From Ontologies to Knowledge Graphs

The Evolution of the Semantic Web towards Explainable AI and Retrieval Augmented Generation (RAG)

Ana Mª Fermoso García

Computer Sciences Faculty. Pontificia University of Salamanca (UPSA)

Content

- Semantic Web and Ontologies
- Evolution towards Knowledge Graphs (KG)
- RAG and the role of KGs in Explainable AI
- Practical Applicatiobn. Use case Archive of “Acción Católica Española” (ACE)

SEMANTIC WEB AND ONTOLOGIES

Fundamentals and state of the art

Universidad Pontificia de Salamanca

Content

- Introduction to Semantic Web
- Ontologies
- RDF
- OWL
- SPARQL

Semantic Web Introduction

Universidad Pontificia de Salamanca

Origin and visión of Semantic Web

- Evolution of the traditional web proposed by **Tim Berners-Lee** in the 2000s.
- Objectives
  - Data on the web is not only readable by humans, but also **understandable by machines**, allowing **automatic reasoning, interoperability and semantic retrieval**.
  - It seeks to represent knowledge through **structured and meaningful metadata**.

"The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning"

Berners-Lee, Hendler, & Lassila (2001), Scientific American.

Current Web vs Semantic Web

- **The Current Web links documents; The Semantic Web links data.**

- ***Current Web***
  - Increasing size => Problem efficient searches
  - Understandable by people but not by machines
- ***Semantic Web***
  - W3C late 90s
  - Machine Understandable Web content
    - Automation of tasks
    - More efficient search results
    - Facilitates location, sharing and integration of information and services to get more out of Web resources
    - Classifies, structures and describes Web resources for processing in software applications
  - Graph or hierarchical structure: network of nodes typified by classes and relationships
    - Nodes=Web resources; each resource associated with a type or meaning; Arcs=relationship between nodes or concepts

WEB SEMANTIC TECHNOLOGIES

- **ONTOLOGIES**
- Languages
  - **RDF**
    - Matadata and ontology definition
    - Based on XML
    - Triple chaining (node=subject // objet, arc=predicate)
    - RDFSchema: class hierarchy
  - **OWL**
    - RDF →DAML+OIL → OWL
    - RDF + classes through conditions, instances ennumeration, properties characteristics (cardinality, transitivity)
  - **SPARQL**
    - Ontology Query Language
- Tools for ontologies creation
  - Protege

Ontologies

Universidad Pontificia de Salamanca

ONTOLOGIES

- Some **definitions**...
  - Formal specification of a shared conceptualisation. Gruber (1993)
  - Common vocabulary for people who need to share information from a particular domain
  - Way to represent web knowledge in a machine-readable and reusable
  - Hierarchy of concepts with attributes and relationships to define semantic networks and interrelated information units.

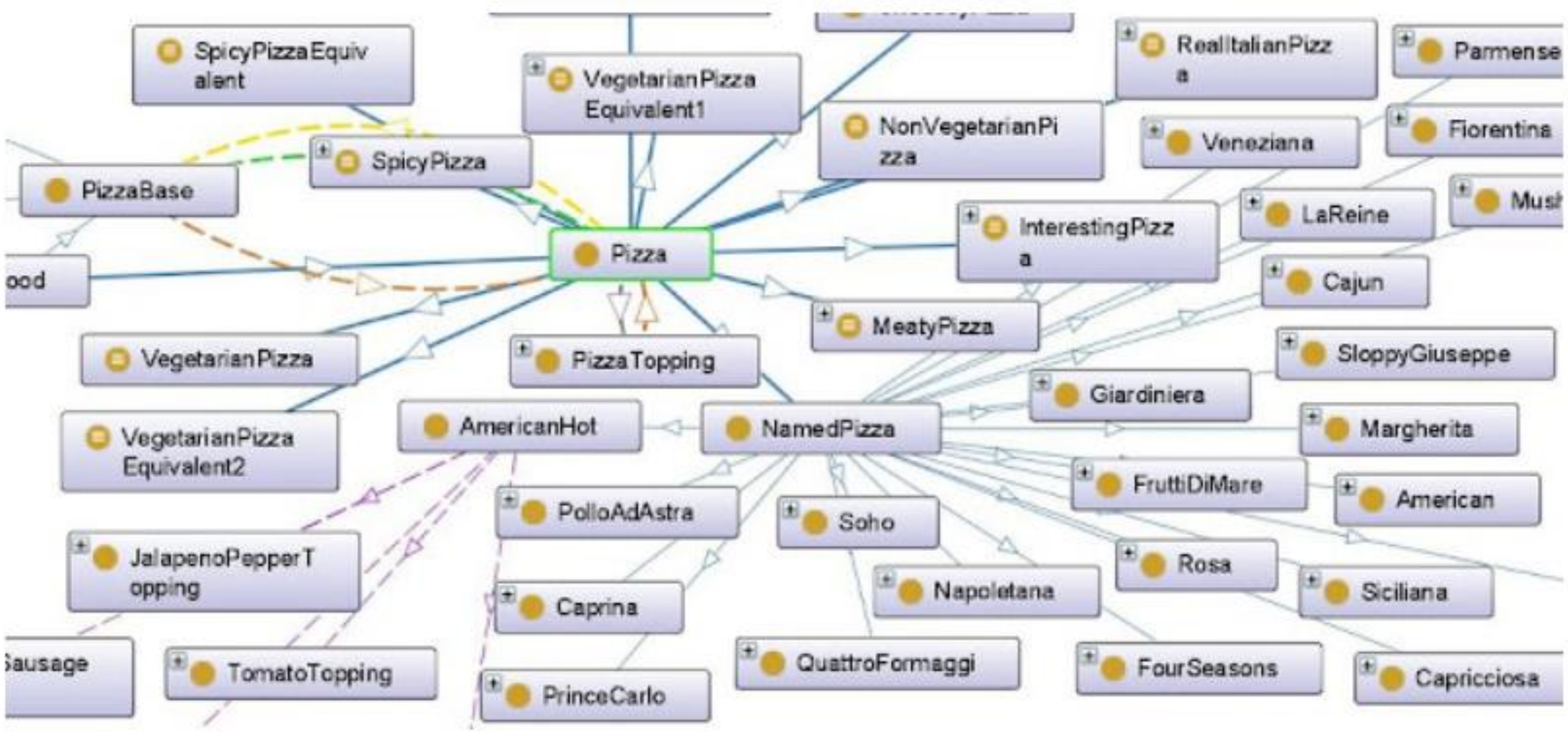
Components and advantages of Ontologies

- **Components** for representing knowledge...
  - **Concepts or classes**: basic ideas (also subclasses)
  - **Relations**: links between concepts
  - **Properties:** attributes of the classes
  - **Instances**: concrete objects of a concept
- **Advantages**
  - Knowledge reusing
  - Provides common understanding or knowledge between people, researchers or software agents
  - Separates knowledge domain from its management
  - Analysis of knowledge domain in terms of its concepts

Ontologies and Reference vocabularies

- **FOAF** (Friend Of A Friend) : Semantic social network
  - Persons descrition with their activities and relations with other people and objects
  - Concepts: person, name, fullname, mbox, knows, …
  - Prefix: http://xmlns.com/foaf/0.1/
- **vCard**
  - Exchange of personal data. Personal card and contacts,. Using in email.
  - Concepts: name, email, work, title, tel. …
  - Prefix : http://www.w3.org/2006/vcard/ns#
- **SKOS** (Simple Knowledge Organization System)
  - Model for representing the basic structure and content of conceptual schemes such as lists of subject headings, taxonomies, classification schemes, thesauri, ...
  - Concepts: prefLabel, altLabel, narrower, broader, closeMatch, inSchema, …
  - Prefijo: http://www.w3.org/2004/02/skos/core#
- **Dublin Core** (DC)
  - Standard for describing information resources and facilitating their retrieval. It started as a cataloguing format for libraries, but has now spread to other fields
  - Concepts: title, creator, subject, description, publisher, contributor, date, type, format, identifier, source, language, relation, coveraghe, rights
  - Prefix : http://purl.org/dc/elements/1.1/
- **Geonames**
  - Geographic database in semantoic format
  - Concepts: lat, long, …
  - Prefix : http://www.w3.org/2003/01/geo/wgs84_pos#
- **CIDOC-CRM:**
  - Cukltiral heritage description
  - Concepts: person, group, plave, event, activity, documernt…
  - https://cidoc-crm.org/

