KGE - Knowledge Graph Engineering

Knowledge Graph Engineering

Building Reusable Knowledge Graphs

Fausto Giunchiglia

Contents

Build KGs - What does it mean?

- Knowledge Graph Engineering (KGE) defines how to build a KG, starting from the user's Purpose (usually expressed informally as a natural language sentence).
- KGE is a complete approach for the creation of KGs defining:
  - *Activities*: which are the activities required to build KGs, without running into high production costs.
  - *Rules*: the rules to be followed to build KGs, like the order of execution of the activities, as well as the user's roles involved in those (skills required).
  - *Best practices*: which are the steps to consider in order to built **quality** and **reusable** KGs, like **sharability** of the resources produced.

Build KGs - What does it mean?

- The set of activities, rules and best practices, included in KGE, aim to find a solution for the different problems to be solved when a KG has to be built.
  - **Purpose formalization**: the analysis of the Purpose expressed by the user, with the objective to extract a more detailed set of requirements.
  - **Resources collection**: retrieve the resources (knowledge and data) required to satisfy the Purpose.
  - **Resource management**: deal with the heterogeneity of the resources collected.
  - **Resource integration/composition**: integrate, within a single information structure (KG) the required resources.
  - **Resource sharing**: share the outcomes to support future KG building (new) and evolution (existing).

Which is the level of effort required to solve for each problem ?

Contents

Purpose formalization

- The Purpose, as expressed initially by the user, can be unclear, incomplete, hiding important details to be considered.
- Let's consider the previous example Purpose:

"The user want build a KG able to support the access to the health facilities in Trento (Italy) and all the medical cares that they can offer to the citizens."

- What does it means "access" ?
- Which are "all" the medical cares considered ?
- ...

Purpose formalization

- A formalized version of the Purpose describes as much as possible the details specifying the the information that has to be included in (and accessible through) the final KG.
- This KGE problem che be properly solved **only** with the support of a **domain expert**, where, the domain is the the information field in which the Purpose lives.
- Even if sometimes a specific assistance is required from external domain experts, usually the principal expert about the Purpose is the user.

Contents

Resource Collection

- The collection of the resources required to build a KG, is **driven on top by the user Purpose**.
- Resource collection always considers knowledge resources and data resources (see KG definition).
- KGE considers the collection of resources which can bring the following two results:
  - Increase the number of **entities/entity types**
  - Increase the number of **entity attributes/entity type properties**
- There are two main sub-problems to be addressed regarding resource collection:
  - Resources identification
  - Resources retrieval

