladen wurden.
   Falls nein:
   - Ermittle für jeden Stage-Record die zugehörigen Dimensions-IDs. Trage diese zusammen mit der Ankunftsverspätung und der Import-ID in die Faktentabelle fct_anversp ein.
2. Prüfe, ob Daten des Betriebstags bereits ins hochaggregierte Modell geladen wurden.
   Falls nein:
   - Ermittle für jede auftretende Kombination aus Betreiber und Verspätungsniveau die Häufigkeit des Auftretens. Trage diese Werte gemeinsam mit den Dimensions-IDs und der Import-ID in die Faktentabelle fct_agg_top ein.
3. Prüfe, ob Daten des Betriebstags bereits ins mittelaggregierte Modell geladen wurden.
   Falls nein:
   - Ermittle für jede auftretende Kombination aus Betreiber, Stunde und Verspätungsniveau die Häufigkeit des Auftretens. Trage diese Werte gemeinsam mit den Dimensions-IDs und der Import-ID in die Faktentabelle fct_agg_stunde ein.

Eine Veränderung oder Ergänzung der Dimensionsdaten erfolgt nicht (SCD0).

Da das Open Data-Portal im Rahmen dieser Arbeit als «single source of truth» angesehen wird (vgl. 1.3) gibt es keine Massnahmen zur Qualitätskontrolle und Bereinigung (Cleansing) der Daten.

4.4 Ablaufsteuerung, Fehlerbehandlung und Logging

Die drei vorstehend beschriebenen Phasen werden durch R-Skripte umgesetzt, aus denen die erforderlichen SQL-Befehle abgesetzt werden.

Die Datenlieferung zum Vortrag wird auf dem Open Data-Portal üblicherweise zwischen 3:00 Uhr und 4:00 bereitgestellt. Hin und wieder kommt es zu Verzögerungen von wenigen Stunden bis zu zwei Tagen. Um neue Lieferungen zeitnah zu übernehmen, wird Phase 1 (Bezug der Quelldaten) stündlich ausgelöst. Falls dabei neue Dateien bezogen werden, werden automatisch zunächst Phase 2 (Laden des Import-Bereichs) und danach Phase 3 (Laden der dimensionalen Modelle) angestossen.

Zusätzlich zu diesem automatischen Datenbezug und Load existiert ein Skript, das sukzessive für alle archivierten CSV-Dateien Phase 2 und Phase 3 durchführt. Dies ermöglicht einen partiellen oder vollständigen Reload von Import-Bereich und dimensionalen Modellen.

Die wichtigsten Arbeitsschritte und -ergebnisse aller Skripte werden in einer Log-Datei protokolliert.

Alle Skripte sind so aufgebaut, dass bei mehrfacher Ausführung keine Doubletten angelegt und keine bestehenden Records beeinträchtigt werden. Bei nicht erfolgreicher oder unklarer Beendigung eines Skripts kann dieses also nach Problembehebung einfach erneut ausgeführt werden.

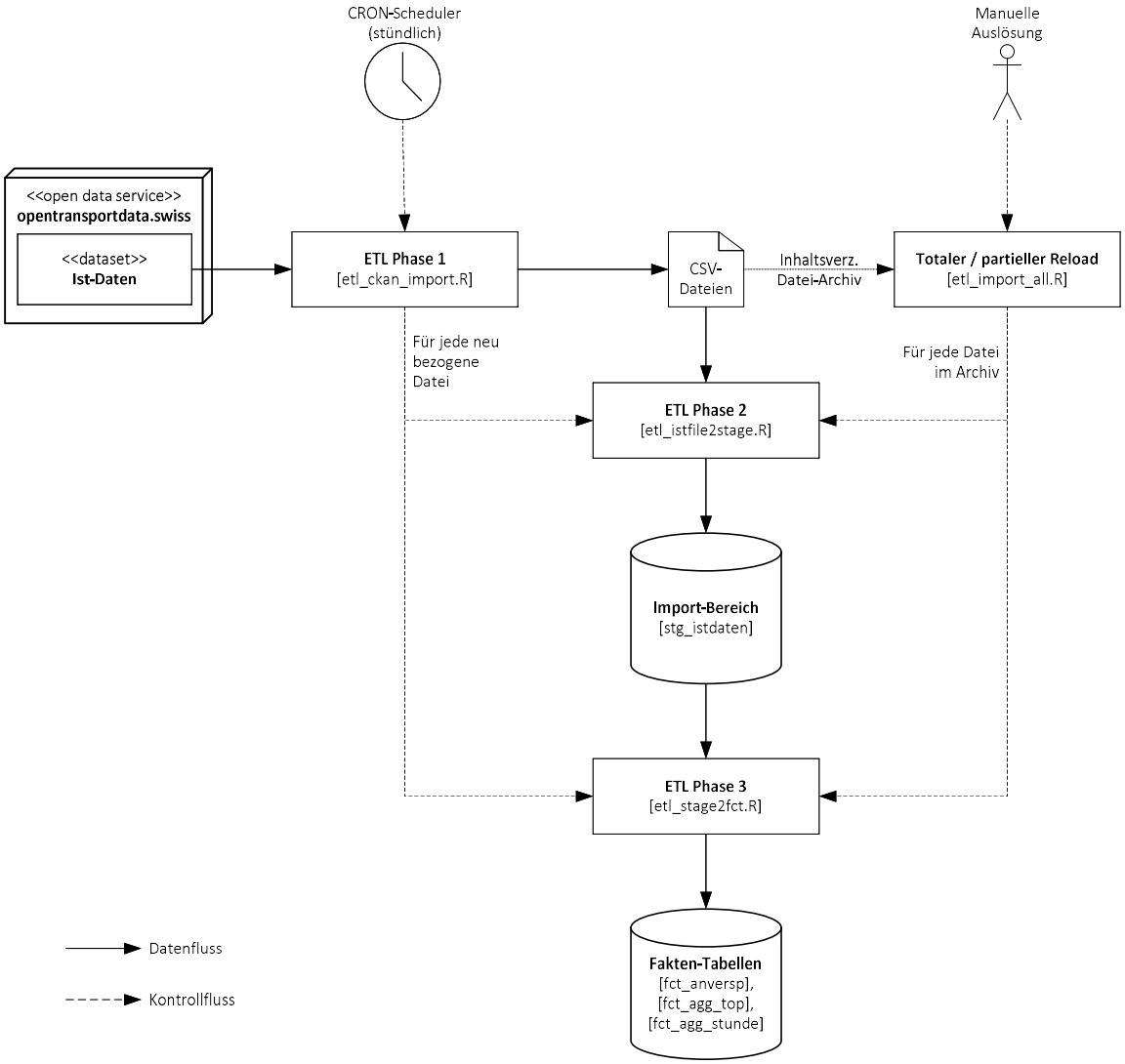

Abbildung 11: ETL-Verarbeitungsschritte und zugehörige Kontroll- und Datenflüsse.

