Linked Data and Semantic Web

1 Introduction

1.1 Semantic Web

Create an infrastructure that can be navigated in a coherent way. Resources are interconnected by links. World Wide Web human processable information is made of linked resources. It doesn't matter how they are linked. A web page is a rendering of an html document. A machine doesn't interpret what a link is. They are all treated the same. Machines were limited in navigating information, because they can't tell the difference between them.

The semantic web has interconnected meaningful resources. Semantic web is an extension of the current web, where information and services are well defined, and can be shared and used by machines and humans.

Semantic web links make the relation explicit, they give them meaning. Resources can be interlinked regardless of their location (dynamic linking). There are multiple data sources.

1.2 Linked Data

How data is represented in the semantic web. LD is a set of best practices for publishing and connecting structured data on the web using standard formats and interfaces. It supports easy combination of multiple linked data sources.

Everything on the web right now is at least 1 star linked data.
⋆ Has to be linked/available on the web.
⋆⋆ Has a machine readable structure
⋆⋆⋆ Machine readable but in a non-proprietary format
⋆⋆⋆⋆ W3C standard approved (systematic approach)
⋆⋆⋆⋆⋆ Is linked to other data. **NOT** that it's open access.

1.3 RDF

Data model for linked data.
Resource pages, concepts, ideas (everything that can have an URI)
Description attributes, features and **relations** of the resources
Framework model, languages and syntaxes for these descriptions

RDF is a standard format for data exchange on the web.

XML is a tree model, RDF is a triple model. Formed by a **subject**, a **predicate** (relation), and an **object**[1]. Triples can be combined to form complicated knowledge graphs.

Linked Open Data (LOD) is LD which is released under an open license.

1.4 How to publish data as LD

LD principles:
- Use URIs as names for things;
- Use HTTP URIs so that people can look up those names;
- When someone looks up an URI, provide useful information using the standards (RDF, SPARQL)
- Include links to other URIs, so people can discover more things (5⋆ rule).

Steps:
1. Create our own URIs
2. Look at knowledge graphs/ontology repositories to reuse existing vocabularies

[1]The object can be the subject in another or multiple triples.

3. Convert the concepts from natural language to another term in the coding language (abstract from the real item). Machine just need the encoding/identifiers of concepts.
4. In the definitions, put links to other sources.

1.5 RDFS

It's **not** RDF-schema. It extends RDF with the possibility to define classes and associated properties. It allows RDF applications to agree on a common data description vocabulary. The most used RDF syntax is **turtle**.

1.6 SPARQL

SPARQL Protocol and RDF Query Language is a SQL-like query language for RDF data. It's a simple protocol for querying remote database over HTTP. It has different queries:
- `select`: projections of variables and expressions;
- `construct`: create triples based on query results
- `ask`: check whether a query returns results
- `describe`: describe resources in the graph

1.7 OWL

Web Ontology Language: knowledge representation language designed to formulate, exchange and reason with knowledge about a domain of interest.

2 Kadaster Knowledge Graph (KKG)

How is linked data is being used in the real world. Kadaster maintains a wide range of geographical (and not) registers. The knowledge graph is born to provide a faster way to make queries between registers, without having to rely on IDs. Can be used to detect tax evasion from different data. Challenges:
- Accessing data from geo-spatial services
- Provide easy access to your own data to avoid copies

Linked data was used because:
- it provides the semantics of the data so that there is a share understanding of what the data means;
- it ensures data authenticity to users;
- it provides information about how the data should be used.

Kadaster uses only 5-star **open** data.

Model timeline:

First Experiments no standardized ontologies, schema is based on the API schema

New Approach to LD publication reuse of international standards, ontologies are manually defined based on information models, non-standardized KKG model

Production Ready reuse of international **standards**, MIM[2] based ontologies, **standardized** KKG model.

There are key register silos for Kadaster, Topographies, Restrictions, Demographic data,….

Companies use either use knowledge graph or SQL database.

In triplestore, it should be possible to reference other knowledge graphs via SPARQL queries.

