
Propuesta de una aplicación web para la consulta y visualización de datos semánticos RDF

Proposal for a web application for querying and visualizing RDF semantic data

Ana LÓPEZ-MORALES (1), Juan-Antonio PASTOR-SÁNCHEZ (2), José A. RUIPÉREZ-VALIENTE (3)

(1) Universidad de Murcia, ana.lopezm3@um.es (2) Departamento de Información y Documentación, Universidad de Murcia, pastor@um.es (3) Departamento de Informática y Sistemas, Universidad de Murcia, jruiperez@um.es

Resumen

Se presenta el desarrollo e implementación de una aplicación web para la consulta y visualización de datos semánticos RDF, junto con su evaluación. El alcance cubre todo el ciclo: definición de requisitos (funcionales y no funcionales), análisis y diseño (casos de uso, arquitectura multicapa y comparativa tecnológica), implementación del prototipo y evaluación con usuarios. Los resultados muestran una solución que integra un dashboard con métricas y visualizaciones interactivas, un sistema de búsqueda de instancias mediante filtrado, un espacio de consulta SPARQL y exportaciones en CSV/TSV/JSON y Turtle. En la evaluación (n=16) se alcanzó el 100 % de efectividad en tareas y una usabilidad percibida excelente (SUS=81,41). Se concluye que el modelo reduce la carga cognitiva, mantiene la trazabilidad entre criterios y resultados y equilibra accesibilidad para perfiles no expertos con profundidad para usuarios avanzados. En coherencia con los hallazgos, se recomiendan mejoras y se identifican futuras acciones como la incorporación de ayudas contextuales y documentación, visualización de vocabularios/ontologías, gestión de datasets desde la interfaz, o desarrollo de nuevas visualizaciones.

Palabras clave: Web semántica. Datos enlazados. RDF. Desarrollo web. Visualización de datos semánticos. Exploración de datasets.

1. Introducción

La expansión de la World Wide Web ha multiplicado la creación y circulación de contenidos (Castells, 2005), pero también ha fragmentado el ecosistema informacional y complejizado la recuperación eficaz de información (Méndez Rodríguez, 1999). Ante esta tensión, la web semántica, concebida por Tim Berners-Lee, propone una evolución hacia datos comprensibles por máquinas mediante codificación estructurada, favoreciendo la interoperabilidad y el intercambio sin pérdida de contexto (Angles, 2015). Sobre este fundamento emergen los conjuntos de datos semánticos o datasets, cuya definición alude a colecciones estructuradas y relacionables entre

Abstract

This paper presents the development and implementation of a web application for querying and visualizing RDF semantic data, together with its evaluation. The scope covers the full cycle: requirements definition (functional and non-functional), analysis and design (use cases, multilayer architecture, and technology comparison), prototype implementation, and user evaluation. The results show a solution that integrates a dashboard with metrics and interactive visualizations, an instance search system via filtering, a SPARQL query area, and exports to CSV/TSV/JSON and Turtle. In the evaluation (n = 16), task effectiveness reached 100% and perceived usability was excellent (SUS = 81.41). We conclude that the model reduces cognitive load, maintains traceability between criteria and results, and balances accessibility for non-expert users with depth for advanced users. In line with the findings, we recommend enhancements and identify future actions such as adding contextual help and documentation, visualizing vocabularies/ontologies, enabling dataset management from the interface, and developing new visualizations.

Keywords: Semantic Web. Linked Data. RDF. Web development. Semantic data visualization. Dataset exploration.

sí, creadas con tecnologías semánticas para habilitar su procesamiento automático, con claras ventajas en búsqueda y clasificación (The Information Lab, 2023; Sota Martínez et al., 2016). Su representación se apoya en RDF, extendido por RDFS y, cuando se requiere mayor expresividad, por OWL para la definición ontológica. La consulta se articula con SPARQL como estándar. (RDF 1.1 Primer, 2014).

En paralelo, el auge del Big Data ha puesto el foco en volumen, velocidad y variedad de los datos, dimensiones que desbordan las herramientas tradicionales y crea la necesidad de nuevos modelos usables que soporten este tipo de datos. Asimismo, ante estos grandes conjuntos de datos se subraya el valor de los estándares semánticos

para tratar la heterogeneidad e integrar sistemas (Hitzler & Janowicz, 2013). La visualización se consolida así como instrumento clave para sintetizar, explorar y extraer conocimiento, facilitando razonamiento y descubrimiento de patrones (Michalos et al., 2012, (Desimoni y Po, 2020), especialmente en datos enlazados donde la exploración de relaciones es central (Ivanova et al., 2019). No obstante, la consulta semántica sigue siendo exigente: SPARQL requiere pericia técnica y, pese a avances, la consulta en lenguaje natural aún no es un estándar operativo (Wohlgeannt et al., 2019; Boumechaal & Boufaïda, 2023). Aun así, su capacidad para integrar dominios y ofrecer acceso cercano al tiempo real aporta un valor diferencial (Hartig, 2013).

Asimismo, para alcanzar una comprensión operativa de grandes grafos RDF, la arquitectura de la información resulta determinante: estructura, organización, etiquetado y diseño de procesos de interacción que sostienen la localización y el sentido de los datos, para que el usuario pueda comprender en todo momento lo que visualiza e interactuar adecuadamente con la información (Pérez-Montoro, 2010; Resmini & Rosati, 2011).

De esta forma, la integración de visualización, consulta semántica y una arquitectura de la información sólida es condición necesaria para que los datos semánticos, cada vez más estratégicos en lo social y empresarial, se presenten de forma clara, explotable y orientada a la toma de decisiones.

Este trabajo se centra en proponer un modelo para la consulta y visualización de datos semánticos RDF de gran tamaño, partiendo de que una representación visual adecuada favorece una comprensión profunda y holística de la información, potencia su reutilización y facilita la explotación de su valor. Con ello, se persigue mejorar claridad interpretativa para la toma de decisiones mediante el acceso y análisis de grandes volúmenes de datos estructurados, apoyar la investigación ofreciendo fuentes accesibles y explotables, y contribuir a combatir la desinformación al presentar al público datos comprensibles a través de herramientas que favorezcan su interpretación.

En consecuencia, se formulan una serie de objetivos específicos. En primer lugar, analizar las herramientas y enfoques existentes de arquitectura de información para la consulta y visualización de datos RDF, identificando sus aportaciones y límites. A continuación, determinar los requisitos del modelo, tanto funcionales como no funcionales. Seguidamente, se busca diseñar el modelo, definiendo su arquitectura de la información y el flujo de interacción desde la visión global hasta el de-

talle. Tras esto, se desarrolla el prototipo que materializa el modelo y permite su contrastación; y, finalmente, se realiza una evaluación de la usabilidad y la efectividad del modelo.

2. Metodología

El desarrollo del proyecto se abordó desde una perspectiva de ingeniería del software con carácter iterativo, integrando en cada fase los resultados de la anterior y validando las decisiones mediante pruebas funcionales y revisiones cruzadas. El objetivo fue asegurar coherencia entre la concepción teórica, la materialización técnica y la experiencia de uso.

El proceso completo se organizó en cinco fases principales:

- *Fase 1. Especificación de requisitos:* en primer lugar, se llevó a cabo un análisis comparativo de herramientas existentes para la consulta y visualización de datos RDF. Este estudio permitió identificar patrones comunes, limitaciones y vacíos de diseño. A partir de ese diagnóstico se formularon los requisitos funcionales y no funcionales del sistema, priorizando la usabilidad, la escalabilidad y la claridad de la interfaz.
- *Fase 2. Diseño de la Arquitectura de la Información:* con los requisitos definidos, se diseñó el modelo general del sistema, definiendo un conjunto de casos de uso que representan los flujos principales de interacción desde la perspectiva del usuario final; determinando la arquitectura y tecnologías que se iban a utilizar durante la implementación; y elaborando los primeros mockups para validar la arquitectura de la información y la distribución de componentes antes de su desarrollo. Durante estas etapas, se integraron de forma explícita los principios de arquitectura de la información: jerarquía visual coherente, rótulos comprensibles, consistencia entre vistas, y conexión entre clases, instancias y propiedades. Se definieron las rutas de navegación, los menús persistentes y la estructura semántica de las páginas, garantizando que la consulta de los datos RDF se mantuviera contextualizada y comprensible. Esta fase sirvió como puente entre la teoría del diseño informacional y la implementación práctica.
- *Fase 3. Implementación del prototipo:* el desarrollo se realizó con metodología incremental, liberando versiones funcionales en ciclos cortos que permitían validar cada módulo antes de incorporar nuevas funciones. La pila tecnológica seleccionada —Django para el desarro-

llo web, Plotly.js para la generación de visualizaciones interactivas y Apache Jena Fuseki como triplestore RDF— permitió integrar el frontend, la lógica de aplicación y el backend de manera coherente. Durante esta fase se implementaron la gestión de datasets, el dashboard con métricas globales, el sistema de búsqueda mediante filtrado, la ficha de instancias y la consulta SPARQL avanzada. Cada módulo fue probado de forma independiente para verificar su funcionamiento y su interoperabilidad con el resto del sistema.