Example of ontology



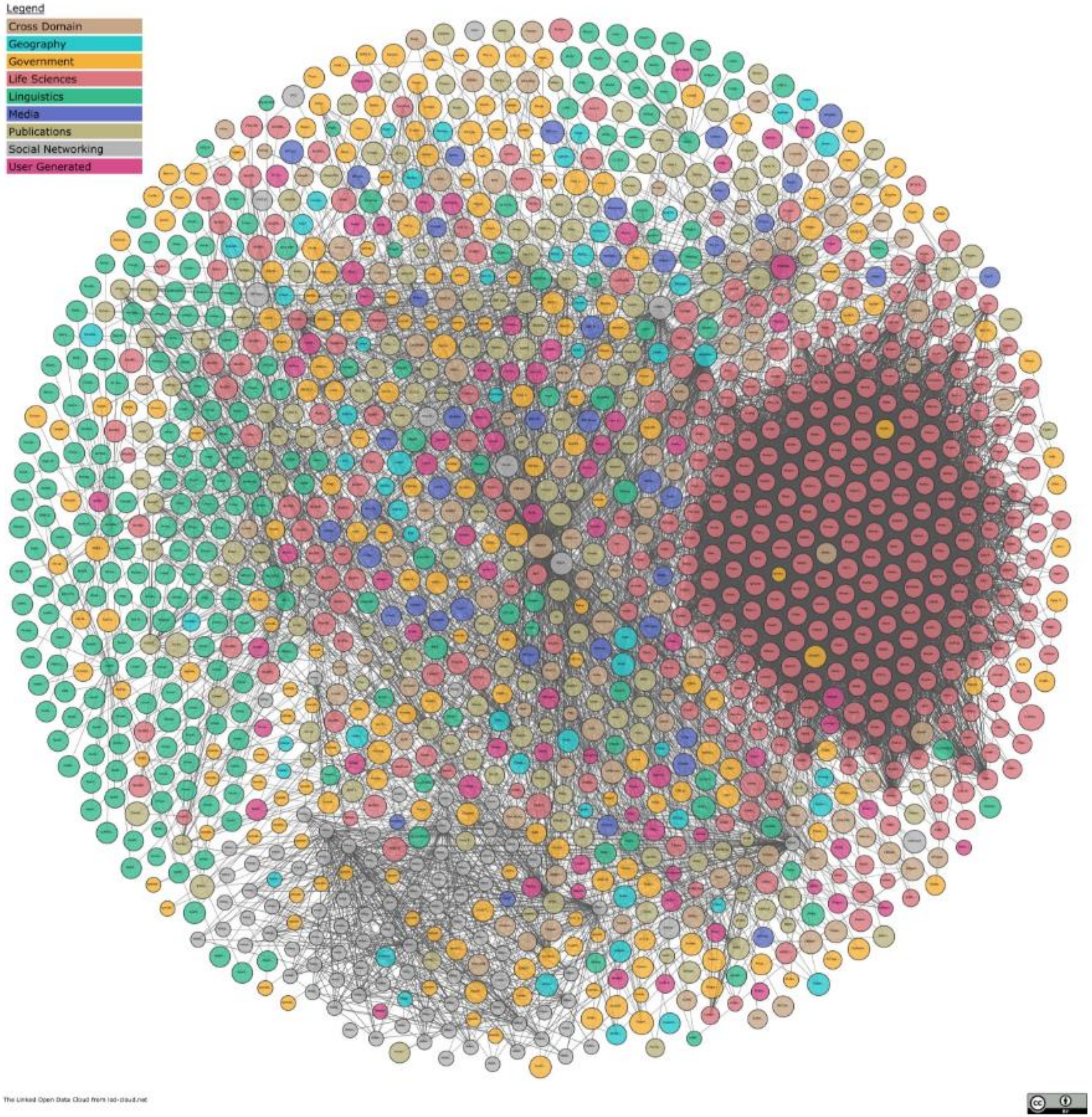
Metodology for Ontology Definition

1. Domain and scope of the ontology
   - What will it be used for, what questions will it answer, who will use it,...
2. Investigate reuse of ontologies
   - Libraries: Ontolingua, Swoogle/// Biomedical ontologies: OLS (https://www.ebi.ac.uk/ols/index), BRENDA
   - WebVOWL(http://www.visualdataweb.de/webvowl/): Ontology visualisation tool
3. Terms + important ontologies
4. Classes and hierarchical organization
   - Bottom-up and/or top-down approach
   - Hierarchy = ‘is a’
   - Consistent nomenclature
   - Identify synonymous classes, sibling, disjoint, multiple inheritance,...
   - Ontology scope
   - Class, property or instance - depends on context
5. Class properties
   - Intrinsic, extrinsic, parts of a structured object or relations with other individuals
6. Characteristics or facets of properties
   - Data typeAllowed and/or default values
   - Domain or range of values
   - Cardinality
   - Inverse properties
7. Creation of instances
   - Assigning values to properties

Applications and use cases of ontologies

- Key areas:
  - Semantic interoperability: different systems share a common vocabulary (e.g. libraries, museums, archives).
  - Heterogeneous data integration: facilitating federated queries and source reconciliation.
  - Automatic reasoning: logical inferences on existing data.
  - Domain modelling: precise structuring of knowledge in areas such as medicine, industry, culture or science.
- Examples:
  - *Europeana* and its use of EDM (Europeana Data Model), derived from RDF and OWL.
  - *DBpedia*: extracts structured knowledge from Wikipedia.
  - *Linked Open Data Cloud*: interconnected set of RDF datasets.

The Linked Open Data Cloud



RDF

Resource Description Framework

Universidad Pontificia de Salamanca

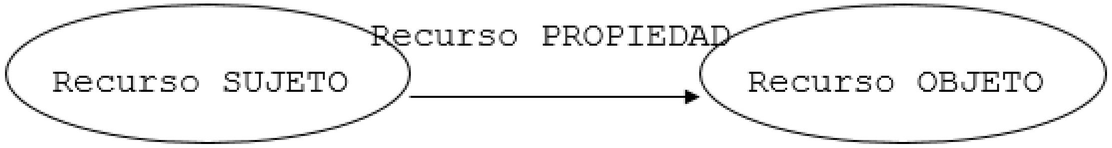
What is RDF (Resource Description Language)

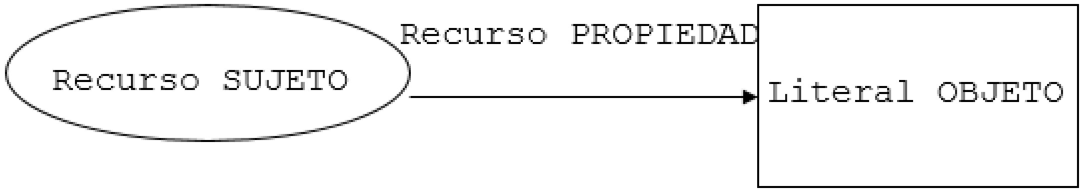
- *Definition*
  - *Infrastructure or mechanism used for the description of Web resources, but not restricted to any particular application or knowledge domain.*
  - *Facilitates information exchange and interoperability between applications by enabling automated processing of Web resources*
- *Features*
  - *XML-based syntax (RDF provides the model and XML provides the syntax to describe it)*
  - *RDF Triples: Each RDF statement/sentence consists of a subject (the resource), a predicate (the property or relationship) and an object (the related value or resource).*
  - *Unique identifiers: Uses URIs to uniquely identify each resource*
  - *Application areas: search engines, agents, information cataloguing, exchange and interoperability between applications, ...*

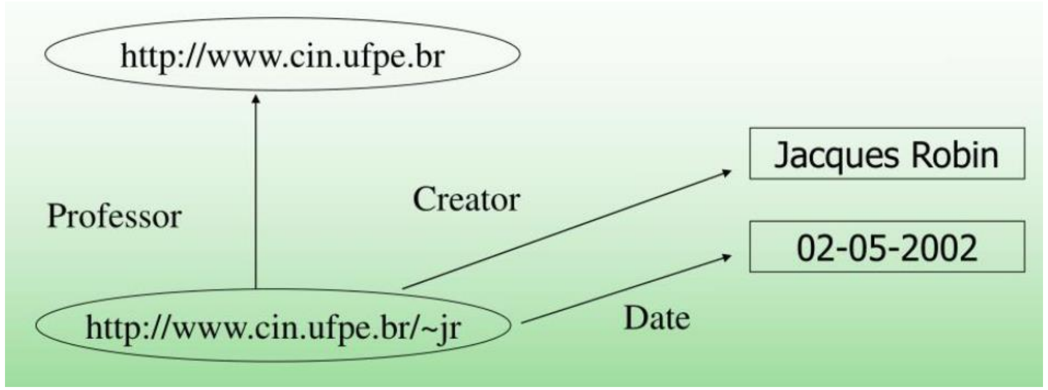
RDF MODEL

- Data model with three types of objects:
  - RESOURCES
    - Descriptions of RDF sentences.
    - URI identification
    - A resource can be a web page, a part of a web page, an element of a page, a collection of pages, ...
  - PROPERTY
    - Feature (attribute or relation) of a resource
    - Associated to: allowed values, other properties, resources that describes, ...
  - SENTENCES OR "TRIPLES"
    - A resource consists of three elements: subject, predicate and object
    - Subject: to whom the sentence refers / Object: to whom it is linked / Predicate: through whom it is linked.
    - Subjects and objects can be resources (URI) or literals. Predicates are always resources.
      - Literal: string, XML-tagged data or XML primitive type.