Lessons Learned

Governance
  - Adoption Strategy
  - Linksets: how two datasets are connected, who should create and maintain them?

[2]Meta model for information models

- Ownership

Modelling

- Modelling for a given use case
- Modelling resources

Scope

- Use case driven vs. data push driven: only use data that is useful

3 Ontology

Ontology is a branch of philosophy that studies concepts such as existence, being, becoming and reality. Ontologies work on a system of categories (substances, relations, states of affairs) that provides an encompassing classification of all entities.

DEF: Ontology

A data model that represents knowledge as a set of concepts within a domain and the relationships between these concepts.

An ontology is a formal[3] explicit[4] definition of a shared conceptualization[5].

Conceptualization refers to the theory, while the ontology is the artefact that represents this theory.

Ontologies are needed to share conceptualizations and semantics (meaning). They improve the understanding between people **and machines**.

3.1 Types of ontologies

Thesaurus list of words in groups of synonyms and related concepts;
Taxonomy categorization or classification of concepts;
Vocabulary body of words used in a particular natural language;
Ontologies concepts and their relations.

Thesauri, taxonomies and vocabularies are often seen as simpler forms of ontologies.

Ontologies can be classified based on their accuracy. Fine grained ontology is precisely defined to represent the intended models. Coarse grained ontology is a minimal set of axioms that rely on existing agreements.

Ontology Quality

- Syntax correctness;
- Semantic correctness: no logical inconsistencies, no unintended relations
- Ontological commitment;

Types of ontologies according to their generality

Top level/foundational ontology describes general concepts;
Domain ontology describes concepts related to a domain;
Task ontology describes concepts related to a task or activity;
Application ontology describes concepts of a particular domain and task, specializing the related ontologies.

[3]Formal: grounded in formalism (some logic)
[4]Explicit: explicitly specifying what is intended in the real world.
[5]Shared conceptualization: common understanding of the concepts.

3.2 Ontology Applications

Ontology is used to integrate originally disparate information systems and give them common meaning.

Ontology is a form of knowledge management. It captures the knowledge within an organization as a model, that can be queried by users to answer question and display relationships across an enterprise.

Data is found in many different forms. This makes it hard to understand existing relationships between existing data. Data needs to be represented in a format that allows relationships to be discovered. Ontologies allow these relationships to become visible.

3.3 Ontology Contents

The two standards that govern the construction of ontologies are Resource Description Framework (RDF), Web Ontology Language (OWL). In accordance with RDF and OWL, ontologies are made of classes and relationships. Classes and relationships can be combined to assert statements about the real world.

> Classes can have instances. I.e. Todd is an instance of the class Person.

The combination of classes and relationships is known as a triple, composed of a subject, predicate and object. Triples can be merged to provide a comprehensive view of the real world within an ontology.

Within an ontology concepts are defined in terms of relationships with other concepts.

In description logics (used in OWL2), ontologies can be separated in two parts:
TBox for universally quantified assertions (classes, properties and axioms)
ABox for assertions on individual objects (instance assertions)

> **Ontology Engineering**
>
> Many methodologies have been defined in literature, with different approaches. Different scenarios can be implemented by combining steps as described in the NeOn project.

Ontologies can be developed:
top-down from scratch, by extending foundation ontologies or by reusing ontological patters.
bottom-up by extracting knowledge from available resources.

Ontologies are easily extensible: instead of rewriting lines of code new relationships can be added to existing ontologies. Using ontologies for knowledge management is an alternative to source code. This approach is hard to manage, and can be managed only by a small group of engineers. Ontologies are a new way to manage knowledge and capture relationships. Ontologies are a suitable choice for a large changing organization because it allows more individuals to get directly involved and ease of modification.

3.4 Systematic Approach for Building Ontologies

SABIO is an ontology engineering methodology inspired by software engineering principles. It focuses on a specific scenario, usually the development of reference ontologies and their corresponding operational ontologies. The use of OntoUML is recommended for defining reference ontologies. It also recommends to ground ontologies in foundational ontologies. These simplify the modelling task due to their expressiveness and precision. The development phases are indicated as sequential, but any model can be used.

