Generic Schema Descriptions for Comma-Separated Values Files of Environmental Data

Stephan Mäs, Daniel Henzen,
Lars Bernard, Matthias Müller
Chair of Geoinformatics
TU Dresden
Dresden, Germany
stephan.maes | daniel.henzen | lars.bernard |
matthias_mueller@tu-dresden.de

Simon Jirka
52°North GmbH
Martin-Luther-King-Weg 24
Münster, Germany
jirka@52north.org

Ivo Senner
Fraunhofer Institute for
Computer Graphics Research IGD
Fraunhoferstr. 5
Darmstadt, Germany
ivo.senner@igd.fraunhofer.de

Abstract

Comma-Separated Values (CSV) files are commonly used to publish data about environmental phenomena and environmental sensor measurements. Due to its simplicity, this format has many advantages. However, at the same time there is no official standard for CSV and no possibility to specify schematic constraints or other metadata. As a result, CSV files come in many variations and often with no metadata that would support interpretation or further processing, analysis and visualization. In this paper, we propose a framework for the specification of schema descriptions for CSV files as they are used in the environmental sciences. It allows to constrain the structure and content of a CSV file and also to specify relations between files, for example when they are published in one data package. The framework is extensible, also to other spatial data formats such as GeoTiff. The schema descriptions are encoded in JSON or XML to be published in the Web as a supplement to the data. It comes as a lightweight solution that provides metadata required to publish OGC compliant services from CSV files. It helps to overcome the heterogeneities of different data providers when exchanging environmental measurement data on the Web.
Keywords: tabular data, generic schema language, CSV, comma separated values, metadata.

1 Introduction

In the last decades, huge amounts of data have been published and made open and available to everyone. To a large portion this data is provided in a tabular form. For example, in 2014 over 90% of the open data released by the British Government at data.gov.uk is tabular data (Tennison 2014). This particularly counts for data about environmental phenomena and environmental sensor measurements. For this type of data, the prevailing file format is Comma-Separated Values (CSV).

CSV files are plain text files to store and exchange simple structured tabular data. Typically, each line in the table represents a separate data record. The fields of a record are separated by commas (despite the naming other characters, e.g. semicolon or tab, are also used) and all records must have the same structure, i.e. the same sequence of fields. The common file extension is ".csv". The CSV file format is open, well known and widely supported by text editors, spreadsheet programs, databases and programming languages. Due to its simple text-based structure, it can be easily produced or edited manually. As a sequence of lines, it can be incrementally processed with scripts or programs and supports streaming. Numerous Web applications such as CKAN[1] or Google fusion tables[2] support the generic interactive visualization of CSV data as graphs or maps.

However, despite its wide use there is no official and generic standard for CSV files. Only a recommendation of the Internet Engineering Task Force (IETF) is commonly used as a specification (Shafranovich 2005). The constraints defined by this recommendation are relatively weak. Therefore, many variations of CSV files are in use with different encodings, variable quoting of special characters, and variable line endings (Tennison 2014). Thus, most of the developed tools can only handle very limited variations of CSV files (Chaochaisit et al. 2016). A further problem is the lack of a standardized metadata or schema description for CSV files and relations among files (e.g. in JSON or XML). Contextual information of the data is not encoded in the CSV files and there is no possibility to specify the data types used in the different columns and value restrictions or the semantics of values (Chaochaisit et al. 2016, Arenas et al. 2016). When published on the Web, such descriptions are usually provided separately on related websites, but not in a formalized and machine-readable way. In the following, we propose a simple framework for the specification of schema descriptions for CSV files as they are commonly used in the environmental sciences. Typical examples we support with our solution are time series measurements, measurements of platforms that observe different phenomena (e.g. weather stations) or alert messages with a spatial reference.

2 Related Work

In the literature, numerous approaches to harmonize heterogeneous and inconsistent CSV data or to annotate CSV files or other tabular data with metadata and data schemata can be found. Many solutions focus the support for transformations to other formats such as RDF or JSON. Most of the approaches have limitations in terms of their support of arbitrary CSV and semantic relations or dependencies between fields. To the best of our knowledge, none of the solution specifies types for geometries, spatial coordinates and

[1] https://ckan.org/

[2] https://support.google.com/fusiontables/answer/2571232?hl=en

respective links to the spatial reference system, as well as units of measurements.