4.5 Heuristik für die Linien-Bildung

Wie in 3.4.1 erläutert wurde, ist eine automatisierte Herleitung der Linien-Information nur näherungsweise möglich, da diverse Entscheidungsspielräume existieren, welche eine fachliche Beurteilung erforderlich machen. Hier wird nun eine Heuristik skizziert, die eine solche näherungsweise Linien-Bildung ermöglicht.

Das Verfahren ist parametrisierbar und somit grundsätzlich geeignet, unterschiedliche fachliche Vorgaben zu berücksichtigen. Aufgrund des hohen Rechenaufwands empfiehlt sich die Anwendung auf eine begrenzte Zahl von Betriebstagen – das dabei erzielte Ergebnis sollte verallgemeinerbar für den übrigen Zeitraum sein. Folgende Schritte sind auszuführen:

1. Bilde für jede Fahrt eine korrekt sortierte Haltestellen-Abfolge und speichere ihre Identifier als Zeichenfolge Z1 (in MySQL: … group_concat(BPUIC)…). Schliesse dabei Fahrten aus, für die eine eindeutige Sortierung nicht möglich ist.
2. Fasse Fahrten mit gleichem Z1 zu «Sublinien» zusammen.
   Optional: es müssen auch weitere Attribute übereinstimmen, z.B. der Betreiber oder die in den Quelldaten gelieferte Linien- oder Produktinformation.
3. Entferne Sublinien, deren Z1 im Z1 einer anderen Sublinie enthalten ist.

4. Optional: Entferne Sublinien, die im betrachteten Zeitraum weniger als n_1 mal auftreten.

Die so erhaltenen Sublinien stellen bereits eine brauchbare Zusammenfassung dar, berücksichtigen aber noch nicht ausreichend, dass auch Fahrten zusammengefasst werden sollen, deren Haltestellen-Abfolgen nicht identisch, sondern nur ähnlich ist. Aus diesem Grund werden die folgenden Schritte durchgeführt:

5. Definiere eine Untermenge P aller Haltestellen, die als «prägend» für die Linienbildung angesehen werden. Die Festlegung kann manuell oder automatisch erfolgen. Eine brauchbare Heuristik besteht darin, jene Haltestellen als prägend einzustufen, an denen mindestens eine Fahrt beginnt oder endet.
6. Bilde für jede Sublinie die korrekt sortierte Abfolge ihrer prägenden Haltestellen (analog zu Schritt 2) und speichere die Identifier als Zeichenfolge Z2.
7. Fasse Sublinien mit gleichem Z2 zu «Linien» zusammen.
   Optional: es müssen auch weitere Attribute übereinstimmen, z.B. der Betreiber oder die in den Quelldaten gelieferte Linien- oder Produktinformation.
8. Entferne Linien, deren Z2 im Z2 einer anderen Linie enthalten ist.
9. Optional: entferne Linien, die im betrachteten Zeitraum weniger als n_2 mal auftreten.

Damit liegt eine Menge von Linien vor. Bei ungenügendem Ergebnis kann das Verfahren mit modifizierter Menge P wiederholt werden: Wenn stärker zwischen Linien differenziert werden soll, sind prägende Haltestellen geeignet zu P hinzuzufügen; wenn mehrere Linien zusammengefasst werden sollen, müssen jene Haltestellen aus P entfernt werden, in denen sie sich unterscheiden.

Für die Zuordnung zukünftiger Fahrten und die Verwendung im Prototypen werden noch weitere Attribute benötigt:

10. Ermittle eine eindeutige Haltestellen-Abfolge über alle Sublinien der Linie. Hierfür kann folgender Algorithmus verwendet werden:
    - Wähle aus allen Haltestellen-Kandidaten jene, die in keiner Sublinie einen Vorgänger hat. Füge diese der Abfolge hinzu.
    - Wiederhole den vorherigen Schritt, wobei bereits in der Abfolge enthaltene Haltestellen nicht mehr als Kandidaten oder Vorgänger betrachtet werden.
    - Beende, wenn alle auftretenden Haltestellen der Abfolge zugewiesen wurden.
11. Ermittle einen Prioritäten-Wert für die Linie. Mit diesem wird bei zukünftigen Datenlieferungen beeinflusst, welche von mehreren mögliche Linien einer Fahrt zugeordnet wird. Der Wert kann je nach fachlichen Vorgaben z.B. aus der Länge der Linie oder der Häufigkeit ihres bisherigen Auftretens abgeleitet werden.
12. Ermittle eine Bezeichnung für die Linie. Diese kann sich z.B. aus Betreiber-Name sowie erstem und letzten Haltestellen-Namen zusammensetzen.

Versuchsweise Anwendungen dieses Verfahrens (u.a. auf die Fahrten von Bernmobil, Zentralbahn BLS sowie die IC und ICN der SBB) erzielten brauchbare Start-Lösungen, bei denen jedoch eine manuelle Nachbearbeitung erforderlich war. Eine vollständige Implementierung und Integration in den Prototypen war im Rahmen der Arbeit nicht möglich.

5 Auswertungen

Gegenstand dieser Arbeit ist die Internet-Publikation von grafischen und tabellarischen Auswertungen zur Pünktlichkeit im öffentlichen Verkehr. Bisher wurden die dafür erforderlichen Vorarbeiten und Grundlagen beschrieben. Dieses Kapitel widmet sich nun der Auswertungsschicht und den dort erzeugten Grafiken und Tabellen. Ein beträchtlicher Teil der vorgestellten Ideen und Entwürfe geht zurück auf eine Arbeit des Verfassers im Modul «Datenvisualisierung» des CAS Datenanalyse.[28]

5.1 Grundlegende Überlegungen zur Visualisierung von Pünktlichkeitsdaten

Bei der Wahl der Darstellungsmethode sind zwei eng miteinander verbundene Fragen zu beantworten:

1. Welche Indikatoren sollen der Auswertung zu Grunde gelegt werden?
   Wie bereits in 3.4.2 thematisiert wurde, handelt es sich bei den betrachteten Pünktlichkeitswerten um Verteilungen, die in ihrer Gesamtheit nicht darstellbar sind, sondern geeignet verdichtet werden müssen. Hierzu können verschiedene statistische Kennzahlen verwendet und auch kombiniert werden, z.B. Mittelwerte, Anteile, Perzentile. Dabei besteht folgender Zielkonflikt: einfache Indikatoren führen zu einem starken Informationsverlust, komplizierte Indikatoren sind dem Anwender schwer vermittelbar. Zu beachten ist weiterhin, dass zusätzlich zur Verteilung auch noch weitere Dimensionen (z.B. Entwicklung im Zeitverlauf, Vergleich mehrerer Unternehmen) durch die Auswertungen transportiert werden sollen.
2. Wie werden die gewählten Indikatoren visualisiert?
   Dies ist nicht vollständig von der ersten Frage zu trennen, denn nicht alle Diagrammtypen eignen sich gleich gut für alle Kennzahlen. Verbreitete Darstellungen für einfache Indikatoren sind z.B. Linien- und Balkendiagramme; für die Visualisierung von Verteilungen können z.B. Boxplots und Violin-Diagramme zur Anwendung kommen.