- **Graphical Representation of a Triple or Sentence :**







RDF CONCEPTS

- Three types:
  - Fundamentals (:rdf)
    - To represent the objects of the RDF data model
  - RDF Schema (:rdfs)
    - To create some resources from others => To define new vocabularies or rules to derive new resources
    - To define the context in which a resource is to be used
  - Utility

Fundamentals Concepts

- rdf:Resource
  - To describe resources. Its interpretation depend on context
- rdf:Property
  - Resource properties. They are also resources (a property may contain, among others, an attribute rdf:type, rdf:datatype=‘&xsd;tipoXML’, or none at all when its value is a literal or string)
  - Predicate statements
- rdf:Statement
  - Resource to define a triple.
  - Three parts: rdf:subject, rdf:object, and rdf:predicate.

Concepts of RDF Schema

- rdfs:subPropertyOf
  - Property indicating a relationship between resources (predicate)
  - If A is a subProperty of B, any triple that has subProperty A, will also have B (e.g. ‘mother’ subPropertyOf ‘parents’).
- rdfs:Class, rdf:type, rdfs:subClassOf
  - Class: resource that defines a set of resources. ‘mother’ subPropertyOf ‘parents’) rdfs:Class, rdf:type, rdfs:subClassOfClass: resource defining a set of resources
  - rdf:type:
    - Every resource has an rdf:type property indicating the class (rdfs:Class) to which it belongs
    - Every property has an rdf:type property indicating the property (rdf:Property) to which it belongs.
    - With rdfs:subClassOf and rdfs:subPropertyOf you define hierarchies of classes and properties.
      - All classes inherit (have as value of "subClassOf) from rdf:Resource
- rdfs:domain, rdfs:range
  - Property characteristics. Their value must be a class.
  - Domain: resources that can be assigned to a given property
  - Range: set of valid values that a property can take
  - A property can have several domains, but only one range value

RDF SINTAX

- XML language used to define metadata with the RDF model
- Formats: Serialized or abbreviated
- Descriptions
  - An RDF document is a set of descriptions
  - With the rdf:Description elements we define and/or use already defined resources
  - Each description
    - An ID attribute (definition) or about (already existing ref.)
    - Formed by a set of properties
    - Each property: resource (resource attribute) , literal or other description
    - If the property takes a specific value within a context: rdf:value -> <property rdf:value=‘context_value’>

Serialized sintax RDF

RDF ::= ['<*rdf*:RDF>'] description* ['</*rdf*:RDF>']

Description::= '<*rdf*:Description' idAboutAttr? '>' propertyElt* '</*rdf*:Description>'

idAboutAttr::= idAttr | aboutAttr

aboutAttr ::= 'about="' URI-reference '"'

idAttr ::= 'ID="' IDsymbol '"'

propertyElt ::= '<' propName '>' value '</' propName'>'

| '<' propName resourceAttr '/>'

| ‘<' propName rdf:datatype=“&xsd;tipo”>val</propName '/>'

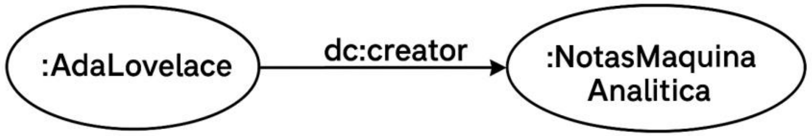
propName ::= Qname

value ::= description | string

resourceAttr ::= 'resource="' URI-reference '"'

RDF formats and examples

Subject: :AdaLovelace
Predicate: dc:creator
Objet: :NotasMaquinaAnalitica



turtle

```turtle
@prefix : <http://ejemplo.org/> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .

:NotasMaquinaAnalitica dc:creator :AdaLovelace .
```

xml

```xml
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/"
         xmlns:ex="http://ejemplo.org/">

  <rdf:Description rdf:about="http://ejemplo.org/NotasMaquinaAnalitica">
    <dc:creator rdf:resource="http://ejemplo.org/AdaLovelace"/>
  </rdf:Description>
```

json

```json
{
  "@context": {
    "dc": "http://purl.org/dc/elements/1.1/"
  },
  "@id": "http://ejemplo.org/NotasMaquinaAnalitica",
  "dc:creator": {
    "@id": "http://ejemplo.org/AdaLovelace"
  }
}
```

OWL

Ontology Web Language

Universidad Pontificia de Salamanca

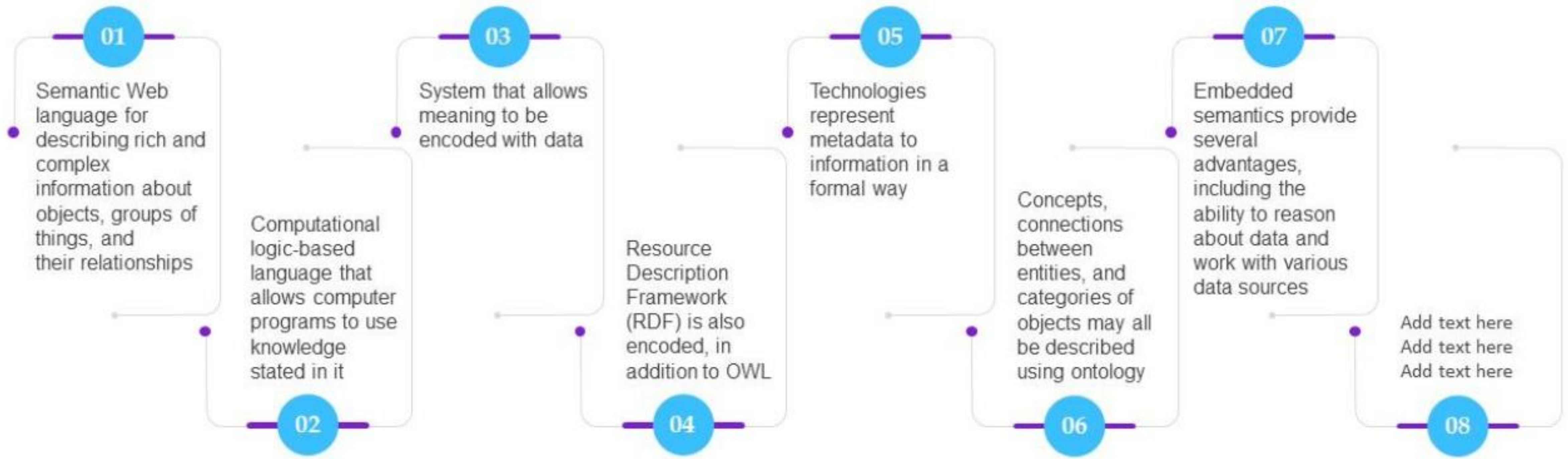
What is OWL (Ontology Web Language)

- Definition:
  - Allows to represent the meaning of the terms of a vocabulary about a specific area of knowledge and the relationships between these terms.
  - It allows to represent formalized and logical knowledge.
  - It defines classes, properties and relations between concepts. This representation of terms and relations between them is what we call ontology.
- Characteristics:
  - Definition of classes, properties and relations: Allows to specify classes, properties, relations and restrictions between them.
  - Automatic reasoning: Facilitates the inference of new information from existing data.
- Base languages OWL (derived from RDF)
  - XML (syntax) -> XML Schema (doc.XML structure) + RDF (data model) + RDF Schema (classes, properties, hierarchy) -> OWL (+ vocabulary for classes and properties: cardinality, equality, relations,...)

Header document Ontology

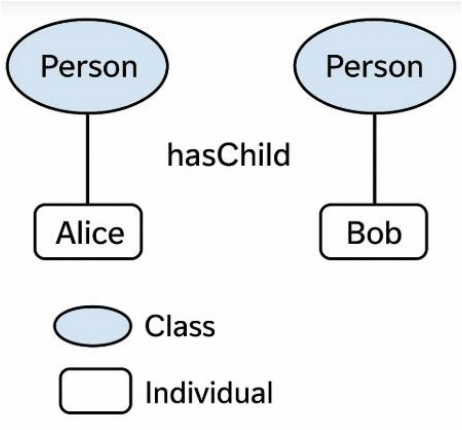
```
<rdf:RDF
    xmlns:owl ="http://www.w3.org/2002/07/owl#"
    xmlns:rdf ="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:xsd ="http://www.w3.org/2001/XMLSchema#">
```

What is OWL?