- *Fase 4. Evaluación y mejora iterativa:* la última fase consistió en la evaluación con usuarios, que permitió valorar la utilidad, la usabilidad y la eficacia del modelo. Los resultados se analizaron cuantitativa y cualitativamente, identificando mejoras que fueron incorporadas en versiones posteriores. Entre ellas, destacan la retroalimentación visual en acciones críticas, la confirmación tras la carga de datasets y la clarificación terminológica en los paneles de búsqueda.

El conjunto del proceso se caracterizó por su orientación práctica, analizando cada decisión para garantizar que la aplicación final respondiera a necesidades detectadas y fuera técnicamente sólida, escalable y comprensible.

3. Modelo de Arquitectura de la Información para la consulta y visualización de datasets RDF

La arquitectura de la información constituye el eje estructural del modelo propuesto. Su función no es únicamente organizar contenidos, sino transformar la complejidad estructural de los grafos RDF en una experiencia de consulta comprensible, progresiva y orientada al usuario. El objetivo no es mostrar tripletas, sino facilitar su interpretación manteniendo el contexto informacional en todo momento.

El modelo se fundamenta en un análisis comparativo de herramientas existentes para la consulta y visualización de datos RDF, tanto académicas como de uso extendido en entornos de datos abiertos. Este análisis permitió identificar los principales ejes de interacción empleados, así como limitaciones recurrentes desde el punto de vista de la usabilidad, la escalabilidad y la comprensión global del dataset.

Del estudio de las aplicaciones analizadas se identificaron cinco ejes de interacción predominantes. Estos ejes no son excluyentes y, en algunos casos, una misma herramienta combina varios de ellos.

- *Visualización basada en grafos:* Herramientas como LodLive priorizan la representación visual del grafo RDF. Cada nodo representa una instancia y las relaciones se expanden dinámicamente mediante consultas SPARQL. Este enfoque facilita la exploración relacional puntual y resulta intuitivo para comprender conexiones locales. Sin embargo, cuando el número de entidades o relaciones aumenta, el grafo se densifica rápidamente. En datasets medianos o grandes, la visualización pierde legibilidad y deja de ofrecer una visión estructural del conjunto.
- *Visualización ontológica:* Aplicaciones como WebVOWL u OntoGraf centran la interacción en la ontología subyacente. Representan clases, propiedades y jerarquías conceptuales. Este enfoque resulta útil para comprender el modelo semántico del dominio, pero está orientado principalmente a usuarios expertos. Además, no facilita la exploración directa de los datos instanciados ni la consulta orientada a tareas concretas.
- *Navegación textual y por fichas:* Herramientas como LodView y Pubby adoptan un enfoque basado en fichas de instancia. Presentan descripciones RDF legibles, enlaces a entidades relacionadas y resultados estructurados en tablas. Su principal fortaleza es la claridad y la trazabilidad entre recursos. No obstante, la navegación es fundamentalmente lineal. El usuario accede a una instancia cada vez, sin mecanismos de síntesis, filtrado guiado o visión global del dataset.
- *Búsqueda guiada y filtrado semántico:* Este eje se basa en la interacción mediante formularios, filtros y selectores semánticos, que permiten construir consultas sin necesidad de escribir SPARQL. Algunas funcionalidades de Rhizomer se aproximan a este enfoque al combinar navegación semántica y visualizaciones básicas. Sin embargo, en las herramientas existentes el filtrado suele ser rígido, poco explicativo o escasamente adaptable a la estructura concreta del dataset cargado.
- *Consulta técnica y agregación:* Plataformas como LodCentral priorizan la consulta directa mediante SPARQL y la agregación de múltiples fuentes RDF. Este enfoque ofrece gran potencia y escalabilidad, pero presenta una elevada barrera de entrada. Requiere conocimientos técnicos previos y desplaza la carga cognitiva al usuario.

La Tabla I sintetiza la clasificación de las herramientas analizadas según los ejes de interacción

identificados, junto con sus principales fortalezas y limitaciones.

A partir de este análisis comparativo, se identificaron varios patrones comunes:

- La mayoría de las aplicaciones enfatiza uno de los tres ejes —consulta, visualización o navegación—, pero pocas integran los tres de forma coherente.

- Las herramientas centradas en SPARQL priorizan la flexibilidad, pero sacrifican accesibilidad.
- Las que apuestan por la visualización gráfica pierden eficacia en datasets grandes.
- Y las basadas en navegación por fichas carecen de una capa de contextualización que ayude a comprender la “forma” del grafo.

Herramienta	Enfoque	Fortalezas principales	Limitaciones
LodLive	Visualización basada en grafos; Consulta técnica	Exploración relacional intuitiva	Saturación visual en datasets grandes
WebVOWL/OntoGraf	Visualización ontológica	Comprensión del modelo conceptual	Poco orientada a datos instanciados
LodView	Navegación textual por fichas	Claridad y trazabilidad	Navegación lineal, sin síntesis
Pubby	Navegación textual por fichas	Publicación legible de URIs	Sin filtrado guiado ni visualización
Rhizomer	Navegación semántica; Búsqueda guiada parcial	Enfoque híbrido inicial	Interfaz poco adaptable al dataset
LodCentral	Consulta técnica; Agregación	Potencia y escalabilidad	Alta barrera de entrada para no expertos

Tabla 1. Comparativa de herramientas de consulta y visualización RDF según ejes de interacción

El análisis comparativo permitió identificar patrones comunes en las herramientas estudiadas:

- Predominio de un único eje de interacción, sin integración coherente de los demás.
- Alta dependencia del conocimiento técnico cuando se prioriza la consulta SPARQL.
- Pérdida de eficacia de las visualizaciones basadas en grafos en datasets grandes.
- Navegación por fichas sin mecanismos de contextualización global.
- Filtrado semántico poco flexible o escasamente adaptado al dataset activo.

Además, el análisis puso de manifiesto un vacío funcional relevante: ninguna de las herramientas analizadas incorpora de forma explícita un eje de exploración estadística y agregada del dataset. No se ofrecen métricas globales, distribuciones ni visualizaciones estructurales que actúen como punto de entrada cognitivo y faciliten una comprensión inicial del conjunto de datos.

Estas observaciones motivaron el desarrollo de un modelo equilibrado, en el que la arquitectura de la información articula los tres componentes (consulta, visualización y navegación) y los traduce a una estructura cognitiva clara y estable. El modelo propuesto se construye sobre tres ni-

veles interdependientes: macroestructural, mesoestructural y microestructural, que organizan la interacción del usuario con los datos RDF.

1. *Nivel macroestructural.* Define la organización global del sistema y los principales modos de acceso a la información. Incluye las rutas de navegación, el menú principal y los puntos de entrada: el dashboard con métricas y gráficos, la búsqueda guiada por filtrado, la consulta SPARQL y la exploración por instancias. Este nivel garantiza la coherencia general y permite recorrer el conjunto de datos de forma progresiva, desde la visión global hasta el detalle. Su objetivo es mantener la orientación del usuario y asegurar que cada acción conduzca a un resultado comprensible.
2. *Nivel mesoestructural.* Establece la jerarquía interna y las relaciones entre las entidades. Aquí se materializa la adaptación del modelo RDF a la experiencia informacional: las clases actúan como categorías base; las propiedades, como criterios de búsqueda; y las instancias, como nodos de exploración interconectados. El sistema traduce la lógica del grafo a una interfaz ordenada y predecible. Los filtros se adaptan al contexto, mostrando únicamente las propiedades relevantes para cada clase, y las fichas de instancia enlazan dinámicamente con otras entidades o recursos externos. Este nivel reproduce de forma visual

los principios de la web semántica —interconexión y trazabilidad—, pero desde la perspectiva del usuario.

3. *Nivel microestructural*. Abarca la presentación visual y la rotulación de los elementos. Se cuida el lenguaje, sustituyendo términos técnicos por etiquetas comprensibles (“Búsqueda de instancias” en lugar de “Filtros”), y se utilizan colores, iconos y mensajes de estado para ofrecer retroalimentación inmediata. Cada vista incluye ayudas contextuales y confirmaciones visuales, lo que refuerza la confianza del usuario y mejora la percepción de control. El diseño se apoya en principios de usabilidad clásica: consistencia, visibilidad del estado del sistema y prevención de errores.

El análisis comparativo de herramientas permitió extraer principios prácticos que se incorporaron directamente al modelo:

- *Consistencia y persistencia de navegación*. Los menús y rutas se mantienen estables entre vistas, evitando saltos o rupturas de contexto. Asimismo, se busca la optimización de dichas rutas, evitando crear niveles de navegación demasiado profundos que confundan al usuario.
- *Jerarquía informacional*. Cada nivel —dataset, clase, instancia, propiedad— se presenta de forma progresiva, preservando la relación entre los elementos.
- *Contextualización y explicación*. A diferencia de interfaces opacas, el modelo intenta incorporar rótulos explicativos para que el usuario sepa en todo momento lo que está viendo.
- *Adaptabilidad*. Los formularios y desplegados se ajustan dinámicamente a la estructura real del dataset activo, garantizando relevancia y reduciendo errores.
- *Escalabilidad*. El uso de un archivo JSON intermedio permite manejar conjuntos voluminosos sin degradar el rendimiento, conservando la fluidez de la experiencia.