Ein naheliegender Indikator ist die durchschnittliche Verspätung über alle betrachteten Fahrten. Durchschnittswerte sind jedoch angesichts der starken Streuung und Schiefe der Verteilung (vgl. Abbildung 7) wenig aussagekräftig und können zudem von einzelnen Extremsituationen stark beeinflusst werden.

In der Branche ist es verbreitet, Pünktlichkeit auf Basis eines Schwellwerts zu definieren: Es wird der Anteil jener Halte ausgewiesen, die mehr als z.B. 3 Minuten verspätet waren (weitere in der Schweiz gebräuchliche Schwellwerte sind 90 Sekunden und 5 Minuten).[29] Aussagekraft und Repräsentativität sind hier deutlich grösser, zudem ist der Indikator unempfindlich gegenüber Extremwerten. Nachteilig ist die starke Verdichtung: es wird nicht differenziert zwischen Verspätungen, die knapp über dem Grenzwert liegen (z.B. 3:01 min) und solchen, die ihn beträchtlich überschreiten (z.B. 30 min), obwohl dies aus Sicht der Betroffenen äusserst relevant ist.

Um ein reichhaltigeres Bild einer Verteilung zu vermitteln, sollten Ausmass und Häufigkeit von Verspätungen gemeinsam visualisiert werden. Gebräuchliche Methoden hierfür sind Boxplots und Violin-Charts, die jedoch oft als wenig gefällig und kompliziert wahrgenommen werden. Eine Vereinfachung besteht darin, für ausgewählte Häufigkeiten (z.B. 50%, 75%, 90%, 95%) das zugehörige Verspätungsausmass in unterschiedlichen Farbtönen darzustellen.

[28] Vgl. Gutweniger (2017).

[29] Vgl. SBB (2017). In anderen Ländern sind die Schwellwerte teils deutlich höher, bei der Deutschen Bahn z.B. 6 Minuten, vgl. Wikipedia (2017).

5.2 Im Prototyp implementierte Auswertungen

Im Prototyp werden sowohl Pünktlichkeitswerte als auch die Verteilung von Verspätungen visualisiert. Zusätzlich werden die den Grafiken zu Grunde liegenden Werte tabellarisch dargestellt.

5.2.1 Visualisierung von Pünktlichkeitswerten

Der Anteil pünktlicher Ankünfte bezogen auf einen Schwellwert wird gegenüber einer anderen Dimension aufgetragen, um Vergleiche zu ermöglichen. Im Prototypen implementiert ist ein Vergleich im Jahresverlauf (Datums-Dimension), im Tagesverlauf (Zeit-Dimension) und zwischen den Unternehmen (Betreiber-Dimension).

In den ersten beiden Fällen handelt es sich um eine kardinale Vergleichs-Skala und die Darstellung erfolgt als Liniendiagramm (siehe Abbildung 12). Um eine ausreichende Auflösung zu erzielen und andererseits die Proportionen nicht zu verfälschen, ist die Skalierung der Grafik so gewählt, dass das untere Ende der Y-Achse i.d.R. nur durch vertikales Scrollen sichtbar wird. Aus diesem Grund ist die X-Achse am oberen Rand der Grafik positioniert. Um dem Betrachter Anhaltspunkte für die Grössenordnungen zu geben, werden die auftretenden Maximal- und Minimalwerte angeschrieben und zusätzlich der Median über alle Vergleichsobjekte (Kalendertage bzw. Tagesstunden) eingezeichnet.

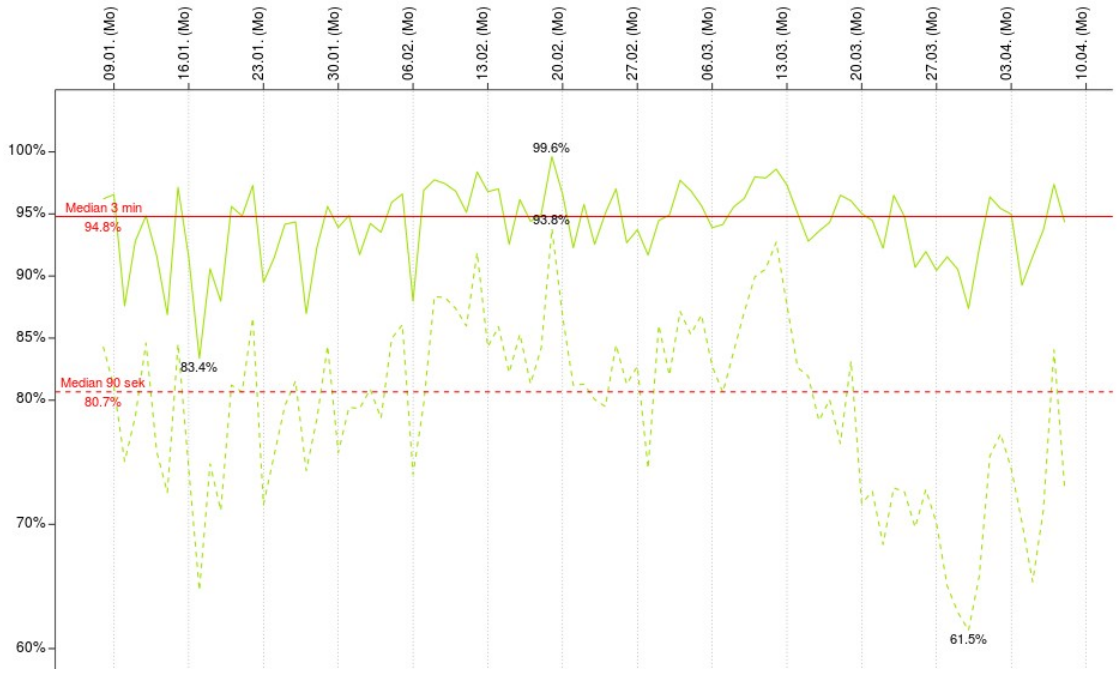

Abbildung 12: Visualisierung von Pünktlichkeitswerten im Jahresverlauf (Ausschnitt).

Beim Vergleich von Betreibern handelt es sich um eine nominal skalierte Vergleichsdimension. Hierfür wird ein um 90 Grad gedrehtes Balkendiagramm verwendet. Die Unternehmen werden dabei absteigend nach Pünktlichkeitswert sortiert. Die erzielten Pünktlichkeitswerte, der jeweilige Rang und der Median über alle Unternehmen werden dargestellt.

Abbildung 13: Vergleich der Pünktlichkeitswerte verschiedener Betreiber (Ausschnitt).