Reference ontology is a conceptual model mainly for communication and analysis.
Operational ontology is a computational artefact with some desirable computational properties.

These next steps are important for domain experts and ontology users.

3.4.1 Purpose and Requirements

Functional requirements are captured with competency questions: questions that the ontology is supposed to answer. Non-functional requirements are not related to contents. Complex ontologies should be decomposed in sub-ontologies.

3.4.2 Capture and Formalization

This process is supported by knowledge acquisition. It defines concepts, relations and constraints. A natural language definition of the axioms is written, followed by a formalization (first order logic or OCL). This step is formally iterative and results in a reference ontology.

3.4.3 Design and Implementation

The reference ontology and non-functional requirements are used to derive a compliant operational ontology design specification to be implemented. The operational ontology is then coded in the indented operational language (OWL2). Tools can help to translate the reference ontology to the language of the operational ontology.

3.4.4 Testing

In this phase, there is a strong involvement from the ontology tester and ontology user. Sub-ontologies should be tested separately fist (unit, integration and system tests). Tests should be *competency-questions driven* and black box. The ontology should be tested considering its intended use. Individuals are then generated to check whether the intended models are correctly represented.

3.4.5 Support Processes

Knowledge acquisition brainstorming, inspection of sources, interviews;
Reuse looking for similar ontologies, reuse of foundation ontologies, ontology patterns.
Documentation uniformity via document templates, naming conventions;
Evaluation verification (built correctly) and validation (right ontology);

4 RDF

> **DEF: Interoperability**
>
> The ability of two or more systems or components to exchange information and to use it.

> **DEF: Semantic Interoperability**
>
> Focus on implementation of shared data in an unambiguous way, ensuring that the understanding of information is the same for the senders and the receivers.

4.1 Naming

URI sequence of characters to uniquely identify a resource.
IRI extends URIs with unicode[6].
URN uniform resource name: resource identifiers with the specific requirements for enabling location independent identification for a resource
URL uniform resource locator, subset of URIs that in addition to identifying a resource, provide the means to locating the resource by describing its primary access mechanism.

[6]URIs and IRIs will be used interchangeably

4.2 Syntaxes

RDF/XML RDF syntax based on XML,
N-triples simple syntax for line delimited N-triples
Turtle Superset of N-triples, with shortcuts to facilitate readability
Notation3 Superset of of Turtle and RDF. Contains non-RDF features like rules, scoping graphs,…
RDFa RDF in attributes, it allows to embed RDF into XHTML, HTML or XML in general. It's a mix of machine and human readable language. This was mainly used by Google to scrape the web. Has some issues regarding readability, maintainability and FAIR principles.
JSON-LD JSON for Linked Data, created to facilitate the use of RDF by software developers.

RDF literals are used for values of predicates such as strings, numbers and dates (basic data types). All literals are between quotes and must have their data type specified in order to not be considered as strings (`"25"^^xmls#integer`).

4.3 RDF Types

4.3.1 Blank Nodes

Blank nodes represent the existence of something that was not given an IRI. When you don't want to create an identifier, you can use `_:BlankNode`. Especially useful with nested objects in a data structure. Blank nodes can only be usd locally.

4.3.2 Containers

Containers are an unbounded resource type that contains groups of resources (members). Containers are open, there is no mechanism to know the number of members.
Bag unordered set
Seq ordered set
Alt unordered set of alternatives (mutually exclusive), in which the first is the default option.

4.3.3 Collections

Collections are closed, members can be counted. They are linked lists (`rdf:List`). Some properties are `rdf:first`, `rdf:rest`, `rdf:nil`.

4.4 RDFS

RDF Schema extends RDF with the possibility to define classes and associated properties. Allows RDF applications to agree on a common description vocabulary.

RDF defines specific relations, such as:

- `rdf:type` (`my:City` $\xrightarrow{\texttt{rdf:type}}$ `rdfs:Class`)
- `rdf:property` and `rdf:subPropertyOf`
- `rdfs:domain`
- `rdfs:range`
- `rdfs:subClassOf`