El modelo propuesto no pretende sustituir los enfoques existentes, sino integrar sus aportaciones en una arquitectura de la información coherente, orientada a facilitar la consulta y visualización de datos RDF de gran tamaño y a reducir la carga cognitiva asociada a su exploración.

4. Resultados

El desarrollo del proyecto generó resultados concretos en cada una de las fases previstas. A continuación, se detallan los principales logros, orga-

nizados en cuatro bloques: especificación de requisitos, diseño de la herramienta, implementación y evaluación.

4.1. Especificación de requisitos

El análisis comparativo de herramientas permitió identificar las carencias más frecuentes en los entornos de consulta y visualización de datos RDF: interfaces poco intuitivas, escasa integración entre visualización y búsqueda, y alta dependencia del conocimiento técnico.

A partir de este diagnóstico se formularon *requisitos funcionales (RF)* y *no funcionales (RNF)* que guiaron todo el desarrollo posterior. Estos requisitos fueron clasificados y numerados según el esquema [TIPO]-[CATEGORÍA]-[NÚMERO].

Los RF definen lo que el sistema debe hacer para cumplir sus objetivos. Describen las acciones, operaciones o procesos que la aplicación ejecuta y que resultan visibles para el usuario. En este proyecto, los requisitos funcionales abarcan todas las capacidades esenciales relacionadas con la visualización, la navegación, la interacción y la consulta de datos RDF.

Por otro lado, los RNF establecen las condiciones bajo las cuales el sistema debe operar para asegurar su calidad, eficiencia y sostenibilidad. No describen funciones específicas, sino propiedades globales que afectan al rendimiento, la usabilidad, la compatibilidad o la escalabilidad de la aplicación.

De esta forma, los RF se agruparon en cuatro categorías: visualización, navegación, interacción y consulta de datos.

4.1.1. Visualización (RF-VIS).

- *RF-VIS-01. Dashboard con métricas globales*. Debía mostrar indicadores clave (número de tripletas, clases, propiedades e instancias) y gráficos que representaran la estructura general del conjunto.
- *RF-VIS-02. Visualizaciones interactivas*. Gráficos dinámicos que permitieran interpretar distribuciones y relaciones entre entidades.
- *RF-VIS-03. Paginación y control de densidad*. Los resultados debían mostrarse de forma progresiva, evitando la sobrecarga informativa.
- *RF-VIS-04. Coherencia visual*. La interfaz debía mantener una jerarquía estable, con secciones claramente identificadas.

4.1.2. Navegación (RF-NAV)

- *RF-NAV-01. Menú persistente.* Acceso permanente a las secciones principales: inicio, estadísticas, búsqueda, consulta SPARQL y ayuda.
- *RF-NAV-02. Flujo continuo.* Transiciones suaves entre dashboard, filtrado y fichas de instancia, sin pérdida de contexto.
- *RF-NAV-03. Retorno controlado.* Posibilidad de regresar en cualquier momento al punto de partida, preservando el dataset activo y los filtros aplicados.

4.1.3. Interacción (RF-INT)

- *RF-INT-01. Filtrado dinámico.* El usuario debía poder combinar varios criterios (clase base, propiedad, valor, operador lógico AND/OR).
- *RF-INT-02. Eliminación y limpieza de filtros.* Control individual o global sobre los filtros activos.
- *RF-INT-03. Gestión de datasets.* Subida, actualización o eliminación de conjuntos RDF desde la interfaz.
- *RF-INT-04. Retroalimentación visual.* Confirmaciones y mensajes de progreso tras acciones críticas como la carga o regeneración de datasets.

4.1.4. Consulta (RF-CON)

- *RF-CON-01. Editor SPARQL integrado.* Permitir la ejecución directa de consultas sobre el dataset activo.
- *RF-CON-02. Resultados descargables.* Exportación de los datos en formatos estándar (CSV, TSV, JSON, Turtle).
- *RF-CON-03. Subgrafo RDF de filtrado.* Generación de un conjunto reducido a partir de los resultados de búsqueda.
- *RF-CON-04. Mensajes de estado.* Información clara ante consultas vacías o errores de conexión.

Por su parte, Los RNF se definieron en torno a tres categorías: rendimiento, usabilidad y arquitectura y compatibilidad:

4.1.5. Rendimiento (RNF-REN).

- Tiempos de respuesta razonables.
- Carga progresiva mediante paginación.

- Uso de archivos intermedios (JSON) para acelerar las operaciones frecuentes.

4.1.6. Usabilidad (RNF-USA)

- Interfaz clara y coherente.
- Pantallas de carga con información visual.
- Etiquetas comprensibles para usuarios no expertos.

4.1.7. Ayudas y mensajes de orientación contextual

- Arquitectura y compatibilidad (RNF-ARC).
- Adaptabilidad a distintos conjuntos RDF.
- Escalabilidad para conjuntos de gran tamaño.
- Uso de tecnologías abiertas y compatibles con estándares semánticos.

El catálogo de requisitos se utilizó como una hoja de ruta para el desarrollo del proyecto, asegurando la trazabilidad entre objetivos, decisiones de diseño y resultados finales.

4.2. Diseño de la herramienta

La fase de análisis y diseño se divide, a su vez, en 3 subetapas —definición de los casos de uso, comparativa y selección de la arquitectura tecnológica, y elaboración de mockups— en las que se aplican de manera progresiva los principios de la arquitectura de la información para conceptualizar la herramienta.

4.2.1. Casos de uso

Para describir el comportamiento del sistema desde la perspectiva del usuario final se definieron una serie de casos de uso que cubren el ciclo completo de uso de la aplicación: visión global del conjunto, exploración por clases e instancias, consulta y exportación, y gestión básica de datasets. De este modo, se pretende representar de forma estructurada las acciones que los usuarios podían realizar dentro de la aplicación y las respuestas que el sistema debía ofrecer en cada situación. Su finalidad fue garantizar que el desarrollo respondiera a necesidades reales de interacción con los datos RDF y que las funciones implementadas cubrieran todo el ciclo de uso previsto. Para su elaboración se partió del conjunto de requisitos funcionales, analizando cada uno y agrupándolos según las tareas que implicaban una interacción directa con el usuario. Posteriormente, se diseñaron los flujos de trabajo asociados a cada tarea, definiendo desencadenantes, condiciones iniciales, secuencias de pasos, excepciones y resultados esperados.

El proceso de identificación de los casos de uso se apoyó en la observación de patrones comunes en herramientas similares y en la experiencia práctica durante las primeras pruebas del prototipo. Se identificaron las acciones más frecuentes que debía realizar un usuario no experto para explorar un dataset RDF: acceder al dashboard, aplicar filtros, inspeccionar instancias, ejecutar consultas SPARQL o exportar resultados. Cada una de estas acciones se tradujo en un caso de uso independiente, documentado con un título, un objetivo y un flujo principal de interacción. También se añadieron flujos alternativos para contemplar posibles errores o situaciones no previstas, como la ausencia de resultados o la carga de archivos no válidos.

Los casos de uso resultaron fundamentales para planificar el desarrollo y la validación de la aplicación. Permitieron comprobar que cada módulo respondía a una necesidad funcional concreta y que la interfaz mantenía coherencia entre vistas. Además, sirvieron como base para las pruebas con usuarios, ya que las tareas definidas en la evaluación se correspondieron directamente con los casos de uso. Se identificaron ocho casos de uso:

- *CU-01. Visualización del dashboard del dataset.* Proporciona una panorámica del conjunto de datos mediante métricas y gráficas. El usuario selecciona un dataset activo y accede a Estadísticas; si no hay filtros aplicados, el sistema presenta automáticamente el panel de indicadores y las visualizaciones asociadas.
- *CU-02. Visualización de clases.* Permite examinar en detalle una clase concreta (instancias y propiedades asociadas). Tras seleccionar el dataset y acceder a Estadísticas, el usuario elige una clase base y el sistema muestra el listado de instancias y un desplegable de propiedades restringido a dicha clase, incluyendo su frecuencia de uso. Si no existen datos, se muestra un mensaje informativo.
- *CU-03. Filtrado.* Habilita la construcción de consultas guiadas mediante un formulario con parámetros (sujeto, propiedad, objeto, clase base, clase adicional) y combinación AND/OR. Al ejecutar la búsqueda, el sistema actualiza el listado de instancias. Se contemplan estados de sin resultados, eliminación individual de filtros y limpieza total del criterio aplicado.
- *CU-04. Visualización de instancias.* Facilita la inspección de una entidad concreta, sus atributos y valores. Desde el listado filtrado, el usuario abre la ficha de instancia y puede navegar a entidades relacionadas. Si la instancia carece de información, se informa explícitamente.
- *CU-05. Consulta de datos semánticos.* Ofrece un espacio de consulta avanzada donde introducir SPARQL y obtener resultados en tabla paginada. En ausencia de respuestas, el sistema muestra mensajes de estado que orientan sobre la situación.
- *CU-06. Exportar datos consultados.* Permite descargar los resultados de una consulta formal en formatos estándar (CSV, JSON, TSV). Tras visualizar la tabla, el usuario selecciona el formato y el sistema genera y entrega el archivo; los errores de exportación se comunican de forma clara.
- *CU-07. Exportación de resultados filtrados.* Genera un subgrafo RDF a partir del resultado del filtrado, descargable en Turtle (.ttl). El botón de exportación solo aparece cuando hay resultados; en caso contrario o ante errores en la generación, se informa al usuario.
- *CU-08. Carga de nuevos datasets.* Cubre la ingesta de conjuntos RDF desde la interfaz. El flujo incluye selección del archivo, confirmación, almacenamiento en el repositorio de datos, generación del resumen estructurado y retorno a la página principal con el nuevo dataset disponible. Los formatos inválidos o fallos de conexión desencadenan mensajes explicativos.