Resource Collection - Identification

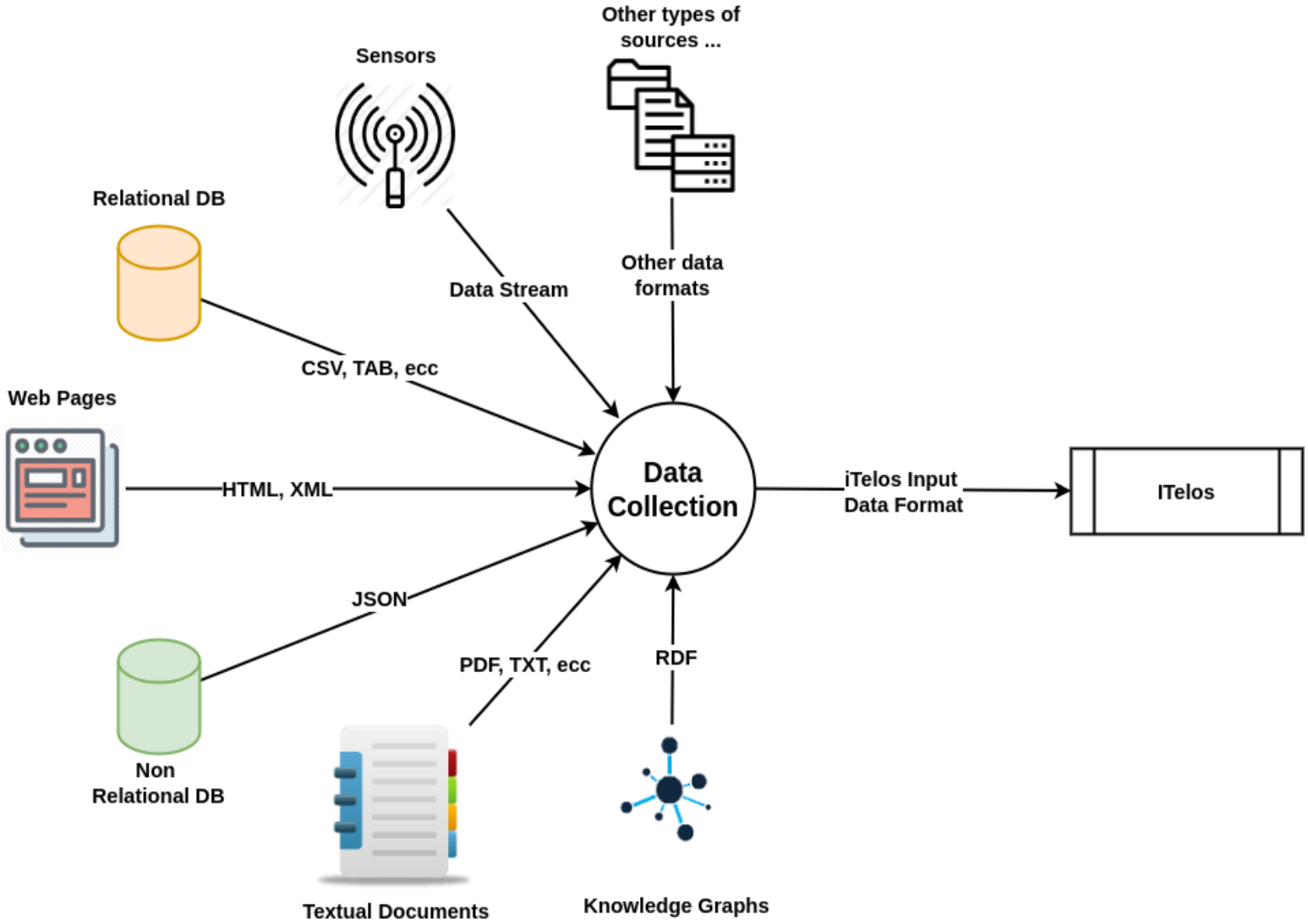
- The identification of the right resources is a crucial problem to address. More in details, it consists in the identification of the **sources** which can provide the information required.
- Several sources are available for the same domain of interest, Purpose, even for the same information we are trying to collect.

  How to choose between the multiple data sources ?

- **Metadata** (like "author" and "provenance") are used to provide more information over the data within a certain sources.
- Nevertheless, metadata are not always present and updated to be used in such a decision.

Resource Collection - Retrieval

- Moreover, the data sources have an high level of heterogeneity



Data sources types

We can divide the possible data sources in the following categories, based on the kind of data they provide:

- **Structured**: CSV, TAB, JSON, Spreadsheet, XML, and others.
- **Semi-structured**: web pages (HTML), data obtained by web scraping.
- **Unstructured**: textual document (PDF, txt).
- **Media**: images, videos.
- **Streams**: continuous data obtained from sensors.

How to address this kind of heterogeneity ?

Contents

Resource Management

- The management of the resources collected consists in dealing with the heterogeneity presents into the single resources (data and knowledge).
- To this end, the following activities have to be considered:
  - **Resource filtering**: It is not always the case in which the sources selected provide all and only what you are looking for. Useless resources have to be cut out to reduce the overall process cost.
  - **Resource cleaning**: Most of the time the resources collected include noise, represented by dirty or incomplete information (missing values, attributes, entities, ...).
  - **Resource formatting**: In particular, the resources collected are affected by different kind of *syntactic heterogeneity*.