Bei beiden Darstellungsarten kann das Datums-Intervall vom Benutzer gewählt werden (bei der Darstellung des Jahresverlaufs skaliert die x-Achse entsprechend). Weiterhin ist der Schwellwert (90 Sekunden, 3 Minuten oder 5 Minuten) wählbar. Mehrere Schwellwerte können kombiniert werden.

5.2.2 Visualisierung von Verspätungsverteilungen

Die Darstellung der Verspätungsverteilung im Zeitverlauf erfolgt mit übereinanderliegenden Bändern («Ribbons»). Dabei nimmt die Farbintensität von der Mitte (Median) zu den Rändern (25/50%, 10/90%, 5/95%) ab. Zum besseren Verständnis sind diese Perzentil-Stufen am rechten Rand der Grafik angeschrieben, links findet sich die Skala für die Verspätungsminuten. Diese ist so skaliert, dass der Normalfall (Gros der Verspätungen im Bereich unter 5 Minuten) als «klein» und «undramatisch» wahrgenommen wird. Zur besseren Orientierung wird ein wählbarer Schwellwert mit einer roten Linie eingetragen – dieser hat aber keine Auswirkung auf die Berechnung der Bänder. Ebenfalls ist das Datums-Intervall vom Benutzer wählbar.

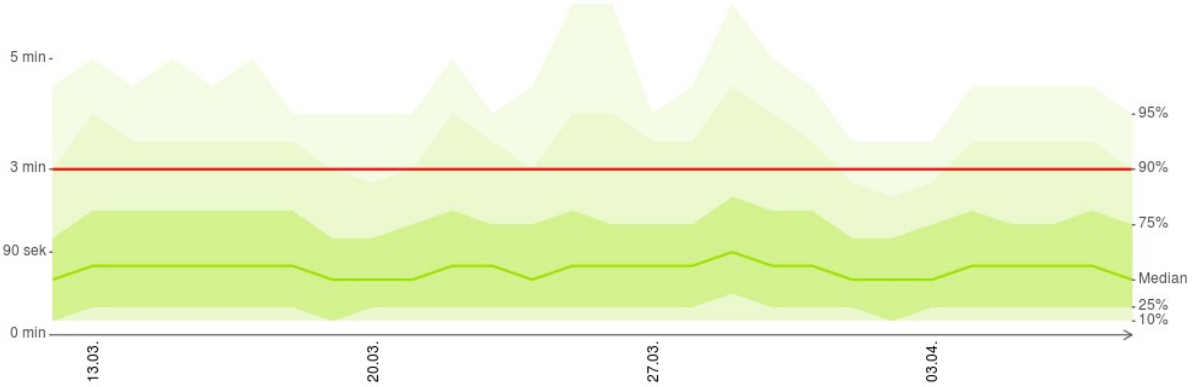

Abbildung 14: Verspätungsverteilung im Jahresverlauf (Ausschnitt).

Da Daten aus dem aggregierten Modell verwendet werden, ergeben sich genäherte Werte mit «Stufen» entlang der definierten Verspätungsniveaus. Eine Glättung könnte durch Modifikation des Berechnungsverfahrens erreicht werden (z.B. lineare Interpolation zwischen den Werten zweier benachbarter Niveaus), dies erscheint aber nicht als dringlich.

Diese Darrstellungsart ist im Prototypen für den Vergleich im Jahres- und im Tagesverlauf implementiert, für den Vergleich nominaler Werte (z.B. Betreiber) existiert sie noch nicht. Hierfür könnten übereinanderliegende Rechtecke an Stelle der Bänder verwendet werden.

5.2.3 Tabellarische Darstellungen

Für jedes Vergleichsobjekt (Betreiber, Betriebstag, Tagesstunde) kann die Verspätungssituation tabellarisch dargestellt werden. Dabei werden die absolute Zahl der Beobachtungen aufgelistet sowie die prozentualen Anteile, die auf ausgewählte Verspätungsniveaus entfallen. Diejenigen Niveaus, die den selektierten Schwellwerten entsprechen, sind farblich hervorgehoben. Auch hier ist das Datumsintervall vom Benutzer wählbar.

	Stunde	Gesamtzahl	>90 sek	>3 min	>5 min	>10 min	>15 min	>30 min
1	0-1	36607	36.0%	18.2%	8.8%	2.2%	0.8%	0.2%
2	1-2	6939	41.1%	22.9%	12.7%	3.9%	1.8%	0.8%
3	2-3	2269	39.8%	17.3%	6.2%	0.5%	0.3%	0.1%
4	3-4	2534	39.2%	17.9%	7.4%	0.7%	0.2%	0.0%
5	4-5	4422	29.5%	9.9%	4.3%	0.7%	0.5%	0.0%
6	5-6	41888	23.9%	6.2%	1.9%	0.3%	0.1%	0.0%
7	6-7	79620	24.4%	5.7%	1.4%	0.3%	0.1%	0.0%
8	7-8	85617	31.5%	8.6%	2.0%	0.3%	0.1%	0.0%

Abbildung 15: Tabellarische Darstellung der Verspätungsverteilung (Ausschnitt).

5.3 Informationsarchitektur und Navigationskonzept

Die Benutzeroberfläche von www.*puenktlichkeit.ch* ist gegliedert in

- Eine Menüleiste am oberen Rand.
- Elemente für die Benutzereingabe im linken Viertel.
- Die resultierende Auswertung rechts daneben.
- Eine Fusszeile.