En todos los casos de uso se mantuvo una lógica común basada en la progresión y la continuidad de la experiencia. Cada acción fue concebida como un paso dentro de un mismo recorrido informativo. Esta coherencia evita rupturas en la navegación y permite que el usuario mantenga siempre el sentido de lo que está haciendo. Los casos de uso, además, comparten una misma estructura de respuesta del sistema: mostrar información comprensible, ofrecer confirmaciones visuales y permitir retomar el flujo anterior sin perder el contexto.

4.2.2. Arquitectura tecnológica de la aplicación

La aplicación adopta una arquitectura multicapa con separación estricta de responsabilidades en presentación, aplicación y datos. Este enfoque alinea las decisiones de diseño con los requisitos funcionales y no funcionales previamente definidos y facilita la evolución del sistema sin dependencias innecesarias.

En la capa de presentación (frontend) se concentra la experiencia de usuario y la arquitectura de

la información: es la encargada de mostrar los contenidos en el formato más adecuado y de capturar las interacciones del usuario.

Por su parte, la capa de aplicación (backend) implementa la lógica del sistema: procesa las interacciones detectadas y genera las respuestas oportunas, ejecuta las consultas necesarias sobre la base de datos, realiza el procesamiento y cálculo de la información y genera recursos destinados a optimizar el comportamiento del sistema.

La capa de datos almacena los conjuntos RDF y gestiona las consultas remitidas desde la capa de aplicación.

De esta forma, el flujo sistémico es directo: cuando el usuario interactúa desde la capa de presentación, la petición se envía al backend, que interpreta la entrada, ejecuta la consulta sobre la capa de datos y transforma la respuesta para su presentación legible en el frontend.

Definida la estructura de la solución, se realizó una comparativa sistemática por capas para seleccionar las tecnologías concretas, basada en criterios homogéneos como compatibilidad con RDF/SPARQL, rendimiento, escalabilidad, documentación, licenciamiento, facilidad de uso y aprendizaje, lenguaje y adecuación al proyecto. La evaluación empleó un esquema de puntuación que asigna valores según el grado de cumplimiento (muy alto/alto/medio/bajo) y resuelve los casos dicotómicos con asignación binaria, de forma consistente en todas las tablas de decisión.

En la capa de presentación, se valoraron Angular, Django, React y Vue.js, obteniendo Django la mejor puntuación por su capacidad para integrar con sencillez la presentación y la lógica de aplicación mediante vistas y plantillas.

Para la generación de visualizaciones gráficas se compararon Apache ECharts, Chart.js, D3.js y Plotly.js. Considerando la facilidad de uso y la necesidad de construir dashboards interactivos y múltiples vistas con buen tiempo de desarrollo, Plotly.js resultó la opción más adecuada.

En la capa de aplicación (backend) se evaluaron ASP.NET Core, Django, Express y Node.js. De nuevo, Django alcanzó la puntuación más alta, destacando por su cohesión full-stack, su documentación y la rapidez de prototipado.

En la capa de datos se analizaron AllegroGraph, Apache Jena Fuseki, Blazegraph, GraphDB, Neo4j y Virtuoso RDF. Atendiendo a compatibilidad estándar, facilidad de despliegue, documentación y encaje con el resto de las capas, Apache Jena Fuseki ofreció la mejor adecuación para el prototipo y el flujo de trabajo previsto.

Así, del cruce de evaluaciones por capa se derivó la pila tecnológica seleccionada: Django para orquestar la interfaz y la lógica de la aplicación con plantillas HTML/CSS; Plotly.js para las visualizaciones interactivas del dashboard y vistas asociadas; y Apache Jena Fuseki como servidor SPARQL para almacenar los conjuntos RDF y responder a las consultas. Como conector entre aplicación y datos se adoptó SPARQLWrapper, que proporciona una interfaz sencilla y estable para ejecutar consultas y recuperar resultados de forma uniforme. Esta selección maximiza la coherencia entre capas, reduce la fricción de integración y da respuesta directa a los requisitos clave definidos en la primera fase, a la vez que ofrece un margen claro de extensibilidad sin comprometer la estabilidad del núcleo funcional.

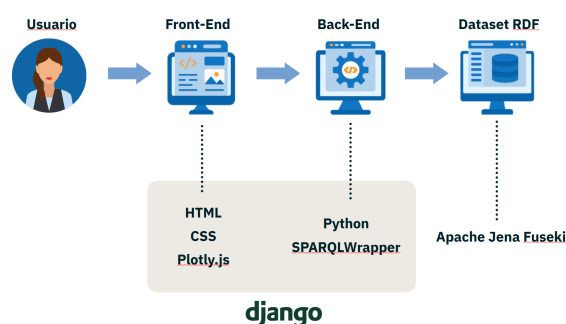


Figura 1. Tecnologías de software utilizadas

4.2.3. Mockups

Los mockups constituyeron una fase esencial dentro del diseño del sistema y se situaron de manera intencionada después de la definición de los casos de uso. Una vez identificadas las acciones principales del usuario y los flujos de interacción, era necesario visualizar cómo se materializaban esas acciones en la interfaz. Los mockups sirvieron para convertir los flujos conceptuales en estructuras visibles, permitiendo verificar la coherencia del modelo de navegación y la adecuación de la arquitectura de la información antes de iniciar la programación. Esta ubicación dentro del proceso metodológico responde a una razón práctica: solo cuando se conocen las tareas y los pasos que el usuario debe realizar tiene sentido representar su entorno visual y la disposición de los elementos que las harán posibles.

El objetivo de esta fase fue anticipar la experiencia de uso y comprobar que la disposición de menús, botones, paneles, gráficos y formularios era clara y funcional. Los mockups permitieron evaluar la jerarquía informativa, la legibilidad de las etiquetas y la relación entre vistas, evitando rediseños posteriores. Actuaron, además, como una herramienta de comunicación, permitiendo que

las decisiones sobre el diseño no dependieran de la intuición, sino de una representación verificable. Su elaboración fue, por tanto, una etapa de validación temprana que consolidó la arquitectura visual del sistema.

Los mockups se desarrollaron con un nivel de detalle intermedio, suficiente para transmitir la estructura, los flujos y los componentes funcionales sin entrar aún en aspectos de estilo o color. Cada vista se diseñó pensando en la tarea que debía resolver el usuario y en la continuidad entre una acción y la siguiente. Se elaboraron cinco pantallas principales, que abarcan el recorrido completo del usuario dentro de la aplicación:

4. *Página de inicio.* Diseñada como punto de acceso a la herramienta. Contiene un selector de dataset, un enlace para subir nuevos conjuntos RDF y accesos directos al dashboard. Desde esta página se establece el primer contacto del usuario con la aplicación. Se cuidó especialmente la claridad del formulario y la visibilidad de los botones de acción. Si no existen datasets cargados, el sistema redirige automáticamente al flujo de subida, evitando pantallas vacías o errores de inicio.

5. *Dashboard.* Representa la visión global del conjunto de datos. Muestra tarjetas con métricas clave (tripletas, clases, propiedades e instancias) y gráficos generados con Plotly.js que describen distribuciones y relaciones. El propósito del mockup fue comprobar que la disposición de los gráficos y de los indicadores permitía una lectura inmediata del estado del dataset. También se validó la integración del formulario lateral de filtrado, garantizando que ambos elementos —estadísticas y filtros— se percibieran como partes de una misma vista y no como secciones separadas.

6. *Resultados de filtrado.* Diseñado para visualizar las instancias que cumplen los criterios definidos por el usuario. El mockup planteó una tabla paginada, con los filtros activos visibles en la parte superior. Cada resultado debía mostrar identificadores legibles y permitir el acceso directo a la ficha de la instancia. Se probó la legibilidad del listado y la posición de los controles para añadir o eliminar filtros, buscando mantener el equilibrio entre la densidad de información y la claridad de lectura.

