Section x

AEGIS vocabulary to support Big Data

x.1 THE AEGIS PLATFORM

This section is based on the following assumptions on the AEGIS platform:

- The AEGIS platform is meant to host *many* datasets (or references to them), together with their metadata; hence certain search functionalities are important.
- A relevant part of the datasets are *Big Data;* hence the typical properties and procedures of Big Data are to be considered.

The propositions made in this section will enhance the expressiveness of the metadata, in particular to support the automated finding of potential dataset *combinations,* a relevant topic in Big Data.

In order to outline the topic of this section, we conceptually split the metadata into three levels:

1. There is meta-information about datasets in general, such as *owner, origin, access rights* and similar. This kind of meta-information can probably be described by existing vocabularies such as DCAT-AP; it is not the topic of this section.
2. → There are statements that can be made specifically about the logical structure of Big Data datasets, and that would probably be useful in the AEGIS platform. The recognition and tentative formalisation of these statement as metadata, and their usage in AEGIS, *is* the topic of this section.
3. There is technical and syntactical meta-information about data sets, e.g., the column separation character and the number of header lines in the case of CSV data. This kind of meta-information is not addressed in this section.

Level 2 and probably also level 3 will result in dedicated new vocabularies, which will merge into the "aegis:" vocabulary being developed.

x.2 BIG DATA AND BIG DATA EVALUATIONS

At the time of writing, not much can yet be said about what kind of applications the AEGIS platform will be used for. However, regardless of this, something can be said about Big Data and Big Data evaluations in general. Many Big Data datasets belong to some of the following categories:

- Text collections in natural language
- hierarchically structured textual data (e.g., XML or JSON texts)
- databases, often of the NOSQL variety, and column stores
- vectors
- graphs, RDF data
- collections of perceptive data, such as images, audio and video samples
- tabulated functions (with various domains and ranges, one- or multi-dimensional).

  This category is meant to comprise tabular data that conceptually resemble functions in the mathematical sense. Examples are time series, geo data, spatio-temporal data etc. They possess one or more key columns, in which every value or tuple is unique, and one or more value columns. (When being plotted, the keys are typically plotted along the X-axis, and the values along the Y-axis.)

These categories are not complete, not outlined sharply and not fully disjoint, what however is not an issue here. We observe that these data structures all are “container data” in some general sense; they possess:

- *structural properties* (they are vectors, tables, graphs, hierarchies, etc.)
- *content-related properties* (they have *element types,* i.e., the types of the vector entries, of the table column entries, of the graph vertices and edges, and so on.).

In some cases, these properties are inherently machine understandable, e.g., in tabular data. In others, they are hidden and difficult to access, as in natural language texts. – Just like most data structures belong to a small set of concepts, something similar can be said about the content types. They reflect something of the real world; they may be

- words or URIs to denominate “things”, which may also be typed or qualified, e.g., as geographic entities, business sectors, etc.
- database keys and foreign keys (surrogate keys; usually integers, but could be anything)
- numbers to count or measure things, and thus they may also bear units
- time points
- spatial or geographic coordinates
- subsymbolic data (texts, images, audio/video samples, whose meanings have yet to be carved out), or pointers to such data.

Also this list is not meant to be exhaustive and will grow in practice, and although it probably will never be complete, it seems that a brief list will be capable of capturing the majority of cases.

As to Big Data *evaluations,* there is a similar situation: Although we don’t know exactly what they are going to be, some typical characteristics of them can be stated nevertheless:

1. Like everywhere else, the selection of Big Data evaluation methods depends first of all on the intended result, and then on the data structures at hand. But in the Big Data context more than elsewhere, *exploratory* analyses are customary, i.e., analyses without an specific aim, directed by what is there. Some data call for numerical or statistical methods, while others call for discrete algorithms; so the choice of an evaluation method is directed by the data at hand.
2. Big Data evaluations may well work on a single dataset; however, it is also very common for them to *combine* datasets, in order to get more global insights. But not every combination of datasets is possible and meaningful. Typically the participating datasets need to exhibit compatible features along which some kind of combination can be carried out. We give the following examples for this:
   - Database tables can be *joined,* provided the columns used for the join have compatible types.
   - Multiple times series can possibly be *correlated* (in the statistical sense of the word). The same holds for multiple geo data collections, spatio-temporal data, etc. The key issue is that they need to have compatible key types and compatible value types.

   The precise meaning of “compatible” in such cases depends on the situation at hand. And even when multiple datasets can be combined and evaluated jointly, it is still a different issue whether this combination gives meaningful results.

Taking this together, *exploratory* analyses on *combinations* of datasets – both typical for Big Data – mean that combinable datasets have yet to be found. To this end, it is advisable that the metadata express some sort of “signatures” of the datasets, whose comparisons tell whether two datasets have something “compatible” in common. These signatures will refer to the dataset properties described in this subsection.

x.3 THE AEGIS PLATFORM AND THE PURPOSE OF METADATA

Within the AEGIS platform, the purposes of the metadata are (cf. subsection x.1):

- **Search**: The metadata allow for quickly finding datasets that serve for a particular purpose, without having to inspect the datasets themselves.
  This is relevant because the AEGIS platform is envisioned to hold a large number of datasets in a variety of formats, which makes searching them directly prohibitively slow and complex.
- **Access**: Once it has been settled which datasets are to be subjected to what evaluations, the metadata support the evaluations by indicating how the datasets can and should be accessed.

