TONY LAIDIG

QUANTITATIVE CONCEPTS FOR 'GOOD' TRANSIT SCHEDULE DATA

Disclaimer.

This guide is the result of a career's worth of experience with transit data. In the process, the author has worked with data from dozens of transportation agencies, of varying sizes, both in the USA and internationally. The information within should not be considered the result of experience with any one particular entity, and where not cited, the opinions expressed are of the primary author alone and not of any other individual, company, agency, government, or interstellar federation.

Introduction

The last decade has seen an explosion in the availability of data, and public transit is no exception. The development of the General (originally Google) Transit Feed Structure (GTFS) in 2005 brought to light data that was previously available to the public only on paper. Previously, when data was available for interchange internally, it was less accessible– usually in more complex (e.g. TransXchange, TCIP), proprietary (e.g. TSDE), or home-grown formats. GTFS brings transit schedule data into a modern, open, and easily-consumable world by presenting a simplified de-facto standard that many developers can use without the knowing the intricacies of transit operations. Since that time, the number of publicly available feeds has grown exponentially, as has the number of ’outsiders’ working with transit datasets has grown tremendously, and a number of applications using this data have been produced.

’Outsiders’ may refer to those within an organization acting as internal clients.

This document is intended to provide a statring point for a dialog between the parties involved in transit data creation and usage through practical qualitative concepts. Unlike many practical guides, it is intended to be useful at any stage of project development, even, and especially at the design stage. It is not intended to supplant the existing official GTFS documentation[1], provide anything more than a brief introduction to scheduling practice, or be a survey of the practice in ITS systems implementation and engineering[2].

Why Schedule?

It is quite common for the layman to ask, that a world of real-time information, what purpose does a schedule serve? In short, a schedule models what happens ’on the ground’ when transit service is delievered. TCRP Report 135 (perhaps the best freely available guide on the topic of transit scheduling) defines four major groups that benefit from the existence of (realistic) schedules:

1. To customers, a schedule provides the essential information needed to plan a trip, defines the arrival and departure times

[1] https://developers.google.com/transit/gtfs/reference

[2] There are better
sources on that, such as
http://www.fta.dot.gov/documents/2010TransitITSArch
_08.29.2011.pdf

and the time the trip will take, makes sufficient capacity of service available so that the customers' trip will be comfortable, and ensures that customers will arrive at their destination at the promised time.

2. To operators, scheduling defines the workday. Operators are the front line in terms of dealing with customers, and the interaction can be affected by [realistic] running and layover times ... Good schedules can reduce the stress inherent in this job, thus improving morale and minimizing absenteeism.

3. To transit agencies, scheduling puts reliable service on the street where it will be most utilized. In addition, scheduling provides data and information to support other sections such as Marketing, Planning, Operations, Administration, and many downstream systems like AVL, APCs, voice annunciators, trip planners, and real time information systems.

4. To general managers and chief financial officers, scheduling has major impacts on the quality and cost of operations... Scheduling is the brain of the transit organism...

For these four groups, the degree to which the schedule actually models reality plays a large part in its usefulness. The concepts discussed in Chapter 2 are intended to improve the final result of this model's representation of reality.

A PACT for Good Data

At many agencies, data on schedules already exists, and new projects are often thought up believing that existing data will work without major rework. Sadly, is often not the case. This chapter presents four crucial qualitatitive concepts that are important in describing the qualities of a system’s data. These are Precision, Accuracy, Concurrency, and Transparency, or, in manager speak, PACT. By reviewing the quality of the datasets involved according to these four concepts, project teams can address possible concerns before beginning development, with the aim of reducing errors during the implementation stage. The quantification of these concepts is not addressed. The exmples presented use an ubiquitous dataset in almost every transit data project, a dataset of stops.

Precision and Accuracy

Statisticians and Geospatial scientists use two terms primarily to describe the quality of a dataset— *accuracy* and *precison*. The *accuracy* of a dataset is the degree of closeness that a record or observation matches the real-world value. *Precision* refers to the level of refinement available in the dataset to store said record. A stop dataset provides a simple case for differing levels of precision and accuracy, and these two concepts are described in the margin.