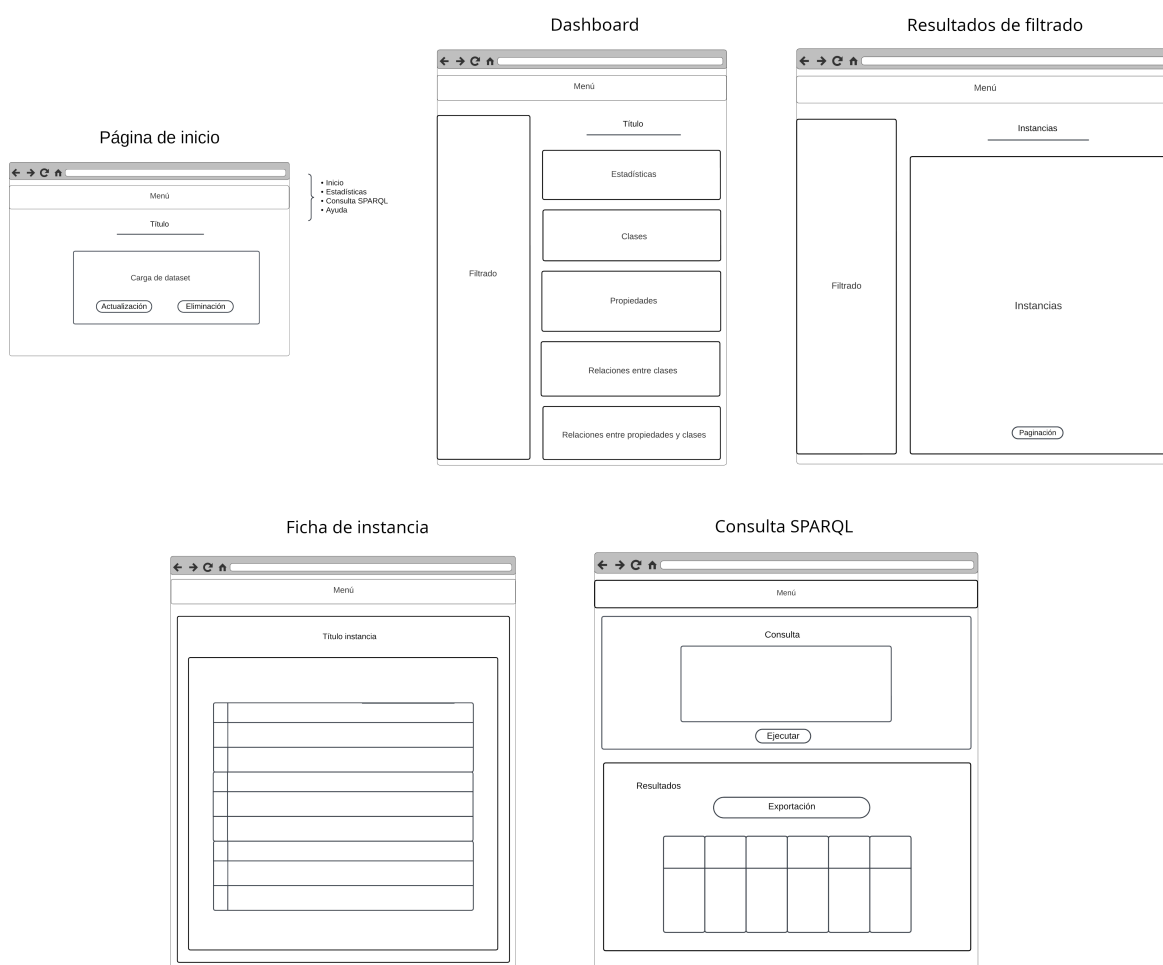


Figura 2. Mockups de la aplicación web

7. *Ficha de instancia*. Esta vista permite examinar una entidad concreta. Los mockups mostraron una tabla con atributos y valores, enlaces a otras instancias relacionadas y a recursos externos. Se diseñó un sistema de navegación cruzada mediante hipervínculos dinámicos, de modo que el usuario pudiera pasar de una entidad a otra sin perder la orientación. Se comprobó también la legibilidad de las URIs y la necesidad de mostrar etiquetas procesadas en lugar de identificadores técnicos.
8. *Consulta SPARQL*. Mockup orientado a ilustrar la página de consulta avanzada. Incluye un campo de texto para introducir sentencias SPARQL, un botón de ejecución y una tabla de resultados. Se añadieron opciones de exportación en CSV, TSV y JSON. Se buscó mantener una estructura limpia que no intimidara al usuario menos experto, pero que siguiera siendo útil para perfiles técnicos.

El desarrollo de los mockups se realizó de manera iterativa. Cada pantalla se ajustó tras revisiones sucesivas. Se compararon las vistas con los casos de uso previamente definidos para confirmar que cada flujo de interacción tenía una correspondencia visual clara. Esta comprobación evitó ambigüedades y aseguró que todas las funcionalidades descritas en los casos de uso tuvieran un reflejo directo en la interfaz.

Además, los mockups sirvieron como base para la implementación en Django, ya que permitieron definir de forma anticipada la estructura de las plantillas HTML y la jerarquía de los bloques reutilizables. Se elaboró una plantilla base para albergar los elementos comunes a cada una de las

páginas, lo que facilitó mantener la consistencia visual y coherencia en la navegación, a la vez que se simplificó el desarrollo. Gracias a este trabajo previo, el paso de diseño a código fue rápido y estable, sin necesidad de modificaciones profundas en la estructura general del sistema.

4.3. Implementación

Tras el diseño, se procedió a la implementación de la herramienta. El código fuente está publicado en un repositorio público bajo licencia Creative Commons Zero v1.0 Universal (CC0 1.0) e incluye un archivo README.md con instrucciones de despliegue y uso (1). De esta forma, la implementación comenzó con la configuración del entorno de trabajo, tanto del framework de desarrollo web como del triplestore para almacenamiento/consulta de datos RDF.

A partir de ahí se generó el esqueleto del proyecto, organizado modularmente para favorecer la mantenibilidad: un fichero central de vistas encargado de procesar las peticiones y orquestar la lógica de la aplicación; un directorio de plantillas HTML que materializa la capa de presentación, extensible desde una plantilla base común; una hoja de estilos CSS; una clase específica para encapsular la conexión con el endpoint SPARQL; y un conjunto de utilidades auxiliares para construir consultas y ejecutar funciones de apoyo.

Una vez desplegadas las tecnologías, el primer paso fue habilitar la ingesta de datos desde la propia interfaz mediante una página de subida que valida formatos RDF admitidos (.ttl, .rdf, .nt) (Figura 3).

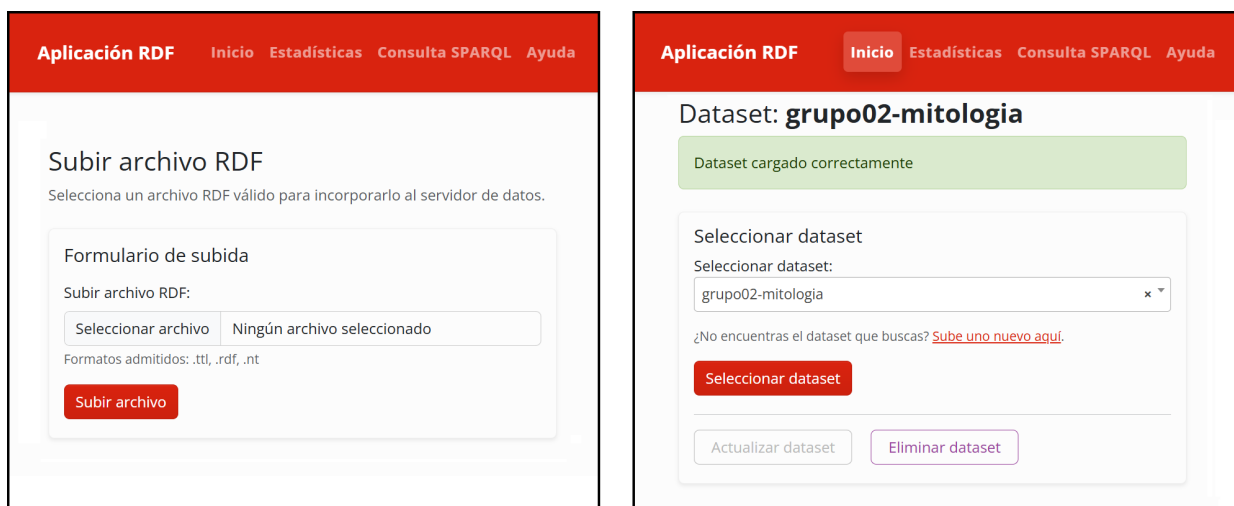


Figura 3. Ingesta de datasets (izquierda) y página de inicio (derecha)

Cada vez que se incorpora un conjunto, el sistema genera automáticamente un archivo JSON

asociado que actúa como resumen estructurado del dataset (con estadísticas globales, estructuras

para gráficos y catálogos de clases y propiedades), redirigiendo después a la página de inicio con el conjunto preseleccionado. Este archivo evita cálculos duplicados y reduce los tiempos de espera, mejorando la experiencia de uso al presentar la información de forma inmediata.