Syntactic Heterogeneity

- What is the data syntactic heterogeneity ?
- Such kind of heterogeneity appears at data value level, when data use different values to represent the same kind of information, thus causing misalignment between different resources.
- There are in general three types of misalignment to consider addressing the syntactic data heterogeneity:
  - Data types misalignment
  - Data value format misalignment
  - Data value language misalignment

Data types misalignment

The data types misalignment appears when the same information is represented using different data types.

Example: consider two different dataset, A and B, containing data about students in Trento.

Entity in dataset A:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- Telephone: 3328877451

Entity in dataset B:

- Name: "Simone"
- Surname: "Bocca"
- Age: "29"
- Telephone: "3328877451"

Data types misalignment

The data types misalignment appears when the same information is represented using different data types.

Example: consider two different dataset, A and B, containing data about students in Trento.

Entity in dataset A:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- Telephone: 3328877451

Entity in dataset B:

- Name: "Simone"
- Surname: "Bocca"
- Age: "29"
- Telephone: "3328877451"

Age and *Telephone* values, in dataset A are represented as Integer and Long values. While in dataset B the same information is represented using Strings.

Data value format misalignment

The data value format misalignment appears when different formats of the same data type are adopted for same information in different datasets.

Example: consider the same two dataset, A and B.

Entity in dataset A:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- BirthDate: "1992-08-13 15:35:03"

Entity in dataset B:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- BirthDate: "713720103"

Data value format misalignment

The data value format misalignment appears when different formats of the same data type are adopted for same information in different datasets.

Example: consider the same two dataset, A and B.

Entity in dataset A:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- BirthDate: "1992-08-13 15:35:03"

Entity in dataset B:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- BirthDate: "713720103"

BirthDate value, in dataset A is represented as a String in ISO date format. While in dataset B the same information is represented using a unix timestamp String.

Data value language misalignment

The data value language misalignment appears when different natural languages are adopted for same information in different datasets.

Example: consider the same two dataset, A and B.

Entity in dataset A:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- Student-type: "Doctoral student"

Entity in dataset B:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- Student-type: "Studente di dottorato"

Data value language misalignment

The data value language misalignment appears when different natural languages are adopted for same information in different datasets.

Example: consider the same two dataset, A and B.

Entity in dataset A:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- Student-type: "Doctoral student"

Entity in dataset B:

- Name: "Simone"
- Surname: "Bocca"
- Age: 29
- Student-type: "Studente di dottorato"

Student-type value, in dataset A is represented as a String in English. While in dataset B the same information is represented using an Italian String.

Contents

Resource Integration/composition

- Integration of resources include a crucial problem to be addressed, the **semantic heterogeneity**.
- Such kind of heterogeneity indicates the presence of multiple representation of the same real world entity, within the resources collected.
- The semantic heterogeneity is a:

  "consequence of the more general phenomenon of the diversity of the world and of the world descriptions." (Giunchiglia, Fumagalli 2020)

Semantic Heterogeneity - Example

Consider two different dataset, A and B, both including entities representing buses.

Bus in dataset A:

- Vehicle-ID: 4321
- Manufacturer: "Iveco"
- Engine-type: "Electric engine"
- Fuel-type: "Electricity"

Bus in dataset B:

- Vehicle-ID: 4321
- Line-number: "13-A"
- Seats: 30
- Daily-travel-time: 650

In dataset A, a bus is seen from the point of view of the manufacturer, while in dataset B the same bus is seen from the point of view of the public transportation company.

The same real word entity is represented using different properties due to the **different function associated to the entity** within the two datasets (i.e., in dataset A the bus is a motor vehicle, while in dataset B is a transportation vehicle).