In 2015 and 2016 the W3C "CSV on the Web" (CSVW) Working Group[3] published a series of deliverables and recommendations on the generation of JSON and RDF from CSV files, best practices for CSV Files on the Web and metadata about tabular data. This solution is mainly supporting the CSV as it is specified by IETF (Shafranovich 2005). A more flexible framework has been proposed by Chaochaisit et al. (2016). They define a generic JSON-based CSV schema language for the transformation into RDF. This solution also allows to specify semantic relations among the fields in a CSV file.

Martens et al. (2015) define a rule based schema language for tabular data. The strengths of their solution are expressions to select and constrain parts of a table.

GeoCSV[4] is a geospatial extension of the CSV as specified by the IETF (Shafranovich 2005). It is used, for example, by the geospatial data translation library GDAL (Geospatial Data Abstraction Library). GeoCSV specifies geometry types such as point, line string and polygon. All geometries of a CSV file must refer to the same coordinate reference system (CRS) that is specified in a separate ".prj" file. Further, the data type restrictions for the fields in the CSV are stored in an additional file with a ".csvt" extension. This allows for simple data type restrictions but there are no formalisms for further schema descriptions, such as dependencies among data fields in the CSV.

3 Generic Schema Descriptors for CSV-Files

The application background of this work is a Web platform for the development of urban early warning systems. Our main focus is on local heavy rain, flooding and cascading events affecting urban water and sewage infrastructures, but also traffic infrastructures. The warning information is derived from sensors, chemical lab measurements, crowdsourced data, geosimulations, as well as administrative and historical data. Most of the data used is provided as CSV files. These files are heterogeneous with no or weak descriptions, that are usually provided on separate Web pages. For the early warning system, the data shall be automatically harvested into a CKAN based data portal and then published with OGC compliant services, such as the OGC Sensor Observation Service (SOS). These services are then used to build applications consuming the data (e.g. data viewers, data analysis and processing tools, early warning systems). The overall process shall be as far as possible automatic and minimize the manual efforts to provide the services. This requires metadata about the CSV files (e.g. structure and contents) that cannot be provided in the files themselves. The schema descriptors introduced here provide the required information and act thereby as a mediator between the systems.

3.1 Requirements

A very detailed analysis of use cases and requirements of the interoperable use of CSV files on the Web has been made by the W3C (Tandy et al. 2016). Many of them are conform to the requirements of our use case. General requirements for environmental applications are the definition of datatypes for thematic attributes and geometries, spatial coordinates and links to the respective reference system, time and time formats, as well as units of measurement.

A typical example for data relevant for our application are, for instance, recent weather observations containing measurements of temperature, air pressure, sunshine duration, cloudiness, precipitation and warnings for extreme events. Such data is provided as a collection of files containing measurements covering the whole of Germany and is typically updated on a daily basis. Thus, for efficiency reasons the metadata should be external to the CSV files. It should also allow for a flexible reaction in case of data schema changes.

Each record in a CSV file should have a primary key. In some cases, the contents of one CSV file relate to the contents of another file, for example one file contains a column for the abbreviations of administrative units and the corresponding geometries are stored in a separate file. Such association of code values with external definitions requires foreign keys to cross-reference between CSV files. Further, all primary keys should be unique, if not globally then at least in the domain of the considered csv-files.

A further requirement is the specification of the relations between the columns of a table. For example, if measurements of a weather station are stored together in one CSV file with one column containing the measurements values and another column the corresponding phenomena that have been observed (e.g. temperature or air pressure). Without a reference between the columns, an interpretation of the measurements is not possible. Similarly, in table 1 the values in the severity column refer to the entries in the event column.

The tabular data is often published as a package of files (e.g. as a zip archive) containing data tables with the same or differing data schemata. It should be possible to describe the interrelationships between the contents of the individual tables and to provide a schematic description for the whole data package.