Para reforzar el rendimiento percibido, se añadió una pantalla de progreso con mensajes dinámicos que informan al usuario del proceso que se ejecuta en segundo plano. Posteriormente, se desarrolló la página de inicio, que incluye, además del selector del conjunto activo, accesos a la subida de nuevos conjuntos, a la regeneración del archivo JSON (en caso de cambios o inciden-

cias) y a la eliminación del dataset previa confirmación. Si no hay conjuntos disponibles, el sistema redirige automáticamente al flujo de ingesta

Seguidamente, se implementó la sección de estadísticas, que contempla dos vistas según el contexto (Figura 4). En ausencia de filtros, se presenta un dashboard con tarjetas de métricas (tripleas, clases, propiedades, instancias) y gráficos interactivos que describen distribuciones y relaciones, ofreciendo una panorámica inicial clara. Cuando el usuario aplica filtros, la interfaz muestra un listado paginado de instancias que cumplen los criterios, junto con etiquetas visibles de los filtros activos y opciones para retirarlos individualmente o limpiarlos todos a la vez.

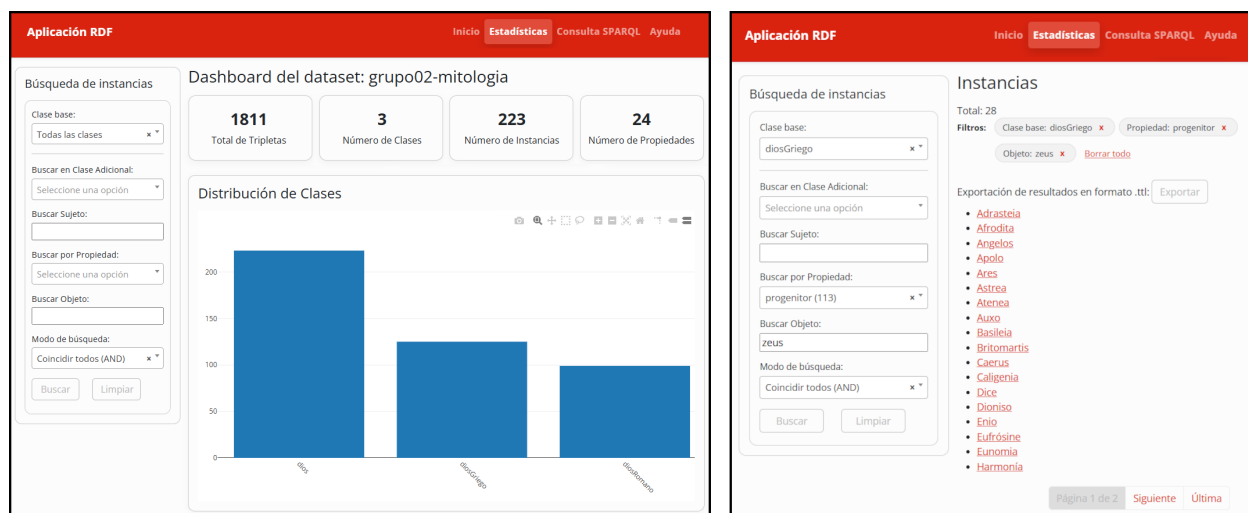


Figura 4. Página de estadísticas (izquierda) y listado de instancias con filtrado (derecha)

En ambas situaciones permanece accesible el formulario lateral de filtrado que permite combinar parámetros: clase base, clase adicional, sujeto, propiedad y objeto, combinables con operadores AND/OR.

La clase base desempeña un papel clave para el control del espacio de búsqueda, ya que fija la población sobre la que se aplican el resto de los filtros. Por ello, no se le aplica el operador AND/OR, pues define el universo sobre el que se ejecutan las condiciones restantes.

A partir de ella, la interfaz adapta el desplegable de propiedades restringiéndolas a las efectivamente utilizadas por dicha clase e incorporando su frecuencia de uso. Este mecanismo permite acercar la búsqueda de instancias a distintos perfiles, haciéndola más accesible e intuitiva incluso para quienes no dominan SPARQL.

Además del filtrado en pantalla, se habilitó la exportación de resultados en formato Turtle (.ttl) ge-

nerando un subgrafo con las instancias que cumplen los criterios junto con toda su información, lo que facilita su reutilización y análisis posterior.

Para la inspección pormenorizada se desarrolló la ficha de instancia, que presenta en tabla las propiedades y valores de la entidad seleccionada, procesando la URIs para mostrar etiquetas legibles. Se implementaron enlaces “inteligentes” que, según el tipo de valor, permiten navegar a la ficha de otra instancia relacionada, abrir un listado de instancias ya parametrizado por clase o consultar el recurso externo en una nueva pestaña; con ello se favorece una exploración encadenada sin pérdida del rastro de búsqueda.

La herramienta incorpora asimismo una página de consulta avanzada donde el usuario puede introducir sentencias SPARQL, ejecutarlas sobre el conjunto activo y visualizar los resultados en tabla con paginación. Desde esta misma interfaz se ofrece la exportación de los resultados tabulares en formatos CSV, TSV y JSON.

Instancia: Afrodita
http://gicd.inf.um.es/wd/datamitologia/afrodita_Q35500

Propiedad	Valor
type	diosGriego
type	dios
label	Afrodita
genero	femenino
imagen	Aphrodite8.jpg
progenitor	Zeus
tieneContraparteRomana	Venus
esLoMismoQue	Astarté
conyuge	Hefesto
pareja	Dioniso

Consulta de datos SPARQL
 Introduce una consulta SPARQL válida sobre el dataset seleccionado

Consulta

Introduce tu consulta SPARQL:

```
PREFIX datamitologia: <http://gicd.inf.um.es/wd/datamitologia/>
PREFIX ontomitologia: <https://gicd.inf.um.es/wd/ontomitologia/>

SELECT ?dios WHERE {
  ?dios a ontomitologia:diosGriego ;
  ontomitologia:progenitor datamitologia:zeus_Q34201 . }
```

Ejecutar consulta

Formato de exportación: CSV

Resultados (28)

dios
http://gicd.inf.um.es/wd/datamitologia/apolo_Q37340

Figura 5. Ficha de instancia (izquierda) y Consulta SPARQL (derecha)

Asimismo, se añadió una página de ayuda accesible desde el menú principal con orientaciones de uso y resolución de incidencias frecuentes. Por otro lado, con el fin de garantizar una presentación homogénea y una correcta adaptación a distintos tamaños de pantalla, la interfaz de la aplicación se ha estructurado empleando el framework Bootstrap 5. Sobre esta base se ha definido una hoja de estilos complementaria, reducida a los ajustes necesarios para adaptar la distribución visual de los componentes, mejorar la legibilidad y asegurar una jerarquía clara entre los elementos informativos. Además, dado que la aplicación está en código abierto, el usuario puede acceder y modificar el tema incluido por defecto – simplex – para incluir cualquier tema compatible con Bootstrap 5, de modo que se puede modificar el aspecto visual sin alterar la estructura funcional del sistema.

Concluida la integración del esquema de interfaz, se realizó una comprobación de accesibilidad mediante la herramienta WAVE (WebAIM), cuyo análisis confirmó la ausencia de errores estructurales y de atributos ARIA. Las únicas observaciones detectadas se correspondían con contrastes de color dependientes del tema elegido por el usuario, por lo que no se modificaron, dado que forman parte de la personalización visual y no de la implementación base. La estructura HTML-Bootstrap, así como las etiquetas asociadas, cumplen así los requisitos esenciales de accesibilidad y navegación por teclado.

La implementación cubre el ciclo completo previsto: gestión de datasets, dashboard y filtrado adaptativo, exploración por instancias, consulta avanzada y exportaciones. Todo ello apoyado en

Bootstrap 5 para la personalización y mejora del estilo visual, el archivo JSON para acelerar la experiencia, en la paginación para controlar la carga informativa y en los estados visibles del sistema que refuerzan la usabilidad y la degradación controlada en escenarios de mayor volumen.

4.4. Evaluación

El objetivo de esta fase fue validar la utilidad, eficacia y usabilidad de la herramienta, comprobando si usuarios sin experiencia previa en tecnologías semánticas podían completar con éxito las tareas clave y si la experiencia de uso resultaba clara y satisfactoria. Para ello se diseñó una prueba estructurada en la que participantes interactuaron con la interfaz, resolvieron un conjunto de tareas y valoraron su percepción mediante un cuestionario estandarizado, complementado con preguntas abiertas.

La evaluación se articuló en tres bloques:

- Tareas prácticas. Se propusieron cinco tareas representativas del uso real: subida de datasets RDF, exploración del dashboard, búsqueda mediante filtrado, ejecución de consultas SPARQL y gestión básica de conjuntos. Cada tarea incluía una pregunta de verificación para confirmar su correcta finalización.
- Cuestionario SUS (System Usability Scale). Se administró el instrumento de 10 ítems (Likert 1–5), alternando afirmaciones positivas y negativas. El SUS mide usabilidad percibida; es sensible a la experiencia subjetiva del participante. Se empleó el cálculo estándar:

transformación de ítems (impares: respuesta-1; pares: 5-respuesta) y multiplicación de la suma por 2,5 para obtener la puntuación final (0-100). Se considera una puntuación mayor que 68 como usabilidad por encima de la media y mayor o igual que 78,9 como excelente.