01
Semantic Web language for describing rich and complex information about objects, groups of things, and their relationships
02
Computational logic-based language that allows computer programs to use knowledge stated in it
03
System that allows meaning to be encoded with data
04
Resource Description Framework (RDF) is also encoded, in addition to OWL
05
Technologies represent metadata to information in a formal way
06
Concepts, connections between entities, and categories of objects may all be described using ontology
07
Embedded semantics provide several advantages, including the ability to reason about data and work with various data sources
08
Add text here
Add text here
Add text here

Example and syntax of OWL

- A *Person* is a class, and *hasChild* is a property whose domain is *Person* and range is also *Person*
- *A person can have children*
- Alice is a person / Bob is a person / Alice hasChild Bob



xml

```xml
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
         xmlns:owl="http://www.w3.org/2002/07/owl#"
         xmlns:ex="http://ejemplo.org#">

  <owl:Class rdf:about="http://ejemplo.org#Person"/>

  <owl:ObjectProperty rdf:about="http://ejemplo.org#hasChild">
    <rdfs:domain rdf:resource="http://ejemplo.org#Person"/>
    <rdfs:range rdf:resource="http://ejemplo.org#Person"/>
  </owl:ObjectProperty>

  <owl:NamedIndividual rdf:about="http://ejemplo.org#Alice">
    <rdf:type rdf:resource="http://ejemplo.org#Person"/>
    <ex:hasChild rdf:resource="http://ejemplo.org#Bob"/>
  </owl:NamedIndividual>

  <owl:NamedIndividual rdf:about="http://ejemplo.org#Bob">
    <rdf:type rdf:resource="http://ejemplo.org#Person"/>
  </owl:NamedIndividual>
</rdf:RDF>
```

plaintext

```
Class: Person

ObjectProperty: hasChild
    Domain: Person
    Range: Person

Individual: Alice
    Types: Person
    Facts: hasChild Bob

Individual: Bob
    Types: Person
```

SPARQL

Semantic Query Language

Universidad Pontificia de Salamanca

What is SPARQL

- Query language for RDF and OWL
  - Enables complex queries over RDF graphs, extracting information based on triple patterns.
- Includes functionality for filtering results, sorting, limiting and aggregation.
- Quite similar to SQL

Query formats

- SELECTS
  - Selection on data query
- CONSTRUCT
  - Construction of RDF triples
- ASK
  - Questioning the dataset and answering with ‘Yes’ or ‘No’
- DESCRIBE
  - Description of a resource (e.g. we can know its properties)

SELECT

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:foaf="http://xmlns.com/foaf/0.1/"
    xmlns:admin="http://webns.net/mvcb/">
 <foaf:Person rdf:ID="me">
        <foaf:name>Ana Fermoso</foaf:name>
        <foaf:givenname>Ana</foaf:givenname>
        <foaf:family_name>Fermoso</foaf:family_name>
        <foaf:nick>AFG</foaf:nick>
        <foaf:mbox>afermosoga@upsa.es</foaf:mbox>
        <foaf:homepage rdf:resource="http://afermoso.es"/>
        <foaf:phone rdf:resource="tel:123456789"/>
        <foaf:workplaceHomepage rdf:resource="http://www.upsa.es"/>
        <foaf:workInfoHomepage rdf:resource="Catedrática universidad"/>
        <foaf:knows>
                <foaf:Person>
                        <foaf:name>Nuria</foaf:name>

        <foaf:mbox>nn@gmail.com</foaf:mbox></foaf:Person></foaf:knows>
        <foaf:knows>
                <foaf:Person>
                        <foaf:name>Pedro</foaf:name>
                <foaf:mbox>pp@hotmail.com</foaf:mbox>

                <rdfs:seeAlso
rdf:resource="http://www.pp.es"/></foaf:Person>

        </foaf:knows>

 </foaf:Person>

</rdf:RDF>
```

- SELECT structure

SELECT ?var1 ?var2 ..

WHERE {

condition1 .

condition2 .

… }

- Examples of conditions
  - **With literal: “data”**

SELECT ?x
WHERE {
    ?x foaf:name “Ana Fermoso”
}

  - **With number:**

SELECT ?x
WHERE {
    ?x ?y 30
}

  - **With literal in a language: “data”@es**

SELECT ?x
WHERE {
    ?x foaf:name “Ana”@es

}

From ONTOLOGIES to KNOWLEDGE GRAPHS (KG)

Evolution and convergence

Universidad Pontificia de Salamanca

Content

- Evolution from semantic web to KG
- Ontologies vs KG
- Examples & comparative

From Semantic Web to Knowledge Graph (KG)

The Semantic Web, proposed by Tim Berners-Lee, sought to give the web machine-understandable meaning through the use of ontologies and technologies such as RDF and OWL.

With the exponential growth of data and the need for more dynamic applications, knowledge networks emerged.

Knowledge Graphs (KGs): pragmatic evolution that combines the formality of ontologies with the flexibility of graphs.

Ontologies. Characteristics and limitations

- **Characteristics:**
  - Formality: Based on descriptive logics, they allow inferences and automatic reasoning.
  - Standards: Use languages such as OWL and RDF.
  - Applications: Interoperability, data integration and knowledge modelling (Wikidata)
- **Limitations:**
  - Rigidity: Difficult to adapt to rapid changes in data.
  - Learning curve: Require specialized knowledge for creation and maintenance.
  - Scalability: Can be complex to scale in environments with large volumes of data.

KG. Definition and advantages

- Definition
  - A knowledge graph is a data structure that represents entities and their relationships in the form of nodes and edges, integrating data from various sources and allowing efficient inferences and queries.
- Advantages
  - Flexibility: Easy integration of new data and relationships.
  - Scalability: Suitable for large volumes of heterogeneous data.
  - Interoperability: Facilitates connection between different domains and systems.
  - Applications: Search engines (Google Knowledge Graph), virtual assistants, recommendation systems,...

Ontologies vs Knowledge graphs

Characteristic	Ontologies	Knowledge Graphs
Purpose	To formally and logically represent a domain	To integrate, link, and contextualize real-world data
Level of Formalism	High (based on descriptive logic: OWL, DL)	Variable (can range from informal to OWL-compatible)
Components	Classes, properties, axioms, restrictions	Entities, relationships, attributes (often RDF or JSON-LD)
Orientation	Domain modeling, semantic interoperability	Practical application, retrieval, data navigation
Maintenance	Requires experts	Can be partially automated
Reasoning	Explicit, logic-based (formal inferences)	Implicit or explicit; does not always require formal logic
Examples	CIDOC-CRM, FOAF, Dublin Core	Google KG, Wikidata, DBpedia, Facebook KG

- **Convergence:** *Ontologies as the basis for knowledge graphs*
  - Knowledge graphs offer more flexibility but ontologies provide them with the necessary semantic structure and are sometimes used to define entities and relationships.

Example and comparison

“Miguel de Cervantes writes the Quijote”

- Ontology (OWL/RDF)
  - Firstly it has been defined: “Person”, “Book” and “authorOf” relation

```xml
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:ex="http://example.org/">

  <rdf:Description rdf:about="http://example.org/Miguel_de_Cervantes">
    <rdf:type rdf:resource="http://example.org/Person"/>
    <ex:authorOf rdf:resource="http://example.org/Don_Quijote"/>
  </rdf:Description>

  <rdf:Description rdf:about="http://example.org/Don_Quijote">
    <rdf:type rdf:resource="http://example.org/Book"/>
    <ex:hasTitle>Don Quijote</ex:hasTitle>
  </rdf:Description>

</rdf:RDF>
```

- Knowledge Graph (JSON-LD o RDF)
  - No axioms or formal logical inferences, but structured representation to link data, integrate it and answer questions like "Who is the author of Don Quijote? –
  - Ideal for RAG and intelligent search!