Now recall that Big Data evaluations may well involve several datasets, but that not every pairing of datasets makes sense; recall the compatibility issues mentioned above. The activity of finding matching datasets that can undergo a joint evaluation is particularly dependent on the support from suitable metadata, simply because there are many more pairs of datasets than there are datasets. So there is a third purpose of metadata in AEGIS:

- **Matching**: The metadata allow for checking whether two datasets have something in common so that they can be used jointly in an evaluation. (This refers to the *formal* part of the judgement, the first barrier.)

x.4 CONSEQUENCES

In subsection x.2 we have seen what can be said about Big Data datasets and operations on them, in x.3 we have seen what is important for the AEGIS platform. All in all, we have laid the foundation for the following conclusion for the AEGIS platform: The metadata should formalise the structural and content-related properties of the data, as they have been outlined above. As a result, the – presumable large – collection of datasets can searched, matching datasets can be found, and the data within the datasets can be accessed.

Formalising information as metadata amounts to devising an RDF vocabulary. In AEGIS we proceed as follows:

- For the structural properties, we simply devise RDF properties and classes for the most significant structural concepts, as they have been mentioned in the preceding subsection.
- For the content-related properties, the best thing that can be done at the time being is to devise properties and classes for all significant semantic types that spring to mind from the real-world usages.

Both of these parts may of course be extended when needed. The vocabulary to be devised for AEGIS should make these things expressible in a concise manner; it should be understandable in minutes, not days, it should be presentable in a few slides only, and its description should fit into a few pages, not 100.

x.5 FORMALISATION OF TABULATED FUNCTIONS

To begin with the formalisation of dataset properties in terms of RDF classes and properties, we pick just one of the structural properties from Section x.2, viz. that of tabulated functions.

The transition from the mathmatical parlance to data structures is as follows: Functions have a domain and a range, both of which are sets, which in turn may be Cartesian product sets of multiple factors. When a functions is resembled by tabular data, the (factors of the) domain is/are resembled by the key column(s), and the (factors of the) range is/are resembled by the value column(s). The factor sets are resembled by the column types. And finally, for the key column(s) we have the restriction that every value/tuple therein must be unique (in database parlance, these columns constitute the primary key).

In a simple example, we show how a metadata formalisation of his might look like, in a sloppy notation in the style of N3 or Turtle, and using the yet-to-be-defined vocabulary with prefix “aegis:”:

```
<my-dataset-uri>
        aegis:hasKeyFactor
                [rdf:type aegis:Factor;
                        aegis:columnNumber 1;
                        aegis:columnstype aegis:TimePoint
                ];
        aegis:hasValueFactor
                [rdf:type aegis:Factor;
                        aegis:columnNumber 2;
                        aegis:columnstype aegis:MeasureOrCount,
                        aegis:MeasureOrCountedUnit "degree Celsius"
                ].
```

First, we define two properties that link datasets to their columns, i.e., to their key factors and to their value factors.

Name	rdfs:domain	rdfs:range
aegis:hasKeyFactor	aegis:DataSet	aegis:Factor
aegis:hasValueFactor	aegis:DataSet	aegis:Factor

Each factor (column in the table) bears the type aegis:Factor, and is further described by the following properties:

Name	rdfs:domain	rdfs:range
aegis:columnNumber	aegis:Factor	xsd:integer
aegis:columnHeader (optional)	aegis:Factor	xsd:string
aegis:columnType	aegis:Factor	rdfs:Class
aegis:MeasuredOrCountedUnit (optional)	aegis:Factor	rdfs:Resource

As targets of the *aegis:columnType* property, we tentatively propose the following classes:

Name	Meaning: Entries of described column contain ...
aegis:MeasureOrCount	numbers that count or measure something
aegis:TimePoint	points in absolute time including date; any precision
aegis:GeoName	proper names of geographic entities
aegis:GeoCoordinates	longitude/latitude coordinates
aegis:FreeText	text in any natural language
aegis:Image	image in binary format
aegis:Video	video in binary format
aegis:Audio	audio in binary format
aegis:DatabaseKey	number or other literal, meaningful only as surrogate key of a determined table

Note: This formalisation refers to the *logical* view; i.e., text/image/video/audio may also mean *pointer* to the respective kind of data stored elsewhere.

Now finally, if every column of every table ist described in this way, then, for instance, a visualiser sees what quantities are going to be plotted, and colums on which a table join or a correlation can be based can be recognised automatically. — In Section x.3 we stated that for the latter purpose, column types would have to be "compatible" in some sense. That notion can be used liberally; for example, the geo *name* entries (e.g., cities) from one column and geo *coordinate* entries from another column may be considered as matching, yielding a positive result if and only if the geo coordinates point into the named entity.

This has been the design only of a single aspect of the AEGIS vocabulary for Big Data, in order to outline the procedure.