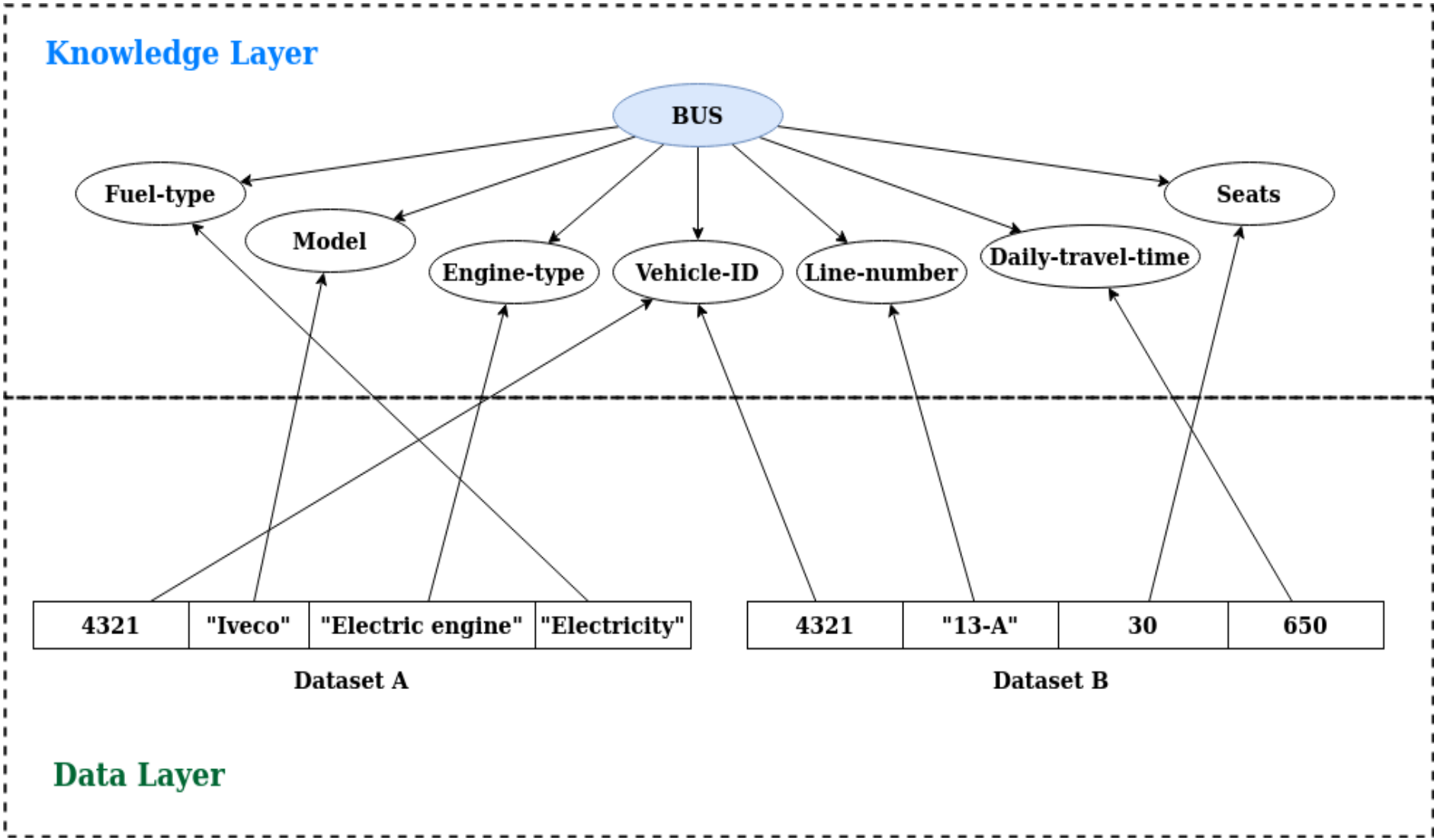
Resource Integration/composition

- Different approaches have been studied to solve this problem, involving common schema/ontologies (OBDA techniques, see Knowledge Graph lecture).
- Such solutions require a common schema/ontology (Knowledge layer of the final KG) that has to be properly defined based on the Purpose.
- Nevertheless, such solutions have to be then concretely applied to the resources collected (datasets), through a dedicated activity called **Entity Alignment**.