```json
{
  "@context": {
    "name": "http://schema.org/name",
    "authorOf": "http://schema.org/author",
    "Book": "http://schema.org/Book",
    "Person": "http://schema.org/Person"
  },
  "@type": "Person",
  "name": "Miguel de Cervantes",
  "authorOf": {
    "@type": "Book",
    "name": "Don Quijote"
  }
}
```

RAG & Knowledge Graphs (KG)

KG and XAI (Explainable AI)

Universidad Pontificia de Salamanca

Content

- KG & IA
- KG Introduction
- RAG Technogies. *Vectors vs. KG*
- KGs

Knowledge Graph (KG) & AI

Explainable AI (XAI)

Universidad Pontificia de Salamanca

KG & IA. Importance of KG in IA

AI & structured data

- Language models (LLMs) are very good at generating text, but they lack structured, up-to-date or verifiable knowledge.
- KGs allow the addition of explicit context, semantic relations and disambiguation, something that vectors alone cannot guarantee.

Knowledge and inference

- Although formal logic is not always used, KGs allow for relationship navigation and graph-based inference (traversal, similarity, clustering).

Explainability of AI (XAI)

- ***KGs make the path of knowledge visible (e.g., Cervantes → author of → Don Quixote), while a text model functions as a 'black box'.***
- This makes them key in environments where traceability is required, such as heritage, health, justice or education.

RAG (Retrieval-Augmented Generation)

- RAG combines text generation with external information retrieval. KGs enrich this process:
- They are used as a structured source.
- They guide retrieval with semantic context.
- They allow enriching answers with metadata or useful connections.

Integration with NLP

- ***It is possible to build KGs from NLP-processed texts (NER, relations), and then use them as the system's 'semantic memory'.***

Use of RAG for Explainable AI (XAI)

- Explainable AI (XAI): processes and methods that allow humans to understand and trust the results of LLMs (not see them as a black box)
- RAG: Combines generative (LLMs) and retrieval (from authoritative sources) models generating more relevant and accurate answers.
- **Knowledge Graphs in RAG for Explainable AI (KG-RAG)**
  - Knowledge graphs guide the retrieval and organization of relevant pieces of information, which are then used to generate more accurate and explainable answers
  - Advantages:
    - **Structured context**: KGs provide explicit relationships between entities, allowing more coherent and grounded answers to be generated.
    - **Multi-hop reasoning**: They facilitate the tracking of multiple inference steps, improving the system's ability to answer complex questions.
    - **Traceability**: Each element of the answer can be linked to specific nodes and relationships in the KG, improving transparency and user confidence.

Some real applications with KG

- **Google Knowledge Graph**
  - Enhances search with structured information panels.
  - Displays entities, relationships, dates and context.
- **Wikidata**
  - Multilingual collaborative database.
  - Supports Wikipedia, virtual assistants and semantic visualisation.
- **Dbpedia**
  - Extracts structured knowledge from Wikipedia.
  - Pioneer of Linked Open Data.

Some real applications with KG

- **Health sector**
  - Relationship between diseases, treatments, patients (e.g. Mayo Clinic, Roche).
- **E-commerce**
  - More precise recommendations on Amazon and Alibaba, based on semantic relationships.
- **Cultural heritage**
  - Modelling with **CIDOC-CRM** in museums, libraries and archives (Europeana, British Museum).
- **Artificial Intelligence (AI)**
  - KGs integrated in RAG enrich answers generated by models such as GPT, providing traceability and context.

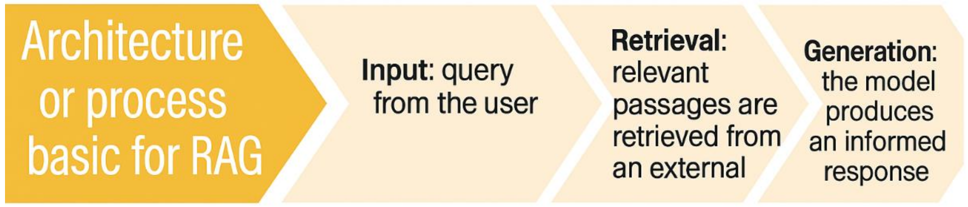
RAG Introduction

Universidad Pontificia
de Salamanca

What is the Retrieval Augmented Generation (RAG)

- **RAG:** Mechanism to optimise the output of an LLM (Large Language Model) to ensure that they generate their information in response to specific and updated external data in order to obtain relevant, accurate and contextually appropriate results.
- RECOVERY & GENERATION:
  - **LLM for text generation + Information retrieval from authoritative** (databases, documents, structured systems, ...) and up-to-date knowledge sources **== More appropriate and useful answers**

RAG process and when can be used



- When and why use RAG
  - **When up-to-date** or specialized **information is needed.**
  - In sensitive domains (legal, health, heritage) **where accuracy is key**.
  - To improve model **robustness and explainability**.
  - To **reduce hallucinations** in LLMs.

RAG vs LLM (i)

- **LLMs hallucinations**
  - Static and outdated data sources
  - Generic information not contextualized
  - Inaccurate answers and/or from non-authoritative sources
- **RAG advantages**
  - Dynamic updating of information ensuring to provide recent and valid information
  - Accuracy in specific answers even with references => greater confidence and reliability for users
  - Smaller models by delegating knowledge to authoritative bases => cost savings by training the model
  - More control for developers => control over information sources to adapt to changing requirements and generate appropriate answers
- **With and Without RAG**
  - Leverages up-to-date data and generates useful, targeted responses.
  - LLM without RAG: Limited to pre-trained knowledge, providing generic or inadequate answers.

RAG vs LLM (ii)

- **Example: Answer genereation for technical support depatment**
  - **Without RAG (only LLM):**
    - A company has a trained LLM until 2022. A customer asks: "What steps do I need to take to configure product X with the latest software update released this month?“
    - The LLM replies, "I have no information on the latest update. Please refer to the documentation.“
  - **Scenario with RAG:**
    - The same company implements RAG, connecting the LLM to a vector database that indexes updated technical documentation, software bulletins and internal FAQs. The customer asks the same question: "What steps do I need to take to configure product X with the latest software update released this month?“
    - The RAG system consults the base and replies:*:

      "To configure Product X with the latest software update (version 5.2.1 released in November 2024), follow these steps:
      - Download the firmware from the link: [updated link].Connect the device to your PC using the USB cable.
      - Run the configuration tool and select “Update”.
        For more details, see the guide here: [link to documentation].“
- **Example: assisting students in a university course**
  - **Non-RAG scenario (LLM only):**
    - A student asks, "What are the deadlines for turning in projects in the Artificial Intelligence course this semester?“
    - The LLM responds: "Sorry, I don't have specific information about your course deadlines. Please check with your instructor or check the student portal".
  - **Scenario with RAG:**
    - The same system implements RAG, connected to the university's learning management system (LMS), which contains deadlines, course content and teacher updates. The student asks the same question, "What are the deadlines for submitting projects in the Artificial Intelligence course this semester?"
    - The RAG system queries the LMS and responds:
      *"The deadlines for projects in the Artificial Intelligence course are as follows: Project 1: 30 November 2024 / Project 2: 15 December 2024. Make sure you upload your submissions to the portal before 11:59 PM. For more details, see the link: [link to LMS]. "

How RAG works. Steps to create a RAG system (i)

1. Definition **USE CASES**
   - Objective: problem to be solved and answers to be provided
   - End-users: profile and expectations

   E.g.: different citizens' profile questions on tax declaration issues based on manuals, legislations,...

2. Selection and Updating **EXTERNAL DATA SOURCES**
   - Give where to retrieve the information based on context or use cases and ensure its permanent updating
     - Possible sources: documents (PDF, manuals, presentations,...); databases (relational or non-relational); structured data (knowledge graphs, APIs); others (reference and trusted websites)
   - Ensure data is updated, organized and accessible

   E. g.: Tax office chatbot for tax return help: PDFs legislation, BOEs, manuals, FAQs, official website of the tax office...

How RAG works. Steps to create a RAG system (ii)

3. **INDEXATION** of data
   - From external sources to formats to facilitate retrieval and search:
     - Vector databases: generation of embedding from data sources using pre-trained models
     - Knowledge graphs: modelling relationships between key concepts and data => define entities, relationships and rules for logical and inferential queries