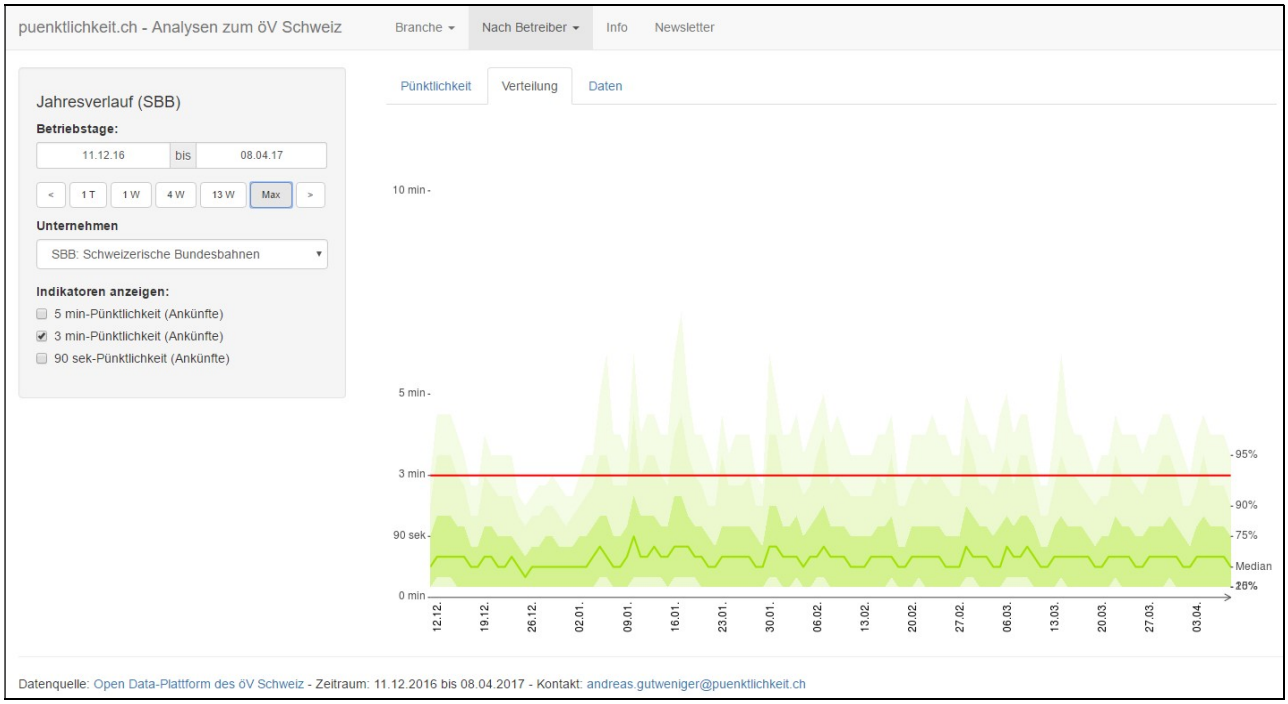

Abbildung 16: Aufbau von *puenktlichkeit.ch*.

Derzeit sind folgende Funktionen über das **Menü** auswählbar:

- Bezogen auf die Branche insgesamt:
  - Vergleich im Jahresverlauf.
  - Vergleich im Tagesverlauf.
  - Betreibervergleich.
- Bezogen auf einzelne Unternehmen der Branche:
  - Vergleich im Jahresverlauf.
  - Vergleich im Tagesverlauf.
- Informationen über die Website.
- Möglichkeit, einen E-Mail-Newsletter zu abonnieren.

Die im linken Bereich dargestellten **Elemente zur Benutzereingabe** sind abhängig vom gewählten Menüpunkt.

Über zwei oder drei Reiter («Tabs») kann bei jeder **Auswertung** die Darstellungsart (Pünktlichkeitswerte, Verspätungsverteilung, Tabellarische Darstellung) gewählt werden.

Die **Fusszeile** gibt Auskunft über Datenherkunft, Datenstand und Kontaktmöglichkeit.

5.4 Realisierung mit R Shiny

R Shiny sieht eine Aufteilung jeder Applikation in einen «UI»- und einen «Server»-Teil vor.[30]

Im Programmcode zum **UI** wird die Benutzeroberfläche festgelegt: Struktur der Oberfläche, Menü, Eingabe-Elemente, anzuzeigender Text etc. Die Library «Shiny» stellt hierfür diverse Funktionen zur Verfügung, die auf dem Bootstrap-Framework[31] basieren und letztlich HTML-, CSS- und JavaScript-Code generieren. Die Programmierung des UI besteht im Wesentlichen aus einer Verschachtelung dieser Funktionen (Applikation enthält Seite enthält Layout enthält Panel enthält …).

```
shinyUI(
  fluidPage(
    sidebarLayout(
      conditionalPanel("['GU','GJ','GT','BV','BU','BJ','BT','LU','LJ','LT'].includes(input.menu1)",
        sidebarPanel(width=3,
          h4(textOutput("Titel")), p(" "),
          conditionalPanel("['GU','GJ','GT','BV','BU','BJ','BT','LU','LJ','LT'].includes(input.menu1)",
            dateRangeInput("dateRange", language = "de", separator='bis', label = "Betriebstage:", format =
            actionButton("Rueck", "<", style='font-size:80%'),
            actionButton("Tag1", "1 T", style='font-size:80%'),
```

Abbildung 17: Ausschnitt aus dem UI-Programmcode

Der Programmcode zum **Server** definiert die serverseitig auszuführende Funktionalität, d.h. alle Datenbank-Zugriffe, Berechnungen sowie die Generierung dynamischer Grafiken, Tabellen und Texte. Zu den grossen Vorzügen von Shiny gehört die «reaktive Programmierung»[32]: Basierend darauf, welche Eingabe-Elemente vom Benutzer verändert wurden, ist der Server selbständig in der Lage zu entscheiden, welche Teile des Codes für Neuberechnungen ausgeführt werden müssen und welche bereits existierenden Ergebnisse unverändert beibehalten werden können.

Dies stellt grundsätzlich einen mächtigen Mechanismus zur Optimierung von Antwortzeiten dar, setzt jedoch voraus, dass der Programmcode entsprechend modularisiert wird: Wenn z.B. die Generierung

[30] Vgl. RStudio (2014).

[31] Bootstrap ist ein verbreitetes Framework zur Entwicklung responsiver Web-Sites, vgl. getbootstrap.com.

[32] Vgl. RStudio (2014b) und RStudio (2015).

einer Grafik und eine aufwändige Datenbank-Abfrage in derselben Funktion implementiert sind, so wird die Datenbank-Abfrage auch dann ausgeführt, wenn ein Parameter verändert wurde, der lediglich für die Grafik relevant ist – mit entsprechenden Auswirkungen auf die Antwortzeit. Werden die beiden Schritte dagegen auf separate Funktionen verteilt, so kann bei der Neuerstellung der Grafik auf die bereits vorhandenen Ergebnisse der letzten DB-Query zurückgegriffen werden.