A dataset can be accurate but not precise, precise but not accurate, neither, or both. Some organizations conduct expensive, highly detailed stop surveys which are both precise and accurate at the time of their creation, but these lose accuracy over time. It is not uncommon for the very first survey to be considered only as part of a capital project, and the organization in question believes that the work is done. This, however, is often a folly. Due to changes on the ground, accuracy of a stop dataset degrades over time if it is not updated. Streets are paved and closed, businesses open and close, and in some areas entire neighborhoods appear overnight.

Precision of an example stop dataset, in ascending precision:

- A stop is at Main and 1st Street.
- A stop is at Main and 1st Street, Northbound.
- A stop is at Main and 1st Street, Northbound, NE corner.
- A stop is at Main and 1st Street, Northbound, NE corner, with coordinates X, Y.
- A stop is at Main and 1st Street, Northbound, NE corner, with coordinates X.XXXXX, Y.YYYYY, served by Route 101.

Accuracy of the example stop dataset is dependent various factors, including:

- Notification of changes
  - e.g. “The stop at Main and 1st was removed due to construction”
- Master database update frequency
  - e.g. “Stop changes are entered into the database within 1 week.”
- Frequency of exports and releases
  - e.g. ’GTFS is generated at least every 4 months.”

Concurrency

While precision and accuracy are important when considering a project that only uses, or heavily relies upon one dataset. The elephant in the room in projects that draw upon data from multiple systems is the *concurrency* of that data. Concurrency, in this case, refers to the ability to join together, reliably, the data from one or many systems[3]. A medium to large-size public transport operation may have an IT workflow of many stages, covered by different departments or even different organizations or agencies, public and private. The less that the systems involved are concurrent, the more difficult building a new system upon those projects will be. The design of concurrent systems often entails finding reliable techniques for coordinating their execution, data exchange, and execution scheduling to minimize confusion and maximise utility[4]. Data from systems originally designed to work independently may need some finesse to properly match.

The Enemies of Concurrency

While it is difficult to quantify concurrency, it is possible to suggest characteristics of 'weaker' data sources that should be validated before relying upon them. Some of these are:

- Possibility of unsanitized / unvalidated data entry
  - If an operator can enter data that is invalid or non-sensical, then the system that uses it must take this possibility into account. Policies may need to be put in place to ensure correct entry.
- Long update or synchronization intervals
  - A dataset that models activity on the ground (e.g. bus stops) that is only updated every several years is likely to have errors.
- "Static includes"
  - Systems may rely on one particular file as an input not coming from the same repository as other souces; it may be either as an export from an otherwise unconnected system, or a hand-updated file. If policies for updating the file in question (e.g. on a regular basis or simultaneously with other exports), the data in question may fall out of concurrency.
- No / limited versioning
  - Datasets should clearly include the times they were updated; thus, when two datasets are joined they can be sufficiently linked.

[3] Systems may not even be concurrent with themselves. Consider a large organization that builds their schedule in phases; one area may be ready for production while the rest of the network is being worked on. For example, Routes 1 and 2 stop at 1st and Main St. Route 1 has its schedule generated before Route 2. When Route 2 is scheduled, the scheduler has been notified that there will be long term construction at 1st and Main and buses cannot stop there; she then removes the stop for Route 2 only. When data is export from the scheduling system, Route 1 will still be recorded as stopping at 1st and Main, even though the network has been changed.

[4]

Transparancy

Hopefully, when working with data from an existing system, there is documentation on what that data means. Often, however, when the data has only been used for only one purpose or within one organization or department, documentation is not complete. *Transparency* refers to the degree to which the data conforms to the documentation provided.

- Poor and/or obsolete documentation
  - After implementation, some systems start off well-documented, but as time goes on and minor changes are made to both the data and the usage of that data. Documentation is not always updated in accordance. Well-intentioned users may extend the existing data structure beyond the original design by adding additional information that does not strictly conform to original specifications.This information may or may not be passed down through word of mouth, and is often internalized to the point that when asked, the data owners suddenly recall the difference.
  - Organizations often take advantage of 'turn-key' systems that include an openly-accessible database component often add additional functionality 'around the edges.' It is the author's experience that these additions are rarely documented.
- Special cases
  - IT groups may tailor the same application for different users performing the same task within the same organization, often based upon varying user preferences.

* One operating division may prefer that their route numbers begin with a leading zero (e.g 01 vs 1), while another may not. Beware special cases such as this one.