Entity alignment

- The entity alignment activity is responsible for the **merge** of the two KG's layers, knowledge and data.
- In this activity a Data Scientist (DS), using a specific tool, maps the datasets values (representing entities and entity properties) on the etypes and etype properties defined in the common schema adopted for the KG.
- Knowing the data meaning, the DS can map the datasets values on the rights common ontology's etypes and properties, covering in this way the semantic misalignment between the different datasets.

Mapping operations - Example



Mapping operations - Notes

- **Entity identification**: The datasets can include values used to identify the single entities, like *4321* in the previous example.
- Such values are used as *identifiers* to distinguish between the entities across different datasets.
- But the identifiers are not always present (and/or specified) within the datasets ...
- The lack of identifiers, can lead to add in the KG, separate entities which instead refer to the same real world entity.
- This activity used to solve this problem is defined as **Entity Matching**.

Entity matching

- *How to match two separate entities that should be a single one ?*

- A solution to identify the entities, can be considered at knowledge level by defining, for each etype in the KG's common ontology, an *Identifying Set*.

Identifying Set: a set of etype's properties which, through their values, identify uniquely an entity (defined for such an etype) within the whole set of entity considered.

Entity matching - Examples

Consider the same two datasets as before where the identifiers are not available anymore.

Bus in dataset A:

- Production-year: 2007
- Manufacturer: "Iveco"
- Model: "AX-123"
- Engine-type: "Electric engine"
- Fuel-type: "Electricity"

Bus in dataset B:

- Production-year: 2007
- Line-number: "13-A"
- Seats: 30
- Daily-travel-time: 650
- Model: "AX-123"

The Identifying Set (IS) defined as follow:

$$IS_{Bus} = \textit{Production-year, Model}$$

allows the matching between the two *Bus* entities into a single one.

Entity matching - Merge conflicts

Once the entity matching has been identified, the attributes of the entities to be merged, are checked in order to discover possible conflicts while merging the two entities.

For example, considering the two Bus entities as follows:

Bus in dataset A:

- Production-year: 2007
- Manufacturer: "Iveco"
- Model: "AX-123"
- Engine-type: "Electric engine"
- Fuel-type: "Electricity"
- Color: "red"

Bus in dataset B:

- Production-year: 2007
- Line-number: "13-A"
- Seats: 30
- Daily-travel-time: 650
- Model: "AX-123"
- Color: "green"

Q: Which value of the property *Color* should appear after merging the entities ?

Entity matching - Merge conflicts (2)

Such kind of conflicts between entities that have to be merged, can be caused by errors within the dataset or different representation of the same entity (due to different context in which the datasets have been created).

> **A**: Specific kind of metadata associated to the datasets and/or directly to the entities within the datasets, can help to decides a solution to solve conflicts like the one presented in the previous slide. One of them is the **Provenance**.

The Provenance is a metadata indicating the origin of the dataset/entity/value. Thanks to this information the data scientist can select which is the most appropriate values to associate to the etype property that produce the conflict.

Moreover the Provenance improve the level of reliability of the resources considered.

Contents

Resource Reuse

- A complete KGE process requires a huge effort over the resources, for collection, management and integration.
- Such effort, is responsible of the costs involved in building KGs.

How to reduce this cost ?

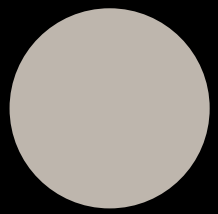
- **Reusing** as much as possible already quality resources.

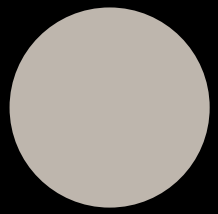
Resource Sharing

- Quality resources are those in which:
  - Noise and incompleteness is already handled.
  - Standard data types and formats are adopted.
  - Metadata are used to describe provenance and data meanings.
  - Identifiers are defined.
  - Data already have its own schema/ontology associated, thus describing formally the information carried.

A KGE process able to produce such quality resources has the "duty" to share them, in order to allow future usages, and thus reducing the effort in KG building, for the whole community.

KGE - Knowledge Graph Engineering





Fausto Giunchiglia

Knowledge Graph Engineering

Building Reusable Knowledge Graphs