In Abbildung 18 ist an einem Beispiel dargestellt, wie im Prototypen «performance-optimiert» modularisiert wird und welche Aufrufbeziehungen dabei entstehen.

```
BetreiberJahresverlaufData <- reactive({

  # Holt die Pünktlichkeitsdaten eines Betreibers für alle Betriebstage aus der Datenbank
  # Erneute Ausführung nur, wenn der Benutzer den Betreiber ändert
})

BetreiberJahresverlaufDataP <- reactive({

  # Transformiert die Daten eines Betreibers in die für ein Pünktlichkeitsdiagramm notwendige Struktur
  # Rechenaufwendig (Casting, Melting), braucht aber nur für diesen Darstellungstyp ausgeführt zu werden
  # Erneute Ausführung nur, wenn Pünktlichkeitsdiagramm angezeigt werden soll und der Benutzer den Betreiber ändert
  # Verwendet BetreiberJahresverlaufData()
})

output$BetreiberJahresverlaufP <- renderPlot({

  # Wält aus den vorliegenden Daten die benötigten Betriebstage und Indikatoren aus und generiert ein Pünktlichkeitsdiagramm
  # Erneute Ausführung, wenn Pünktlichkeitsdiagramm angezeigt werden soll und der Benutzer einen der folgenden Parameter ändert:
  # Datums-Intervall, Betreiber, dargestellte Schwellwerte
  # Verwendet BetreiberJahresverlaufDataP()
})

output$BetreiberJahresverlaufV <- renderPlot({

  # Wält aus den vorliegenden Daten die benötigten Betriebstage und Indikatoren aus und generiert ein Verteilungsdiagramm
  # Erneute Ausführung, wenn Verteilungsdiagramm angezeigt werden soll und der Benutzer einen der folgenden Parameter ändert:
  # Datums-Intervall, Betreiber, dargestellte Schwellwerte
  # Verwendet BetreiberJahresverlaufData()
})

BetreiberJahresverlaufDT <- reactive({

  # Transformiert die Daten eines Betreibers in die notwendige Struktur und formatiert eine Tabelle
  # Rechenaufwendig (Casting, Formatierung), braucht aber nur für diesen Darstellungstyp ausgeführt zu werden
  # Erneute Ausführung nur, wenn Tabelle angezeigt werden soll und der Benutzer den Betreiber ändert
  # Verwendet BetreiberJahresverlaufData()
})

output$BetreiberJahresverlaufT <- DT::renderDataTable({

  # Wählt die Tabellenzeilen mit den benötigten Betriebstagen aus und hebt die Spalten gemäss gewählten Schwellwerten hervor
  # Erneute Ausführung nur, wenn Tabelle angezeigt werden soll und der Benutzer einen der folgenden Parameter ändert:
  # Datums-Intervall, Betreiber, dargestellte Schwellwerte
  # Verwendet BetreiberJahresverlaufDT()
})
```

Abbildung 18: Antwortzeit-optimierte Modularisierung des Server-Programmcodes.

6 Erkenntnisse aus der Erstellung des Prototypen

Im Rahmen dieser Arbeit wurden zahlreiche Auswahl- und Design-Entscheide getroffen und verschiedene Konzepte, Produkte und Technologien wurden eingesetzt. Die praktische Anwendung am Prototypen hat es ermöglicht, unmittelbar Erfahrungen damit zu sammeln. Die wichtigsten Erkenntnisse sollen hier vorgestellt werden.

6.1 Open Data

Die Bereitstellung von Open Data erlebt derzeit einen Boom. Nicht immer deckt sich dabei der Anspruch der Herausgeber (z.B. Transparenz, Erschliessung neuer Nutzenpotentiale, Förderung eines Öko-Systems) mit der Qualität der Umsetzung (die Veröffentlichung von Daten schafft keinen Mehrwert, solange nicht sichergestellt ist, dass diese von anderen nutzenstiftend verwendet werden kann). Das W3C hat erst kürzlich «Best Practices» zur Publikation von Daten im Web veröffentlicht[33] – bis diese sich als «common practice» etabliert haben, wird es wohl noch einige Zeit dauern.

Für den konkreten Fall von opentransportdata.ch kann Folgendes festgestellt werden:

- Der Zugang zu den Daten funktioniert problemlos. Es existiert ein «Bulk Download» über den Daten zuverlässig und zeitnah bereitgestellt werden. Schnittstellen und Datenmodelle sind stabil und erfüllen ihren Zweck. Deskriptive und strukturelle Metadaten sind ausreichend dokumentiert.
- Eine Support-Organisation ist vorhanden. Auf Anfragen wird schnell reagiert. Ebenfalls existiert eine Online-Community, die aber kaum frequentiert wird.
- Die Herkunft und Entstehung der Daten (W3C: «data provenance») ist nicht ausreichend nachvollziehbar. Es kann nicht festgestellt werden, welche Systeme und Organisationen daran beteiligt sind, wie die Daten erhoben und transformiert wurden und welche Regeln und Annahmen dabei gelten. Dieser Punkt ist auch deshalb kritisch, weil die Daten offensichtlich aus unterschiedlichen Quellen stammen und möglicherweise nur eingeschränkt miteinander vergleichbar sind (wie sich am Beispiel der Linien-Information gezeigt hat).
- Aussagen zur Datenqualität sind nicht im gewünschten Ausmass vorhanden. Dies betrifft zum Beispiel die Verlässlichkeit und Präzision der gelieferten Werte, ihre Vollständigkeit, geographische und zeitliche Abdeckung. Die für die Zwecke dieser Arbeit aufgestellte «Korrektheits-Hypothese» ist somit schwer zu untermauern – es haben sich bisher aber auch noch keine gegenteiligen Indizien gezeigt.

6.2 Technologieauswahl und Systemarchitektur

Die Verfolgung eines «Open Source First, Cloud First»-Ansatzes hat sich bewährt:

- Das Aufsetzen der Cloud-Instanzen und die Installation der erforderlichen Software war trotz fehlenden Vorkenntnissen einfach und verlief schnell und problemlos. Dies auch dank der sehr guten Beschreibungen und Erfahrungsberichte im Web.
- Betrieb und Administration sind einfach und zuverlässig.
- Die Skalierung der Cloud-Instanzen funktionierte bisher problemlos.
- Für alle eingesetzten Technologien (Ubuntu, MySQL, R, Shiny, nginx) gibt es eine sehr umfangreiche Community, so dass Lösungen zu Problemen aller Art mit kurzer Web-Recherche zu finden sind.

[33] Vgl. W3C (2017).

- Die Kosten im ersten Jahr («Free Tier») sind vernachlässigbar. Ab dem zweiten Jahr würden sie sich nach grober Schätzung auf etwas 50-80 USD pro Monat belaufen.

Kritisch zu bewerten ist die Auswahl von R Shiny für die Web-Applikation. Sowohl im UI- als auch im Server-Teil blieben Entwicklungseffizienz und Wartbarkeit sehr deutlich hinter den Erwartungen:

- Die UI-Programmierung im «funktionalen Paradigma» mit einer starken Schachtelung von Funktionsaufrufen ist ausserordentlich fehleranfällig, zumal die sehr laxe Code- und Typprüfung von R und die Vieldeutigkeit zahlreicher Funktionen und Parameter dazu führen, dass nur selten eine Fehlermeldung erfolgt.
- Die von Shiny für das GUI verwendeten Konzepte (z.B. «Panels»), sind nicht immer stimmig und konsistent. Die Dokumentation ist gut lesbar und anschaulich, aber oft von geringer formaler Stringenz.
- Wie in 5.4 beschrieben wurde, erfordert das «Reactive Programming»-Paradigma ein spezielles Vorgehen bei der Modularisierung des Codes. Dieses ist schlecht vereinbar mit einer auf Wartbarkeit ausgerichteten Code-Gestaltung. Tatsächlich ergeben sich viele Wiederholungen von gleichen oder ähnlichen Passagen (hohe Code-Redundanz).
- Das Debugging der Shiny-Applikation erwies sich als schwerfällig.[34] Als Gegenmassnahme empfiehlt sich, möglichst grosse Teile des Codes (z.B. DB-Abfragen, Berechnungen, Generierung von Grafiken) zunächst als «normale» R-Skripte bis zur Produktionsreife zu entwickeln und so spät wie möglich in die Shiny-Applikation zu integrieren. Diesem Workaround sind nicht zuletzt aufgrund von «Reactive Programming» jedoch Grenzen gesetzt.