4. **RETRIEVAL** mechanism of relevant information
   - Translation of user questions (prompt) to queries on knowledge bases
     - By semantic similarity (vector bases)
     - Semantic queries (SPARQL) in graphs

       Ex: Deadline for filing income tax return?

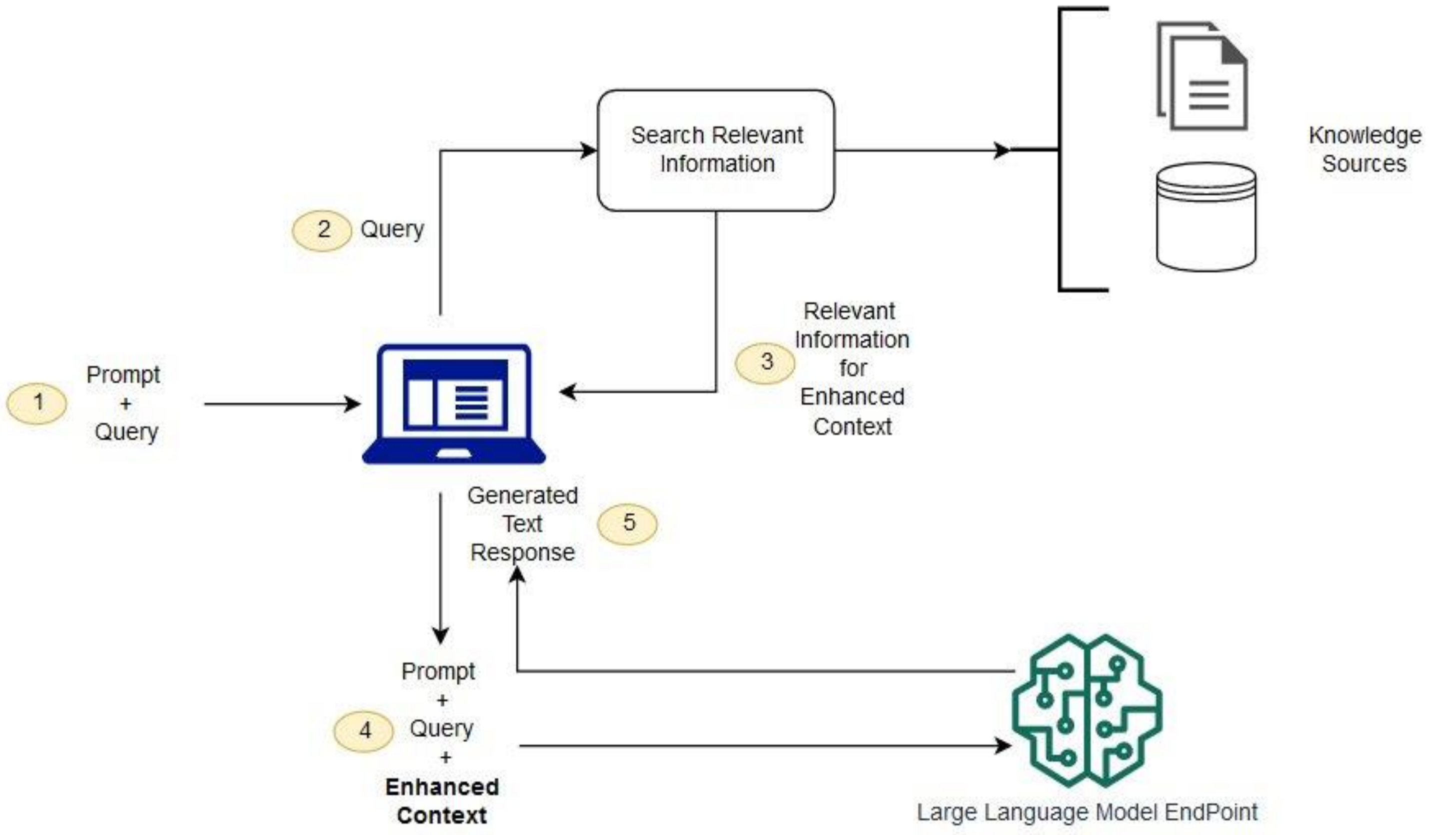
       Vector: returns FAQ documents, regulations or manuals

       Graphs: relation 'procedures and dates'.

How RAG works. Steps to create a RAG system (iii)

5. **CONNECT and ENHANCE** generative **LLM** data model=> increase user input (prompts) by adding context relevant retrieved data
    - Integrate retrieval system into generative model:
        - Query ->Retrieve->LLM input->Generate response
    - Adjust/enrich model prompt
        - E.g. prompt: "based on the following information XX, answer question YY”
6. **EVALUATION and ADJUSTMENT** of the system
    - Quality metrics:
        - Recall (% relevant information retrieved) / Accuracy (% correct information retrieved) / Accuracy (how correct and useful are the answers)
    - Adjustments
        - Search engine optimization for retrieval
        - Refine LLM prompts

How RAG works. Steps to create a RAG system



Source: https://aws.amazon.com/es/what-is/retrieval-augmented-generation/

RAG by programming

1. **Load documents** that form the knowledge database
   - **Langchain** library allows to load documents (loaders) from different sources (PDF, web, ...).)
2. **Split loaded documents** into text fragments
   - Langchain Splitters
3. Index (**embedding**) or convert text fragments (to vector or graph) for **storage in DB**
   - In Vector DB (for example Chroma) or
   - In Knowledge Graph DB (for example Neo4j)
4. **Retrieval** of information based on user prompt and fluent response generation
   - Conversion of queries (prompt) from natural language to DB graph query language (from NLP to Cyber or SPARQL **with Langchain and selected LLM**, OpenIA, for example)

RAG & Chatbots

- **Advantages** of RAG with Chatbots
  - Updated answers
    - Ex: answers to company administrative policies, product availability, recent events...
  - Improved user experience: More useful interactions, relevant content adapted to context and profile, reduced hallucinations,...
    - Ex: Detailed instructions for filing tax returns
  - Improved user experiences
- Some **examples of use**:
  - Customer support (access to FAQs and doc. Updated)
  - Medical assistance (drug databases and trusted protocols: e.g. side effects of drug x?)
  - E-commerce (e.g. product availability or status of a specific order XXX)
  - Personalized education (e.g. advanced level exercises to practice physics topic X)
  - Legal chatbot (e.g. Clarify clause x of rental contract)

RAG Technologies

Vector Databases

vs

Knowledge Graphs

Universidad Pontificia
de Salamanca

RAG Technologies

Tecnologías RAG

- **VECTOR DATABASES**
  - Fast searches in large volumes of unstructured text (document searches)
- **KNOWLEDGE GRAFTS (KG)**
  - Relationships between data critical to answers (legal queries (laws and cases))

RAG Technologies

VECTOR DATABASES

- Definition:
  - Store and retrieve vectors (embeddings). Vector: mathematical representation of text or image that encodes semantic meaning, facilitating semantic search.
- Functioning
  - Generation of embeddings or vectors (conversion of text into vectors) -> Storage in vector databases -> Query for nearest and most relevant vectors
- Advantages and disadvantages
  - ☺ Speed search in large volumes of data/documents and ease of integration (e.g. LangChain Framework)
  - ☹ Dependency on embeddings generation model, limitation for complex relationships, only finds semantically similar documents
- Examples:
  - FAQs (business, care etc.)
  - Medical care based on scientific articles
  - E-commerce with personalized recommendations
  - Example technologies: BD FAISS, Pinecone or Weaviate

KNOWLEDGE GRAPH (KG)

- Definition:
  - Allows modelling complex data and making logical inferences based on organisation in graphs: set of entities (nodes) and their relationships (edges).
- Operation
  - Graph construction (from data to graphs (concepts (nodes) and relationships (edges)) -> Prompts conversion to structured queries (SPARQL or Cyber) -> Response generation
- Advantages and disadvantages
  - ☺ Logical reasoning (inferred) and complex; explainability by showing relationships between data to support response; ideal for highly interrelated domains
  - ☹ Higher initial complexity for modelling relationships and in large graphs that can slow down queries. Lack of standardization and challenges specific to each domain (terminology, ontologies,...)
- Examples:
  - Legal analysis (relations laws, contracts, precedents...)
  - Personalised education Finance (risks, markets, assets...)
- Example TECHNOLOGIES: Neo4j (Cyber), GraphDB (RDF and SPARQL)

Vector Databases vs Knowledge Graphs (i)

Characteristic	Knowledge Graphs	Vector Databases
Data Representation	Entities (nodes) and relationships (edges) between entities, forming a graph structure.	High-dimensional vectors, each representing a data point (e.g., document or sentence).
Retrieval Mechanisms	Traverse the graph structure and follow relationships between entities. Enables inference and derivation of new knowledge.	Vector similarity based on a similarity parameter (e.g., cosine similarity). Returns the most similar vectors and the associated information.
Interpretability	Knowledge representation interpretable by humans. The graph structure and labeled relationships clarify the connections between entities.	Less interpretable for humans due to high-dimensional numeric representations. Hard to understand the relationships or reasoning behind retrieved data.
Knowledge Integration	Facilitates integration by representing entities and relationships in a unified graph structure. Perfect integration if entities and relations are correctly assigned.	More difficult. Requires techniques like vector space alignment or set-based methods to combine information. Ensuring vector compatibility can be non-trivial.
Inferential Reasoning	Enables inferential reasoning by traversing the graph and leveraging relationships between entities. Discovers implicit connections and derives new information.	More limited. Depends on vector similarity and may overlook implicit relationships or inferences. Can identify similar information but not complex graph relationships.

Source: https://www.datacamp.com/es/tutorial/knowledge-graph-rag

Vector Databases vs Knowledge Graphs(ii)

- RAG **limitations if only vectors** (embeddings)
  - **Loss of semantic structure**: explicit relationships between entities are not preserved.
  - **Difficulty in navigating complex relationships** (e.g., multihop reasoning).
  - **Poor transparency**: we do not know why a chunk was retrieved (opacity).
  - **Sensitivity to malformed context**: if an entity is ambiguously named, the vector has no way to disambiguate.
  - **No explicit reasoning**: logical paths between concepts cannot be inferred or explained

- **Example**:
  - Query: "What is the relationship between Miguel de Cervantes and the battle of Lepanto? "
  - A RAG system with only vectors can retrieve text where both words appear, **but without knowing that Cervantes participated as a soldier**, nor why this is relevant to the question.

- **Why the introduction of KG in RAG**
  - ***Semantic context:***
    - *Cervantes → participated in → Battle of Lepanto relationships are represented, which better guides retrieval.*
    - *They allow **disambiguation of entities** ("Amazon" company or river?).*
  - ***Traceability:***
    - *Each KG node and relation is referenced, so the system can explain:*
    - *"This information comes from node X which is connected to Y via property Z".*
  - ***Explainability:***
    - *The user can see or audit the logic behind the answer.*
    - *The **reasoning can be navigated***

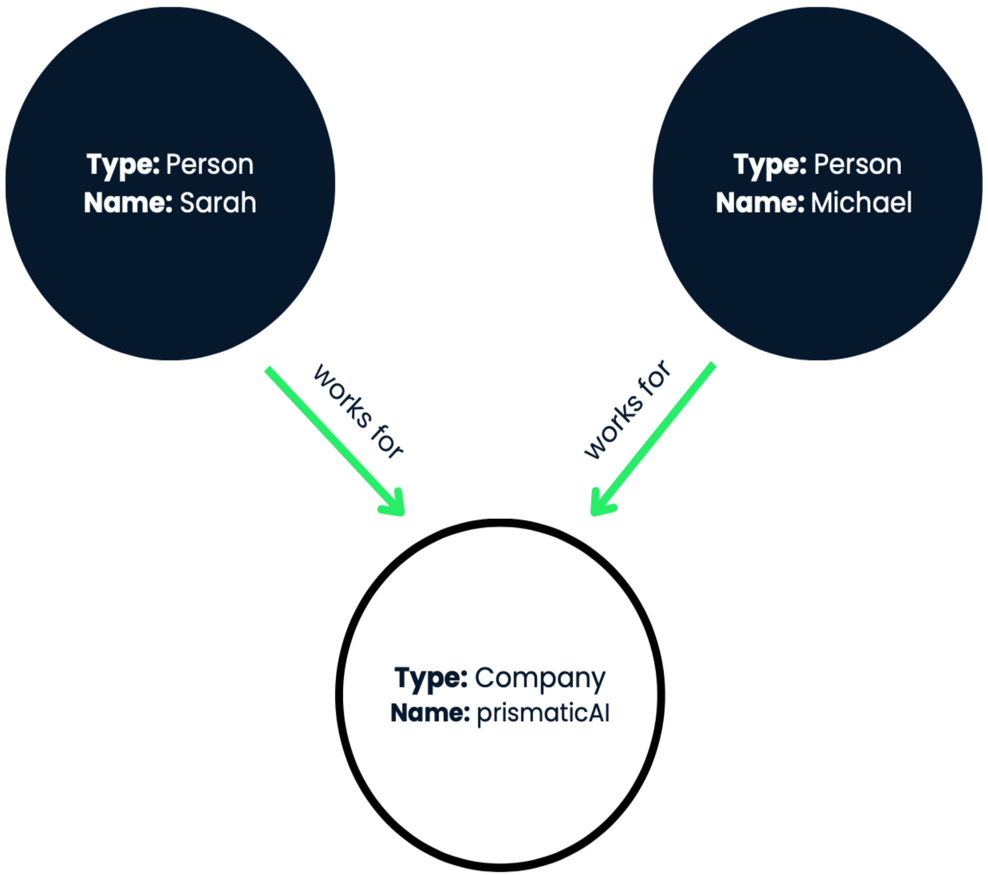
Use of Knowledge Graphs (KG)

Future of RAG

Universidad Pontificia de Salamanca

Knowledge Graph

Representation



Source: https://www.datacamp.com/es/tutorial/knowledge-graph-rag

Knowledge Graph

KG Use. Steps (i)

1. Conceptual model

- Ejemplo:
  - Nodes:
    - Person: "Juan", "María"
    - Enterprise: "Empresa X", "Empresa Y"
    - City: "Madrid", "Barcelona"
  - Links:
    - “live in (vive en)", “work in (trabaja en)", “situated in (situada en)“
  - **Ex KG** = María *works (trabaja en)* Empresa X and lives in (vive en) Madrid

2. Steps for KG creation

1. Domain definition
2. Entity identification. Nodes creation.
3. Relations definition
4. Transform to structured format with IA-KG tools and technology selection (Ex.: KG creation with Cyber language n Neo4j or with SPARQL in GrapfDB)
5. Query the KG

- Crear nodos:

```
cypher
Copiar código
CREATE (p:Persona {nombre: "Juan", edad: 30})
CREATE (c:Ciudad {nombre: "Madrid"})
```

- Crear relaciones:

```
cypher
Copiar código
MATCH (p:Persona {nombre: "Juan"}), (c:Ciudad {nombre:
"Madrid"})
CREATE (p)-[:VIVE_EN]->(c)
```

- Encuentra personas que viven en una ciudad:

```
cypher
Copiar código
MATCH (p:Persona)-[:VIVE_EN]->(c:Ciudad {nombre:
"Madrid"})
RETURN p.nombre
```

KG Use. Steps (ii)

3. Steps to leverage the Knowledge Graph for RAG
    1. Convert question to structured query (use NLP to translate prompt to SPARRQL or Cyber)
    2. Query Graph (query execution)
    3. Generate answer (combine structured result with generative model to get smooth answer)

Grafos Conocimiento

Use of KG (in SPARQL y Cyber)

```
@prefix ex: <http://example.org/> .
@prefix foaf: <http://xmlns.com/foaf/0.1/>

ex:Juan a foaf:Person ;
    foaf:name "Juan" ;
    foaf:age "30" ;
    ex:livesIn ex:Madrid .

ex:Madrid a ex:City ;
    ex:name "Madrid" .
```

Consulta SPARQL para obtener personas que viven en una ciudad específica:

```
sparql
Copiar código
PREFIX ex: <http://example.org/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?person ?name WHERE {
  ?person a foaf:Person ;
          foaf:name ?name ;
          ex:livesIn ex:Madrid .
}
```

- Crear nodos:

```
cypher
Copiar código
CREATE (p:Persona {nombre: "Juan", edad: 30})
CREATE (c:Ciudad {nombre: "Madrid"})
```

- Crear relaciones:

```
cypher
Copiar código
MATCH (p:Persona {nombre: "Juan"}), (c:Ciudad {nombre:
"Madrid"})
CREATE (p)-[:VIVE_EN]->(c)
```

- Encuentra personas que viven en una ciudad:

```
cypher
Copiar código
MATCH (p:Persona)-[:VIVE_EN]->(c:Ciudad {nombre:
"Madrid"})
RETURN p.nombre
```

Modalities of integration KG + RAG

- **Semantic fusion**
  - KG nodes or subgraphs are transformed into rich text (e.g., natural form triples) and used as context for the generative model
  - Example:
    - Convert Cervantes --participated in --> Battle of Lepanto to "Cervantes participated in the Battle of Lepanto in 1571"
- **Hybrid Retrieval**
  - Two channels are combined
    - Vector RAG with embeddings
    - Semantic RAG based on KG
  - The most relevant combined source is selected or weighted.

RAG with and without KG

- Example **- Query**: “What is the relationship between Cervantes and the Ottoman Empire?”
  - **Without KG**
    - Retrieves a fragment about Cervantes and a fragment about the Ottoman Empire.
    - The answer could be vague or disconnected:
      - ‘Cervantes was a Spanish writer and the Ottoman Empire was a historical power’.
  - **With KG**: accurate, verifiable and navigable answer
    - KG includes the triple: Cervantes → fought against → Ottoman Empire through Battle of Lepanto
    - Contextualised answer:
      - ‘Miguel de Cervantes participated as a soldier in the Battle of Lepanto in 1571, where he faced the Ottoman Empire as part of the Holy League’.

USE CASE IN DEVELOPMENT WITH RAG AND KG

Archive of Acción Católica Española

(ACE)

Universidad Pontificia de Salamanca

USE CASE: Digitalization, Publication and Access to ACE

- **What is the Archive of “Acción Católica Española” (ACE)?**
  - Historical archive of the ACE as a documentary source of text in paper format stored in the UPSA library
  - Unique collection of high historical value that offers a deep insight into the role of the Catholic Church in Spain, especially during the 20th century
    - Founded in 1926, the Spanish Catholic Action was a key organization in the religious, social and political life of the country, with a significant impact in various fields, from politics during the Republic and Franco's dictatorship to the democratization of Spanish society.
    - Documents covering the period from the 1930s to the transition to democracy in Spain.
  - Content:
    - Includes documents such as **minutes of meetings, correspondence, reports, circulars** and other records, typed and in some cases also annotated by hand.
  - Interest
    - Preservation of the Spanish historical memory (2nd Republic, civil war, dictatorship, transition) of social and political actors (Catholic workers' movements (Hermandad Obrera Acción Católica - HOAC, Juventud Obrera Católica - JOC) and women's movements (Acción Católica de la Mujer - ACM).

Project: Digitalization, Publication and Access to ACE

- Objective
  - Facilitate the dissemination of and access to the archive through its digitization and ease of consultation of its content both by experts (researchers, historians, archivists) and civil society in general.
  - Promote research on its content
- Proposal:
  - Use of new AI technologies to automate the accessibility and consultation of the relevant information stored in the archive's documents once they have been digitized with quality OCR.
- Challenges:
  - Quality of the documents (old, typed and with handwritten annotations)
  - Lack of prior knowledge and no cataloguing of the content.
  - Huge volume, both to store once digitalized and to process: more than 3000 boxes with an average of 300 documents each with possibly several pages per document.

Digitalization and Access Process

** Scalable process starting with a selection of documents from the ACE archive.

1. Digitalization with OCR of the documents (PDF and TXT)
2. Obtaining document information with IA tools
   - NER (people, organizations, dates and places)
   - Summary
   - Key concepts
3. RAG - Facilitating consultation of documents in natural language (chatbot)
4. Publication in institutional and/or external repository (European Cultural Heritage Cloud), using standard open publication formats for cultural and heritage assets in Europe (CIDOC-CRM, OAI-PMH (Dublin CORE)).

Used Technologies

- OCR digitalization
- LLMs for summaries and NER of documents
- RAG: Chatbot
    - Creation of KG associating data to the document
        - Document + NER, summary, keywords...
    - RAG: Queries with HYBRID retrieval
- IA (LLMs) to obtain metadata in open formats (CIDOC-CRM and Dublin Core ontology).

RAG, KG and ACE archive. Current situation and challenges

- Current status
  - Subset of the collection digitized (e.g. JOC correspondence)
  - Extraction
    - Summarization with LLM (Chatgpt)
    - NER with BERT, FAIR, ...
      - Challenge - improvement: LLM to obtain NER
  - Query
  - RAG with KG+Vectors
    - KG creation (NER association to document)
    - KG for document selection
      - Selection of KG and associated documents, based on NER from prompt queries

Example: Summary and NER from an ACE document

Sube un documento: Examinar... AUPSA_ACE_JN_0062_001_15.pdf

Resume el documento | Clasifica el documento | Obten las palabras clave | Chatbot individual | Chatbot general

Obten los NERs 0,85

Umbral de confianza (0–0.99)

El 29 de mayo de 1964, Santiago Corral Pérez, presidente de la Junta Diocesana de la A.C.Es, se dirige a sus amigos para recordarles sobre un proyecto de difusión del "Día de la A.C." que se llevará a cabo a través de la prensa y la televisión. Menciona que su Junta Diocesana debe contribuir con una cantidad específica para cubrir los gastos de esta propaganda, pero hasta esa fecha no ha recibido el dinero correspondiente. Solicita encarecidamente el envío inmediato de los fondos para poder hacer frente al pago total del proyecto, ya que su economía no permite adelantar dicha suma. Finaliza expresando su afecto y esperando verlos en las próximas Jornadas Nacionales.

Sube un documento: Examinar... AUPSA_ACE_JN_0062_001_15.pdf

Resume el documento | Clasifica el documento | Obten las palabras clave | Chatbot individual | Chatbot general

Obten los NERs 0,85

Umbral de confianza (0–0.99)

Entidad	Tipo Flair	Confianza Flair	Tipo BERT	Confianza BERT	Tipo Roberta	Confianza Roberta
Santiago Corral Pérez	PER	1.00	PER	0.97	-	-
Madrid	LOC	1.00	-	-	-	-
Junta Diocesana	ORG	1.00	ORG	0.83	-	-
Junta Nacional	ORG	1.00	ORG	0.82	-	-
Junta Diocesana de la A.C.Es	ORG	1.00	-	-	-	-
Junta	ORG	1.00	-	-	-	-
Tesorería	MISC	0.95	ORG	0.59	-	-

Example: Clasification and Key concepts from an ACE document

Sube un documento: Examinar... AUPSA_ACE_JN_0062_001_15.pdf

Resume el documento | Clasifica el documento | Obten las palabras clave | Chatbot individual | Chatbot general

Obten los NERs 0,85

Umbral de confianza (0–0.99)

El tipo de documento es: CARTAS (confianza: 0.73)

Sube un documento: Examinar... AUPSA_ACE_JN_0062_001_15.pdf

Resume el documento | Clasifica el documento | Obten las palabras clave | Chatbot individual | Chatbot general

Obten los NERs 0,85

Umbral de confianza (0–0.99)

Las palabras clave de este texto son:

* Junta Diocesana
* A.C. (Asociación Católica)
* Propaganda
* Día de la A.C.
* Tesorería
* Junta Nacional
* Jornadas Nacionales

Estas palabras clave reflejan el contenido del texto, que se centra en la comunicación entre la Junta Nacional y la Junta Diocesana sobre un proyecto de propaganda y el pago correspondiente.

RAG and KG in ACE. Steps of ACE queries

1. **Input:** A question-query (prompt) is received from the user from the page
2. **Extraction of information from the archive** (3 methods) (KGs + vectors):
    - Obtaining NERs from the question and searching for these NERs in Neo4j (KG database)
        - spaCy is used to extract named entities (NERs) from the question. These entities are queried in the KGs database, Neo4j, to identify documents that mention them.
    - Automatic query generation in Cypher (Neo4j language) using generative AI to search Neo4jA
        - Cypher query is automatically generated using a generative AI (GPT) model, based on the user's question. This query is executed in Neo4j to retrieve the most relevant documents.
    - Semantic vector search in ChromaDB
        - The question is transformed into a vector and ChromaDB is searched for semantic similarity (vector), returning texts similar to the question.
3. **Answer generation:**
    - All methods load texts from the corresponding documents that are used by a generative AI model to generate an answer.
    - A **combined answer is generated**, which uses all the texts identified by the 3 methods to provide the most complete view possible.

Example: RAG - ACE

Sube un documento: Examinar... No se ha seleccionado ningún archivo.

Resume el documento | Clasifica el documento | Obten las palabras clave | Chatbot individual | Chatbot general

Obten los NERs | 0,85

Umbral de confianza (0–0.99)

Chatbot general activado. Escribe tu pregunta abajo.

Tú: Me puedes decir quién fue Santiago Corral Pérez?

SPACY:
[Documentos usados: ['AUPSA_ACE_JN_0062_001_9.pdf', 'AUPSA_ACE_JN_0062_001_10.pdf', 'AUPSA_ACE_JN_0062_001_5.pdf']]
Santiago Corral Pérez fue una figura relacionada con la Acción Católica Española en la década de 1960. Su nombre aparece en varias cartas dirigidas a obispos, donde se menciona su papel en la Junta Nacional de la Acción Católica. En estas correspondencias, Corral Pérez se presenta como un interlocutor que se encarga de comunicar orientaciones y actividades de la Junta, así como de recibir sugerencias para mejorar la labor apostólica encomendada a las Juntas Diocesanas. Aunque los documentos no ofrecen mucha información personal sobre él, su implicación indica que tuvo un papel activo en la organización y promoción de la acción católica en ese período.

CYPHER:
[Documentos usados: ['AUPSA_ACE_JN_0062_001_5.pdf', 'AUPSA_ACE_JN_0062_001_10.pdf', 'AUPSA_ACE_JN_0062_001_9.pdf']]
[Consulta Cypher usada: MATCH (d:Document)-[:MENTIONS]->(p:Person {name: "Santiago Corral Pérez"})
RETURN d.name, d.pdf_path]
Santiago Corral Pérez fue un destacado miembro de la Junta Nacional de la Acción Católica Española en la década de 1960. En los documentos se le menciona como presidente de esta organización, donde desempeñó un papel importante en la comunicación de actividades y la promoción de la acción apostólica a través de circulares dirigidas a las Juntas Diocesanas. Su trabajo se centró en la organización y la coordinación de los esfuerzos de la Acción Católica en España durante ese período, y se muestra dispuesto a recibir sugerencias y observaciones de otros miembros del clero.

Escribe tu pregunta... | Enviar

Chatbot general

Católica en España durante ese período, y se muestra dispuesto a recibir sugerencias y observaciones de otros miembros del clero.

CHROMA:
[Documentos usados: ['AUPSA_ACE_JN_0062_001_16.pdf', 'AUPSA_ACE_JN_0062_001_15.pdf', 'AUPSA_ACE_JN_0062_001_13.pdf', 'AUPSA_ACE_JN_0062_001_18.pdf', 'AUPSA_ACE_JN_0062_001_5.pdf']]
Santiago Corral Pérez fue el presidente de la Junta Nacional de la A.C.E. (Asociación Católico-Española). En los documentos proporcionados, se le menciona en varias ocasiones, donde también se refleja su cordialidad y su dedicación a la organización. Además, parece estar involucrado en la comunicación y la promoción de publicaciones relacionadas con asuntos eclesiásticos.

COMBINADA:
[Documentos usados: ['AUPSA_ACE_JN_0062_001_16.pdf', 'AUPSA_ACE_JN_0062_001_15.pdf', 'AUPSA_ACE_JN_0062_001_10.pdf', 'AUPSA_ACE_JN_0062_001_13.pdf', 'AUPSA_ACE_JN_0062_001_9.pdf', 'AUPSA_ACE_JN_0062_001_18.pdf', 'AUPSA_ACE_JN_0062_001_5.pdf']]
Santiago Corral Pérez fue un miembro activo en la organización de asociaciones relacionadas con la actividad apostólica, destacándose en su papel dentro de la Junta Nacional de la A.C. (Asociaciones de Cabezas de Familia) en España durante la década de 1960. A lo largo de los documentos, se le menciona en correspondencia con diferentes líderes y figuras de la iglesia, donde se aborda la organización de proyectos y actividades, así como la coordinación de reuniones y la difusión de iniciativas. Su compromiso se refleja en la gestión de proyectos de propaganda y en el intercambio de información destinada a mejorar la labor de estas asociaciones.

Escribe tu pregunta... | Enviar

ACE and CIDOC-CRM

Entities

Class	Brief Description
E1 CRM Entity	Root class (everything is a CRM entity)
E21 Person	Individual person
E74 Group	Groups, collectives, institutions
E53 Place	Geographical location (country, city, building...)
E52 Time-Span	Time interval
E5 Event	Cultural, historical, biographical event...
E7 Activity	Specific human activity
E31 Document	Document or textual content
E73 Information Object	Symbolic information (documents, texts, images...)
E22 Man-Made Object	Physical object created by humans

[E31 Document]
└─ P70 documents ──▶ [E21 Person]
└─ P70 documents ──▶ [E53 Place]
└─ P70 documents ──▶ [E5 Event]
└─ P3 has note ───────▶ "Resumen o contenido"
└─ P67 refers to ─────▶ [E74 Group], [E52 Time-Span], etc.

Properties or Relations

Property	Meaning
P14 carried out by	An event was carried out by a person or group
P4 has time-span	Temporal association of an event or activity
P7 took place at	Geographical location of an event
P70 documents	A document contains information about an entity
P102 has title	Title of an information object
P3 has note	Comments, descriptions, summaries
P67 refers to	Information refers to an entity (useful for NER)

RDF with CRM-CIDOC (ACE example)

```
:doc_001 a crm:E31_Document ;
    crm:P102_has_title "Carta de José María a Juan" ;
    crm:P3_has_note "Carta sobre la organización de una peregrinación en 1954."
;
    crm:P67_refers_to :jose_maria, :salamanca, :año_1954 .

:jose_maria a crm:E21_Person ;
    rdfs:label "José María" .

:salamanca a crm:E53_Place ;
    rdfs:label "Salamanca" .

:año_1954 a crm:E52_Time-Span ;
    rdfs:label "1954" .
```