- Excessive room for interpretation and/or lack of usage conventions.
  - If a data model allows for the same data can be stored in two places, it is often the case that different users see the data differently. For example, if a stop dataset model has fields for both STOP_ID (intented to be a normalized key) and STOP_CODE (intended to be a short, but human-readable descriptor) and users fail to understand this difference, users at may choose to populate these fields in ways that are not consistent across the dataset. A very basic example is presented in the margin.

STOP_ID	STOP_CODE
1	1001
2	1002
3	3
4	4

- Ambiguous, Undefined and/or complex key structure
  - a 'Primary Key' uniquely defines the characteristics of each record in a dataset, and has to consist of characteristics that are

  unique. If the key structure for a dataset is not clearly defined or not enforced, making the proper relations within the dataset is more difficult and concurrency may be affected.

- Data intended for human visual consumption.
  - Input data should be machine readable, not just human readable. If a system has previously been used as the final point in a data flow with the sole aim of output for human interpretation, it is likely that in places data does not accurately conform to its specification. This, when combined with unvalidated entries, can be especially confusing during implementation.
- Unclear methods of interpolation.
  - When data from one system is interpolated by a "black box" process, the results are often unclear.

A common case of this problem is the interpolation of stop times from AVL systems. There are several methods for determining the time when a bus passed by a particular stop. If a system only presents one time, was it the arrival or the departure time? Was there any interpolation involved in calculating the time, or is it simply the time of the position nearest to the stop in question?

- Specifications that do not agree with established practice or common nomenclature
  - Developers are notorious for not reading documentation. If a format specification happens to borrow a term from the common nomenclature and use that term either in a more loose or strict fashion, it is likely that the generators of the format are using the field according to their definition.

Requirements for data based on project type

As ITS infrastructure evolves and as additional systems or functionality are added to an organization's portfolio, the need for additional precision, accuracy, and concurrency increases. While data may exist, it is important to take into consideration the data's PACT given it's previous uses. Maintaining confidence in the system Most large transit organizations have had a rough progression in systems along the lines of the following.

A *Vehicle Block*, in common transit usage, is vehicle's schedule that begins and ends at a depot. GTFS provides optional block_id information, specifically for when "a passenger can transfer from one trip to the next just by staying in the vehicle." It is not uncommon that GTFS feeds use the common definition, rather than conform strictly to the GTFS specification.

Scheduling: Early scheduling systems took into account only terminals and timepoints, using a simple schematic network between them. Geography or intermediate stops were not taken into account.

Detailed Scheduling: Modern scheduling systems include geographic information for paths and stops and has the ability to generate passing times for each stop, which allows for detailed GTFS creation.

Computer Aided Dispatching: In order to dispatch effectively, locations of terminals and timepoints must be accurate. If a timepoint is on the wrong side of the intersection or block, the CAD system will produce incorrect results in that vicinity.

Static Trip Planning: Passengers need to know where to board and alight the bus.

Real-Time Passenger Information: The location of every stop must be within reason, and every revenue trip must be accounted for.

Reporting and Analysis: Reporting on the work that goes on behind-the-scenes requires the behind-the-scenes data (e.g. runs and blocks) to be correct, above and beyond the passenger information use case.

On Board Announcements: Announcements made too late because of incorrect stop locations or for stops that do not exist render the system useless.

Real-Time Trip Planning: Planning trips in real time, sometimes hours in advance, requires the greatest level of PACT from all of the above– systems must concur for real-time information in the future to be valid.

Changes: They are a'comin.

Given the complex nature of many transit systems and the differing requirements for PACT based on the type of project, the question is not if data will need changing, but when and how. When implementing a complex system depending touching party, it is often easier for all parties when the data producers and users agree on a policy on what data should be updated and at what moment. One framework that may be useful in quantifying the value of updates to data is to define impacts from service changes over three dimensions:

- (temporal) duration, the amount of time that the change will be in effect;
- (spatial) extent, the area over which the change affects; and
- (human) magnitude, the number of customers/passengers that are affected.

As with the proper level of PACT, there is no universal policy for data changes, and one should be developed on a project by project basis. The data owners should define the relevant impact classifications (short versus long duration, small versus large area, etc), and the proper policy will be specific to the organization's needs. One of many possible strategies is outlined below.

Temporal	Spatial	Human	Policy
Very Short	Small	Small	Do nothing.
Very Short	Large	Large	Create a “Service Alert.”
Long	Small	Small	Update data with next scheduled update.
Long	Large	Large	Update schedule data ASAP.