6.3 Entwicklungswerkzeuge

Als sehr positiv ist der R Studio-Server zu bewerten: Die Web-Oberfläche unterscheidet sich in Bedienung und Funktionsumfang fast nicht vom bekannten R Studio-Client und funktioniert sehr zuverlässig (inkl. Versionsverwaltung mit Git). Es ist somit einfach möglich, vom Browser aus direkt auf der Cloud-Instanz zu entwickeln – ohne die Notwendigkeit, Programmcode zu replizieren und ohne die Nachteile einer Remote-Desktopverbindung.

Ebenfalls bewährt hat sich der Verzicht auf ein separates ETL-Werkzeug. Selbst bei den umfangreichen Transformationen zur Linienbildung erwies sich die Kombination von R und Inline-SQL als gute Wahl. Sehr vorteilhaft war der durchgängige Einsatz dieser beiden Sprachen über alle Phasen und Einsatzzwecke (Konzeption, Test/Evaluation, ETL, Web-Applikation) – dies ermöglichte einen starken «Reuse» sowohl von Gedanken und Konzepten als auch von Code-Teilen.

Möglicherweise wäre der Einsatz eines Profilers sinnvoll gewesen, um sich leichter einen Überblick über den vorhandenen «Datenschatz» zu verschaffen und die Qualitätssicherung zu unterstützen.

6.4 Datenmodelle und ETL-Prozesse

Die 2-Tier-Datenarchitektur hat sich bewährt: die Komplexität konnte auch ohne «Core» gut beherrscht werden und Anpassungen am Modell liessen sich jeweils rasch und einfach umsetzen. Auch die Integration der zusätzlichen Dimension «Linie» ins Modell wird – sobald sich die notwendigen Daten ermitteln lassen – einfach möglich sein. Richtig war weiterhin der Entscheid, einfache Berechnungen und Typ-Konversionen bereits beim Laden in den Import-Bereich durchzuführen – dies hat einen zusätzlichen Transformations-Schritt erspart und zu keinen Nachteilen geführt.

[34] RStudio (2014c) beschreibt diverse Methoden für Fehlersuche, Tracking und Fehlerbehandlung bei Shiny-Applikationen. Im vorliegenden Fall erwies sich manches davon jedoch als langsam, aufwendig und ineffizient.

Die Auswertungs-Datenbank performt gut. Abfragen auf der feinsten Granularitätsstufe werden bei angemessener Selektivität rasch verarbeitet. Für Auswertungen auf grossen Teilmengen bewährt sich die Vorberechnung der aggregierten Tabellen unter Verwendung der Verspätungsniveaus. Ausser einer einfachen Indexierung waren Datenbank-seitig keine Performance-Massnahmen notwendig.

Ausgezahlt hat sich auch die Investition in robuste und einfach zu handhabende Import-Skripts. Diese laufen bereits seit mehreren Wochen mit hoher Stabilität. Die vollständige Verarbeitung einer Tageslieferung (Datei-Bezug, Einlesen in Import-Bereich, Einlesen in Fakten-Tabelle, Bildung von 2 Aggregationsstufen, Komprimieren und Archivieren der Datei) dauert weniger als 2 Minuten.

6.5 Nutzen der erstellten Auswertungen

Die Auswahl und Gestaltung der Visualisierungen orientierte sich zu grossen Teilen am Leitsatz «Zeigen, was möglich ist.». Ein Einbezug von potentiellen Nutzern hat bisher nicht stattgefunden, noch nicht einmal die Zielgruppe ist klar umrissen. Tendenziell richten sich die erstellten Auswertungen an ein Experten-Publikum, das sowohl mit der Fachlichkeit als auch den Grundlagen der Statistik gut vertraut ist. Allerdings sind auch deren Anforderungen bisher nicht erhoben und es muss davon ausgegangen werden, dass der vorliegende Stand zwar «interessant» aber nicht uneingeschränkt «nützlich» ist. Im Falle einer Weiterentwicklung besteht an dieser Stelle sicherlich Handlungsbedarf.

7 Zusammenfassung und Ausblick

Gegenstand dieser Semesterarbeit ist die Internet-Publikation von grafischen und tabellarischen Auswertungen zur öV-Pünktlichkeit unter Verwendung der Daten von *www.opentransportdata.swiss.*

Sie orientiert sich an folgenden Zielsetzungen:

1. Zeigen, wie mit wenig Aufwand ein nützliches System erstellt werden kann.
2. Prüfen, was mit vorhandenen Daten und gewählten Technologien machbar ist.
3. Auswertungen sollen leicht anpassbar und erweiterbar sein.
4. Lizenz- und Betriebskosten sollen möglichst niedrig sein.

Folgende Ergebnisse wurden erzielt:

- eine Auswahl geeigneter Technologien (Ubuntu, MySQL, R, Shiny, nginx) und Services (AWS RDS und EC2, Atlassian Bitbucket, Mailchimp, Google Analytics),
- die Erstellung und Beschreibung einer Gesamtarchitektur für die Lösung,
- eine 2-Schicht-Datenarchitektur bestehend aus einem Import-Bereich und einer Auswertungs-Datenbank mit drei Aggregationsstufen,
- die zugehörigen Datenmodelle,
- ETL-Prozesse für den Datenbezug von der Open Data-Plattform, das Laden des Import-Bereichs, das Laden der feingranularen Fakten und die Berechnung der Aggregate,
- eine Ablaufsteuerung zur Automatisierung dieser Prozesse,
- eine Heuristik für die Bildung der Linien-Dimension aus den gelieferten Daten,
- die Konzeption von grafischen und tabellarischen Auswertungen zu Pünktlichkeitswerten und Verspätungsverteilungen,
- die prototypische Umsetzung der meisten dieser Modelle und Konzepte als Web-Applikation *puenktlichkeit.ch*,
- eine Reflektion der dabei gesammelten Erfahrungen namentlich zu Open Data, Technologien, Werkzeugen und Datenarchitektur.