Table 1: Example CSV of alert messages of the German Meteorological Office

SENT	MSGTYPE	SEVERITY	EVENT	AREADESC	longitude	latitude
2017-09-20T02:45:00Z	Alert	Minor	FOG	Kreis Waldeck-Frankenberg	8,8387236747	51,1969596534
2017-09-20T02:45:00Z	Alert	Moderate	RAIN	Kreis Berchtes-gadener Land	12,8844948308	47,7259945858
2017-09-20T02:45:00Z	Alert	Minor	WIND	Nordfriesische Küste	8,5739733278	54,6558522261

Data source: http://www.wettergefahren.de

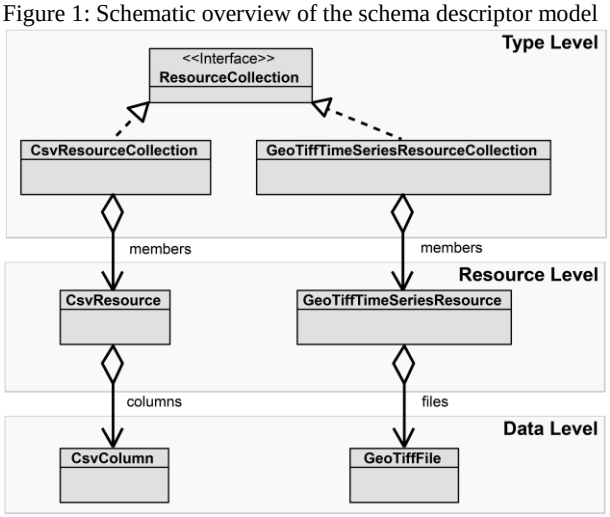
[3] https://www.w3.org/2013/csvw/wiki/Main_Page

[4] https://giswiki.hsr.ch/GeoCSV

3.2 Data model

A schematic overview of the UML model of the schema descriptor is shown in Figure 1. It is structured into three levels: type level, resource level and data level. Main reason for this structure is that our use case requires the description of data packages that possibly contain data files with different schema descriptions. Such a package is named *ResourceCollection* in the model. The schema descriptors are used to describe CSV files, but also other formats such as GeoTiff or NetCDF. In this paper we focus only on the schema descriptors for CSV files.

Figure 1: Schematic overview of the schema descriptor model

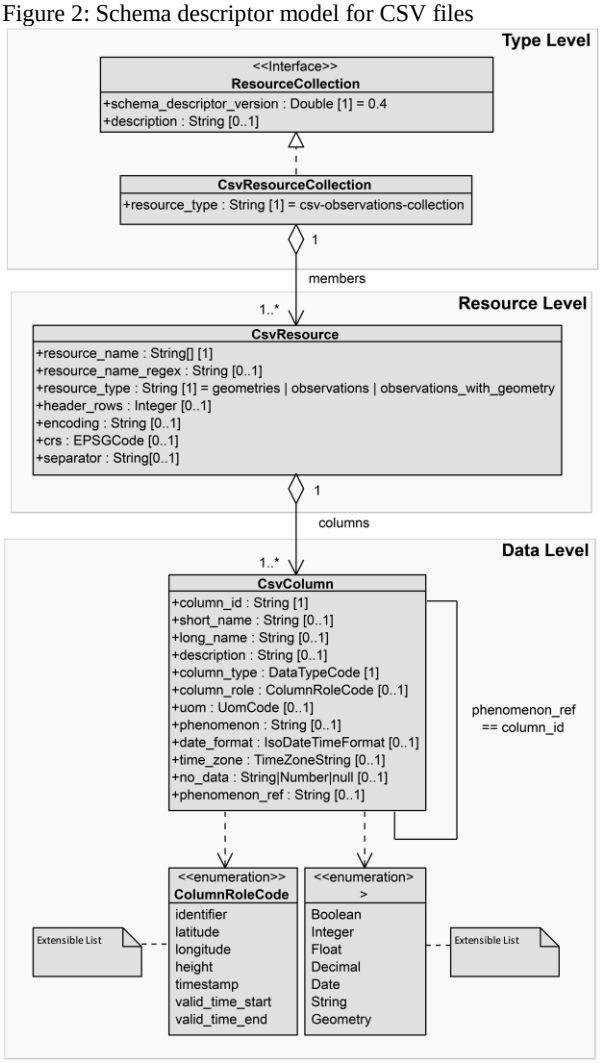


The resource level provides encoding information, including information about the particular encoding, separator or regular expressions in the file names. For CSV schema descriptors each resource has a number of columns that are described at the data level.