- Preguntas abiertas. Se recogieron observaciones sobre elementos confusos, funcionalidades más útiles, mejoras deseables y posibles usos. Las respuestas se codificaron temáticamente con un enfoque cualitativo inductivo, contabilizando la frecuencia de aparición de cada tema para identificar patrones, fortalezas y áreas de mejora.

La recogida de datos se realizó mediante Google Forms, aportando contextualización, instrucciones, tareas guiadas, cuestionario SUS y bloque final de observaciones. En cumplimiento de las directrices éticas de la Universidad de Murcia, se facilitó hoja de información y consentimiento informado; la participación fue voluntaria, anónima y con fines exclusivamente académicos. Se publicó una copia PDF de la encuesta y la hoja de cálculo con resultados agregados en Zenodo (2).

Con respecto a los participantes, la evaluación fue realizada por un grupo de 16 personas con formación en informática y experiencia en el uso de las TIC, sin conocimientos previos en RDF/SPARQL. Este perfil permitió focalizar la evaluación en usabilidad de la interfaz, eficacia de los flujos y curva de aprendizaje, aislando el efecto de la especialización semántica.

Los datos recogidos se exportaron a una hoja de cálculo organizada en tres pestañas correspondientes a cada sección de la evaluación publicada en el conjunto de datos de Zenodo mencionado anteriormente.

Respecto a la resolución de tareas, que estudia la efectividad del sistema, se obtuvo una efectividad global del 100%: todos los participantes completaron correctamente las cinco tareas. Las respuestas de verificación no detectaron errores ni incidencias, indicando que la herramienta es funcional y comprensible para el perfil definido y que la interfaz facilita la interacción con datos RDF sin necesidad de experiencia previa en tecnologías semánticas.

En cuanto al cuestionario SUS; la puntuación media fue de 81,41 puntos, situada en el rango de excelente usabilidad. Esta cifra refleja una percepción positiva en términos de intuitividad, eficiencia y satisfacción con el diseño y el funcionamiento general.

Finalmente, la sección de preguntas abiertas recoge valoraciones más detalladas y extensas. Para cada una se elaboró una tabla con una codificación temática de las respuestas y la frecuencia de aparición.

En la primera pregunta, orientada a identificar elementos confusos o poco intuitivos, una parte significativa de los participantes declaró no haber experimentado dificultades, lo que refuerza la solidez general de la interfaz. No obstante, las respuestas restantes señalaron áreas de mejora concretas. Destaca, en primer lugar, la insuficiente retroalimentación visual en momentos clave del flujo, por ejemplo, al seleccionar o cargar un dataset, lo que genera incertidumbre sobre la ejecución efectiva de la acción. En segundo lugar, se apuntó la falta de contextualización en la sección de consulta SPARQL, que para usuarios sin experiencia previa en tecnologías semánticas puede resultar un punto de entrada abrupto. También aparecieron observaciones puntuales de ambigüedad terminológica (el rótulo "Filtros" no siempre transmite con precisión el funcionamiento del panel), así como dudas sobre los conteos que acompañan al desplegable de propiedades y la ubicación menos esperada de algunos controles de navegación. En conjunto, estos hallazgos sugieren reforzar el feedback de sistema (confirmaciones y estados), clarificar los rótulos (etiquetas y mensajes breves de ayuda) y ofrecer pistas contextuales en las vistas avanzadas.

En la segunda pregunta, centrada en las funcionalidades más útiles, el patrón fue nítido: la búsqueda mediante filtrado fue la opción más valorada. Los participantes destacaron su carácter accesible e intuitivo. En segundo plano, pero con apreciación clara, se situaron la consulta SPARQL, que aporta profundidad para usuarios avanzados. Las visualizaciones del dashboard también se percibieron como una puerta de entrada eficaz para comprender la "forma" del conjunto. De manera más puntual se valoraron el cambio ágil de dataset y la exportación de resultados. En suma, los usuarios reconocen el binomio dashboard-filtrado como la principal palanca para explorar RDF sin requerir conocimientos técnicos.

La tercera pregunta indagó en mejoras y nuevas funcionalidades. Las propuestas fueron diversas, sin concentrarse en un único aspecto, pero pueden agruparse en bloques coherentes. En el ámbito de estadísticas y visualización, se reclamó una mayor coherencia entre el panel de filtrado y las gráficas (p. ej., separar o conectar mejor ambos elementos), contextualizar cada visualización con descripciones breves, incorporar una vista de relaciones (grafo de entidades), ofrecer

estadísticas por clase base, ampliar la flexibilidad del filtrado y representar las ontologías utilizadas. En el plano de interfaz y usabilidad, se sugirió hacer más explícita la navegación (marcar la sección activa), mejorar la adaptación a pantallas grandes, autoaplicar filtros al modificar entradas en el filtrado, facilitar la lectura de nombres largos en el selector de datasets y clarificar aún más el panel de filtrado. Para SPARQL, se propuso añadir una guía breve o ejemplos iniciales. En gestión de datasets, se mencionó la posibilidad de ofrecer operaciones avanzadas (duplicar, modificar, fusionar) y confirmaciones visuales tras la carga. Finalmente, aparecieron peticiones puntuales sobre exportaciones (más metadatos descriptivos) y aspectos técnicos (HTTPS y dominio propio). Estas buscan pulir coherencia, orientación, ampliación de las funcionalidades y continuidad de la experiencia.

La cuarta pregunta abordó la utilidad percibida en contextos personales, académicos o profesionales. La aportación más repetida fue la exploración y análisis ágil de datasets RDF de diversa procedencia: la posibilidad de “cargar y comprender” con rapidez un conjunto se considera especialmente valiosa para diagnósticos preliminares y toma de contacto con datos nuevos. También se resaltó la aplicabilidad en escenarios de gran volumen, donde la combinación de filtrado, consulta y visualización agiliza el trabajo. En cuanto a perfiles, se identificaron usos para expertos en RDF y profesionales de la información, pero también un valor didáctico para el aprendizaje de RDF/SPARQL mediante la experimentación. Se mencionaron, además, casos más específicos: recuperación avanzada mediante SPARQL, organización temática y validación de ontologías, lo que muestra la versatilidad de la herramienta. Una respuesta indicó no ver utilidad en el plano personal, coherente con el carácter especializado del sistema.

Los resultados de la evaluación confirman la solidez general del modelo y la efectividad de la aplicación. La herramienta demostró ser estable, comprensible y funcional incluso para usuarios sin experiencia previa en RDF o SPARQL. El 100 % de efectividad en las tareas y una usabilidad percibida excelente (SUS = 81,41) evidencian que los principios de diseño aplicados —claridad, jerarquía visual y coherencia entre vistas— se tradujeron en una experiencia de uso intuitiva y satisfactoria. El sistema consigue que las funciones principales —visualización, filtrado y consulta— se integren sin rupturas, lo que permite recorrer el ciclo completo de análisis de un dataset RDF desde una lógica progresiva y guiada.

El sistema de filtrado se consolida como la pieza clave del modelo. Su diseño traduce la lógica

abstracta de las tripletas RDF en una experiencia de búsqueda comprensible, basada en formularios dinámicos y operadores simples. La posibilidad de combinar criterios y obtener resultados en tiempo real acerca la consulta semántica a una práctica habitual para el usuario general. En la evaluación, esta función fue la más valorada, no solo por su utilidad práctica, sino porque actúa como mediadora entre la visión global del dashboard y el análisis detallado de las instancias. Permite descubrir patrones sin necesidad de escribir código y, en conjunto, reduce la barrera cognitiva de acceso a los datos enlazados.

El dashboard es un punto de entrada eficaz para comprender la estructura del conjunto de datos. La presentación de métricas globales, combinada con visualizaciones interactivas, ofrece una representación clara de la “forma” del grafo y de sus proporciones internas. Los usuarios lo percibieron como un punto de orientación inicial que ayuda a contextualizar la exploración posterior. Este resultado confirma que la visualización no es un complemento, sino un elemento fundamental para transformar la complejidad de los datos RDF en conocimiento comprensible. La lectura visual del conjunto favorece la interpretación y mejora la percepción general de control sobre el sistema.

Aun así, el rendimiento observado y la buena acogida no eliminan ciertos condicionantes del sistema que limitan su alcance. El más relevante es la dependencia de la calidad del modelado semántico de los datasets cargados (Zaveri et al., 2013). Cuando el grafo presenta incoherencias —clases indefinidas, propiedades mal empleadas, etiquetas ausentes o relaciones incompletas—, la interfaz pierde capacidad explicativa. Las métricas dejan de reflejar la estructura real del dominio y las visualizaciones pueden inducir interpretaciones erróneas. Este problema es inherente a la naturaleza heterogénea de los datos RDF, pero marca un límite importante: la herramienta no puede suplir deficiencias ontológicas de los conjuntos de datos que se cargan en ella.

También se identificaron restricciones operativas ligadas a la escala. En datasets muy extensos, el tiempo necesario para aplicar filtros o generar resultados puede aumentar de forma perceptible, incluso con mecanismos de paginación y resúmenes precalculados. Algunas combinaciones de criterios —por ejemplo, búsquedas sobre clases muy pobladas o propiedades con alta cardinalidad— incrementan el volumen intermedio de datos procesados y, con ello, la latencia en las respuestas. Aunque la arquitectura multicapa y el uso de archivos JSON optimizan la experiencia, persiste un límite práctico al rendimiento bajo car-

gas elevadas. Este comportamiento es coherente con la naturaleza intensiva de las consultas semánticas y con el objetivo del prototipo, que prioriza la claridad y la robustez sobre la velocidad extrema.