Eine Weiterführung dieser Arbeiten erscheint sinnvoll entlang von 3 Stossrichtungen:

1. Es bestehen zahlreiche Ideen für zusätzliche oder verbesserte Auswertungen und Features von *puenktlichkeit.ch*, u.a. Auswertungen nach Linien und Strecken, interaktive Grafiken, geographische Darstellungen (Landkarten), Cockpits, die Analyse von Verspätungs-Entstehung und -Abbau, die Berücksichtigung von Zugsausfällen.
2. In absehbarer Zeit notwendig wird ein Wechsel des Arbeitsmodus vom «Technology Push» zum «Demand Pull»: Dies beinhaltet den Einbezug von möglichen Nutzern zwecks Evaluation der bestehenden Funktionalität und Konkretisierung von Erweiterungsideen. Im Fokus stehen dabei die Fachexperten aus den Unternehmen, Verbänden und Behörden der Branche.
3. Die bisherigen Auswertungen sind rein deskriptiver Natur. Es erscheint interessant, den vorliegenden umfangreichen «Daten-Schatz» auch für die Herleitung von Erklärungs- und Prognosemodellen zu verwenden. Mögliche Fragestellungen sind: Welche Verspätungen sind leicht abbaubar und bleiben lokal begrenzt? Welche tendieren dazu, sich aufzuschaukeln und im Netzwerk auszubreiten? Wo gibt es «Tipping Points» und kritische Momente? Lassen sich Frühindikatoren für bestimmte Verspätungs-Szenarien identifizieren? Ist auf deren Grundlage gar eine bessere Verspätungs-Prognose möglich?

8 Abbildungsverzeichnis

9 Literaturverzeichnis

Amazon (2017): IAM Best Practices, Online: http://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html, abgerufen am 29. März 2017

Banner, Matt (2016): Aws-install.sh - Shiny server on Amazon AWS EC2 in 5 minutes, online: http://whatsgood.io/blog/2016/04/27/shiny-server-on-amazon-aws-in-five-minutes/, abgerufen am 29. März 2017

BAV (2016): Bund schafft Voraussetzungen für Apps mit Echtzeitinformationen zum öffentlichen Verkehr, Medienmitteilung des Bundesamts für Verkehr vom 1. Dezember 2016, online: https://www.admin.ch/gov/de/start/dokumentation/medienmitteilungen.msg-id-64744.html, abgerufen am 8. April 2017.

Beech, Greg (2006): Inline SQL vs. Stored procedures, Online: http://gregbee.ch/blog/inline-sql-vs-stored-procedures, abgerufen am 29. März 2017

Brupbacher, Marc (2016): So pünktlich ist ihre VBZ-Linie, online: http://interaktiv.tagesanzeiger.ch/2016/so-puenktlich-ist-ihre-vbz-linie/, abgerufen am 3. April 2017.

Burns, Patrick (2012): The R Inferno, Verlag lulu.com, sowie online: http://www.burns-stat.com/pages/Tutor/R_inferno.pdf, abgerufen am 30. März 2017.

CKAN (2013): API Guide, online: http://docs.ckan.org/en/latest/api/, abgerufen am 29. März 2017.

Gerny, Daniel (2016): Das Smartphone wird zum wichtigsten Verkehrsmittel, in: Neue Zürcher Zeitung vom 2. Dezember 2016, online: https://www.nzz.ch/schweiz/aktuelle-themen/open-data-das-smartphone-wird-zum-wichtigsten-verkehrsmittel-ld.131923, abgerufen am 8. April 2017.

Gritts, Mitchell (2016): Shiny Server on AWS, Online: http://mgritts.github.io/2016/07/08/shiny-aws/, abgerufen am 29. März 2017

Gutweniger, Andreas (2017): Kommentierte Datengrafik: Verspätungsanalyse der S-Bahn-Linie 3 der BLS AG, unveröffentlichtes Arbeitsergebnis aus dem Modul «Datenvisualisierung» des CAS Datenanalyse an der Berner Fachhochschule. (das Dokument ist dieser Arbeit beigefügt).

Open Data Plattform öV Schweiz (2016): Verwendung des CKAN-API, online: https://opentransportdata.swiss/de/cookbook/verwendung-des-ckan-api/, abgerufen am 29. März 2017.

Open Data Plattform öV Schweiz (2016b): Cookbook: Ist-Daten, online: https://opentransportdata.swiss/de/cookbook/ist-daten/, abgerufen am 7. April 2017.

RStudio (2014): How to build a shiny app, online: https://shiny.rstudio.com/articles/build.html, abgerufen am 7. April 2016.

RStudio (2014b): Reactivity: An overview, online: https://shiny.rstudio.com/articles/reactivity-overview.html, abgerufen am 7. April 2016.

RStudio (2014c): Debugging Shiny applications, https://shiny.rstudio.com/articles/debugging.html, abgerufen am 8. April 2017.

RStudio (2015): How to understand reactivity in R, online: https://shiny.rstudio.com/articles/understanding-reactivity.html, abgerufen am 7. April 2016.

SBB (2017): Hintergrund-Dossier Kundenpünktlichkeit, online: https://company.sbb.ch/de/medien/dossier-medienschaffende/kundenpuenktlichkeit.html*, abgerufen am 7. April 2017.*

Schubert, Johannes (2017): Zugfinder: Aktuelle Zugpositionen und Statistiken deutscher Fernverkehrszüge, online: www.zugfinder.de*, abgerufen am 9. April 2017.*

Starke, Gernot; Hruschka, Peter (2011): Software-Architektur kompakt, Spektrum, Heidelberg, 2. Auflage, 2011.

Ullius, Markus (2005): Verwendung von Eisenbahnbetriebsdaten für die Schwachstellen- und Risikoanalyse zur Verbesserung der Angebots- und Betriebsqualität. Diss., Zürich, Eidgenössische Technische Hochschule, Institut für Verkehrsplanung und Transportsysteme.

W3C (2017): Data on the Web Best Practices. W3C Recommendation 31 January 2017, online: https://www.w3.org/TR/dwbp/*, abgerufen am 8. April 2017.*

Wikipedia (2017): Pünktlichkeit, online: https://de.wikipedia.org/wiki/P%C3%BCnktlichkeit_(Bahn)*, abgerufen am 7. April 2017.*

10 Anhang: Beilagen zur Semesterarbeit

Gemeinsam mit dieser Arbeit werden die folgenden Beilagen abgegeben.

Source Code

- ui.r Benutzeroberfläche der Shiny-Applikation
- server.r Serverseitiger Teil der Shiny-Applikation
- etl_ckan-import.R ETL-Skript zu Phase 1 (Bezug der Dateien vom Open Data-Portal)
- etl_istfile2stg.R ETL-Skript zu Phase 2 (Laden des Import-Bereichs)
- etl_stg2ftc.R ETL-Skript zu Phase 3 (Laden der Fakten-Tabellen)
- etl_import_all.R Skript für den Import aller vorliegenden Dateien in die Datenbank
- linien_konstruktion.R Heuristik zur Bildung von Linien aus den vorliegenden Daten

Unveröffentlichte Quellen

- Datengrafik.pdf Gutweniger (2017): Kommentierte Datengrafik aus dem CAS DA

11 Selbständigkeitserklärung

Ich bestätige, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der im Literaturverzeichnis angegebenen Quellen und Hilfsmittel angefertigt habe. Sämtliche Textstellen, die nicht von mir stammen, sind als Zitate gekennzeichnet und mit dem genauen Hinweis auf ihre Herkunft versehen.

Andreas Gutweniger, Bremgarten bei Bern, 10. April 2017