Figure 2 shows the schema descriptors model for CSV files in more detail. The subclasses of the *ResourceCollection* class differentiate the schema descriptors of different resource types (e.g. CSV or GeoTiff). They combine the descriptors of a resource collection (i.e. data package). A *CsvResource* provides the actual schema description of a CSV file. It contains information about the number of header rows (containing the column names and possibly other metadata), the character encoding and the separator of the tabular fields. The *resource_name* attribute stores references to all CSV files that are annotated by the schema descriptor. This reference between the CSV file and its metadata could also be encoded in the file name (described by the *resource_name_regex* attribute)

A peculiarity of the environmental data we are targeting is that each of the data entries in the CSV has a spatial reference, for example a coordinate tuple or a polygon. To avoid redundancies and to save storage space these geometries are often stored in a separate file and not repeated in each data entry. This is described with the *resource_type* attribute, which has the value *observations*, if the geometries are stored in a separate CSV file. In this case the *CsvResourceCollection* contains the resource description of the geometry CSV and the observation CSV must contain a foreign key to the corresponding geometry. We restrict the CSV files to contain a fixed number of columns and spatial coordinates of only one coordinate reference system (CRS), as this is sufficient for our use case.

Figure 2: Schema descriptor model for CSV files



The columns are described by the *CsvColumn* class. Each column is defined by its identifier, data type, name and description. For columns that contain a set of well-defined values, enumerations can be specified as a data type. The *column_role* attribute specifies typical roles of values in CSV files for environmental applications. The list of roles can also be extended, if further roles need to be distinguished. If column values have a unit the *uom* attribute contains the corresponding UCUM code (Unified Code for Units of Measure). To describe the observed phenomenon, like for instance temperature or pressure, we use "kind of quantity" terms defined by the UCUM. A common vocabulary that specifies observable phenomena could also be used here (like in (Kokkinaki et al. 2016)). To specify relations between columns we use the *phenomenon_ref* attribute. The columns of the referenced by their mandatory *column_id*.

3.3 Example

The schema descriptors are encoded as JSON or XML. Figure 3 shows a simplified JSON code snippet for the weather alert messages shown in Table 1. In this example, the *CsvResourceCollection* contains the schema description of only one *CsvResource*. The *resource_name* attribute provides explicit links to the instantiating CSV files. Since each of the alert messages contains point coordinates as spatial reference the *resource_type* attribute is set to *observations_with_geometry*. The role codes are used to mark the time stamp and the coordinates of the alert messages. An example for the specification of an enumeration as data type is the event column. The enumeration contains the list of possible weather events that cause the alerts (fog, wind, etc.). The severity column is linked to the event column to specify the semantic relation. The *column_id* is used here as a reference to show that the severity rates the phenomenon contained in the event field.

Figure 3: JSON example for weather alert messages

```
{
  "resource_type": "csv-observations-collection",
  "schema_descriptor_version": 0.4,
  "members": [{
    "resource_name": ["//file1", "//file2", "..."],
    "resource_type": "observations_with_geometry",
    "columns": [{
      "column_id": "SENT",
      "column_type": "Date",
      "column_role": "timestamp"
    }, {
      "column_id": "MSGTYPE",
      "column_type": "String"
    }, {
      "column_id": "SEVERITY",
      "column_type": ["Minor", "Moderate", "..."],
      "phenomenon_ref": "EVENT"
    }, {
      "column_id": "EVENT",
      "column_type": ["FOG", "WIND", "RAIN", "..."],
      "description": "weather event"
    }, {
      "column_id": "AREADESC",
      "column_type": "String"
    }, {
      "column_id": "longitude",
      "column_type": "Float",
      "column_role": "longitude",
      "uom": "degree",
      "description": "decimal degrees"
    }, {
      "column_id": "latitude",
      "column_type": "Float",
      "column_role": "latitude",
      "uom": "degree",
      "description": "decimal_degrees"
    }]
  }]
}
```