Otra limitación, de carácter cognitivo, está vinculada a la curva de aprendizaje conceptual. Si bien la herramienta reduce considerablemente la complejidad técnica, sigue requiriendo un mínimo conocimiento de los conceptos básicos de RDF: clase, instancia, propiedad, sujeto y objeto. Entender estas nociones resulta necesario para interpretar correctamente los resultados del filtrado o las relaciones mostradas en las fichas de instancia. Sin este bagaje, la experiencia puede seguir siendo operativa, pero parte del significado semántico se pierde. Algo similar ocurre con la sección SPARQL: aunque está integrada y funcional, su aprovechamiento pleno requiere familiaridad con el lenguaje y con el esquema del dataset activo.

5. Conclusiones

Este trabajo ha desarrollado y validado una aplicación web para la consulta y visualización de grandes conjuntos de datos semánticos RDF. Se han cumplido los objetivos propuestos y se ha demostrado la viabilidad de un modelo operativo que combina visión global, búsqueda guiada y consulta avanzada en un mismo entorno. La herramienta articula tres ejes complementarios: un dashboard que sintetiza el grafo mediante métricas y visualizaciones interactivas; un sistema de filtrado adaptativo que aproxima la consulta semántica a la búsqueda tradicional; y un espacio SPARQL pensado para usuarios con conocimientos técnicos. Todo ello se completa con exportaciones en formatos estándar (CSV, TSV, JSON y Turtle), lo que asegura la trazabilidad, la interoperabilidad y la reutilización de los resultados.

El proyecto ha definido de forma explícita los requisitos funcionales y no funcionales, ha justificado la pila tecnológica (Django, Plotly.js y Apache Jena Fuseki) y ha construido una arquitectura multicapa que separa la presentación, la lógica y los datos. Esta estructura ha permitido una implementación modular, fácilmente mantenible y extensible. La aplicación final cubre el ciclo completo de interacción: ingesta y gestión de datasets, visualización global, filtrado, exploración por instancias, consulta avanzada y exportación de datos. Todo ello se ha logrado manteniendo un equilibrio entre rendimiento técnico y comprensión cognitiva, con un diseño centrado en la claridad y la accesibilidad.

Entre las aportaciones más relevantes destaca la integración coherente de tres niveles de lectura:

global, intermedia y detallada. El usuario puede pasar del conjunto al elemento sin perder la orientación. Este tránsito continuo entre escalas no es habitual en las herramientas semánticas existentes, y constituye una de las principales innovaciones del modelo. La independencia de dominio —al adaptarse automáticamente a la estructura del dataset cargado— amplía además su potencial de reutilización. La publicación del código bajo licencia abierta (CC0 1.0) refuerza esta orientación y facilita la extensión del sistema por parte de otros equipos de investigación o instituciones.

La evaluación con usuarios confirma la validez del enfoque. Se alcanzó una efectividad del 100 % en las tareas y una puntuación SUS de 81,41, que sitúa la herramienta en el rango de excelente usabilidad. Este resultado refleja la solidez de la arquitectura de la información aplicada: una estructura que reduce la carga cognitiva, orienta al usuario en cada paso y mantiene la coherencia visual y semántica. A partir de la evaluación se incorporaron mejoras directas, como mensajes de confirmación, clarificación terminológica y marcadores visuales de navegación, que consolidan la fluidez de la experiencia y refuerzan la comprensión de los datos.

Más allá del desarrollo técnico, el proyecto demuestra el valor de la arquitectura de la información como marco metodológico para el diseño de aplicaciones semánticas. Su aplicación ha permitido traducir la complejidad de los grafos RDF a un lenguaje visual y funcional comprensible, basado en principios de jerarquía, claridad y coherencia. Este modelo puede servir como referencia para futuras herramientas que busquen integrar visualización, filtrado y consulta en un mismo entorno, priorizando la interpretación humana sin renunciar al rigor de los estándares de la web semántica. Supone, en este sentido, un paso hacia sistemas más transparentes y accesibles, capaces de acercar los datos enlazados a perfiles diversos.

En cuanto a las líneas de evolución, se identifican varias direcciones de desarrollo. Una de ellas es el enriquecimiento de la documentación interna y las ayudas contextuales, especialmente en las visualizaciones y en la sección SPARQL. También se prevé la incorporación de un módulo de gestión avanzada de datasets, con funciones de edición, duplicación o combinación de conjuntos. Otra línea es la visualización de vocabularios y ontologías, que permitiría mostrar de forma gráfica las relaciones conceptuales del modelo subyacente. Además, la integración de un modo multilingüe ampliaría su aplicabilidad en entornos internacionales y educativos. Finalmente, se plantea la inclusión de más visualizaciones y vistas

en forma de grafo, que completen la lectura tabular y estadística con una representación relacional más intuitiva.

El modelo aquí desarrollado no debe entenderse como un punto final, sino como una base sólida para nuevas exploraciones. La experiencia acumulada muestra que la combinación de ingeniería del software, diseño informacional y tecnologías semánticas puede generar herramientas potentes y comprensibles a la vez. En este sentido, la propuesta no solo ofrece una solución concreta, sino un enfoque replicable para otros contextos donde la complejidad de los datos exija una arquitectura de la información clara, visual y centrada en el usuario. El equilibrio alcanzado entre accesibilidad y profundidad constituye su principal aportación y su mejor argumento para futuras aplicaciones y líneas de investigación.

Notas

(1) https://github.com/ana03lm/rdf_app.git

(2) <https://zenodo.org/records/15319362>

Declaración de autoría

Ana López Morales, Juan-Antonio Pastor-Sánchez, José A. Ruipérez-Valiente (conceptualización, desarrollo, edición y revisión, igual).

Referencias

- Angles, Renzo (2015). Introducción a las bases de datos RDF. <http://www.renzoangles.net/files/libro.pdf>
- Boumechaal, Hasna; Boufaïda, Zizette (2023). Complex Queries for Querying Linked Data. // *Future Internet*. 15:3. <https://doi.org/10.3390/fi15030106>
- Castells, Pablo (2005). *La Web Semántica*. // Bravo Santos, Crescencio; Redondo Duque, Miguel Ángel (eds.) *Sistemas interactivos y colaborativos en la web*. Universidad de Castilla-La Mancha

- Desimoni, Federico; Po, Laura (2020). Empirical Evaluation of Linked Data Visualization Tools // *Future Generation Computer Systems*. 112, 258-282. <https://doi.org/10.1016/j.future.2020.05.038>
- Hartig, Olaf (2013). An Overview on Execution Strategies for Linked Data Queries. // *Datenbank-Spektrum*. 13:2, 89-99. <https://doi.org/10.1007/s13222-013-0122-1>
- Hitzler, Pascal y Janowicz, Krzysztof (2013). Linked Data, Big Data, and the 4th Paradigm. // *Semantic*. 4:3, 233-235. Disponible en: <https://doi.org/10.3233/SW-130117>
- Ivanova, Valentina; Lambrix, Patrick; Lohmann, Steffen, Pesquita, Catia (2019). Visualization and Interaction for Ontologies and Linked Data. // *Journal of Web Semantics*. 55, 145-149. <https://doi.org/10.1016/j.websem.2018.10.001>
- Pérez-Montoro, Mario (2010). Arquitectura de la información en entornos web. // *El Profesional de la Información*. 19:4, 333-338. <https://doi.org/10.3145/epi.2010.jul.01>
- W3C (2014). RDF 1.1 Primer. <https://www.w3.org/TR/rdf11-primer/>
- The Information Lab (2023). Qué es un dataset. <https://www.theinformationlab.es/blog/que-es-un-dataset/>
- Resmini, Andrea; Rosati, Luca (2011). *Pervasive Information Architecture: Designing Cross-Channel User Experiences*. Burlington (MA): Morgan Kaufmann. ISBN 978-0-12-382094-5.
- Sota Martínez, Sergio (2016). *Navegación visual por datos semánticos guiada por agente software*. Zaragoza: Universidad de Zaragoza. <https://zaguan.unizar.es/record/61209>
- Wohlgemant, Gerhard; Mouromtsev, Dmitry; Paulov, Dmitry; Emelyanov, Yury; Morozov, Alexey (2019). A Comparative Evaluation of Visual and Natural Language Question Answering Over Linked Data // *Proceedings of the 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2019)*. <https://doi.org/10.5220/0008364704730478>
- Zaveri, Amrapali; Rula, Anisa; Maurino, Andrea; Pietrobon, Ricardo; Lehmann, Jens (2016). Quality Assessment for Linked Data: A Survey. // *Semantic Web* // 7:1, 63-93. <https://www.semantic-web-journal.net/content/quality-assessment-linked-data-survey>

Enviado: 2025-11-08. Segunda versión: 2025-12-18.

Aceptado: 2025-12-19.