4 Discussion

The introduced schema descriptors provide the required metadata to publish OGC compliant services from CSV files and help to overcome the heterogeneities of different data providers. To verify the described approach for annotating the structure of CSV files, we have implemented an open source tool to harvesting tabular observation data files stored on a CKAN server. This implementation has shown the robust applicability of the CSV schema descriptors in an early warning system using a broad range of different data inputs. The schema descriptors are applied to flexibly use precipitation data of different data providers as an input for an OGC Web Processing Service (WPS) that calculates the runoff in a sewage system, for example. When designing the schema descriptors, our focus was on the provision of metadata that supports interpretation and further processing, analysis and visualization. A validation of files against the schema description is also possible, but this was not yet our objective. Correspondingly, a detailed analysis of the expressiveness of our formalization and test validations with CSV files has not yet been conducted.

The schema descriptors provide a generic framework for CSV metadata acquisition, but also other data formats such as GeoTiff are supported. The encoding in JSON eases the exchange over the Web and the integration with other – also non-GI – applications. Further, the approach is system and software independent and the schemas can be easily extended or adjusted if the structure of the data changes. With the provided schema information CSV files can be transformed to other formats, like for instance GML.

In order to stay simple, we accepted some limitations in our formalizations. We restrict the schema descriptions to contain only one data table definition for each CSV file. That means, all rows in a CSV file must have the same number of columns and the columns can only contain one data type. Linking to particular data fields is also not possible. Other solutions, like for example (Chaochaisit 2016) and (Martens et al. 2015), are less restrictive in that regard. However, such more generic solutions require a higher implementation effort and may hinder adaptation. To overcome semantic heterogeneities we will elaborate on the explicit linking to external vocabularies or ontologies, e.g. for the phenomena and units of measure or to external code lists for attribute values. To improve the representation of semantics in the schema we will investigate classifications of the relations between the columns of a table.

References

Arenas, M.; Maturana, F.; Riveros, C. and Vrgoc D. (2016): A framework for annotating CSV-like data. In Proceedings of VLDB Endowment, vol. 9, issue 11 (July 2016), 876-887

Chaochaisit, W.; Sakamura, K.; Koshizuka, N. and Bessho, M. (2016): CSV-X: A Linked Data Enabled Schema Language, Model, and Processing Engine for Non-Uniform CSV. IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing

(CPSCom) and IEEE Smart Data (SmartData), Chengdu, 2016, pp. 795-804.

Kokkinaki A.; Darroch, L.; Buck, J. and Jirka, S. (2016): Semantically Enhancing SensorML with Controlled Vocabularies in the Marine Domain. In: GSW 2016 - Geospatial Sensor Webs Conference 2016, Münster.

Martens, W.; Neven, F. and Vansummeren, S. (2015): SCULPT: A Schema Language for Tabular Data on the Web. In Proceedings of the 24th International Conference on World Wide Web (WWW '15). 702-720.

Shafranovich, Y. (2005): *Common format and MIME type for comma-separated values (CSV) files*. RFC 4180 of the IETF, [Online] Available from: https://www.ietf.org/rfc/rfc4180.txt [Accessed 2nd January 2018].

Tandy, J.; Ceolin, D. and Stephan, E. (2016): CSV on the Web: Use Cases and Requirements. W3C WG Note, [Online] Available from: https://www.w3.org/TR/2016/NOTE-csvw-ucr-20160225. [Accessed 5th January 2018].

Tennison, J. (2014): *2014: The Year of CSV*. [Online]. Available from: https://theodi.org/blog/2014-the-year-of-csv. [Accessed 2nd January 2018].