

Cómo generar valor a partir de los datos: formatos, técnicas y herramientas para analizar datos abiertos



1. INTRODUCCIÓN:	3
2. TIPOS DE FORMATOS DE DATOS Y CONVENIENCIA DE USO	5
2.1 Datos para humanos	8
2.2 Datos para máquinas	9
2.3 Formatos de datos híbridos	13
2.4 Otros formatos de datos	15
2.4.1 <i>Formatos de datos para ingeniería de producto</i>	16
2.4.2 <i>Formatos de datos en simulación científica</i>	17
3. EXTRACCIÓN Y CONSULTA DE DATOS ABIERTOS.	18
3.1 Licencias y buenas prácticas de uso.	19
3.2 Ejemplo de utilización de API para extracción y análisis de datos.	21
4. HERRAMIENTAS Y TECNOLOGÍAS PARA EL ANÁLISIS DE DATOS ABIERTOS.	23
4.1. Usuarios intermedios. Ejemplos de uso de algunas API.	24
4.1.1. <i>AEMET</i>	24
4.1.2 <i>EUROPEANA</i>	25
4.1.3 <i>The Sports DB</i>	27
4.1.4 <i>Spotify</i>	29
4.2 Usuarios avanzados. Uso de API mediante lenguajes de programación.	34
5.2.1 <i>Python</i>	34
4.2.2 <i>R</i>	36
5. CONCLUSIONES Y VISIÓN TECNOLÓGICA DE FUTURO	44
6. RECURSOS	46

Contenido elaborado por Alejandro
Alija, experto en Transformación
Digital
y datos abiertos.

Este estudio ha sido desarrollado en el marco de la Iniciativa Aporta desarrollada por el Ministerio de Economía y Empresa, a través de la Entidad Pública Empresarial Red.es, y en colaboración con el Ministerio de Política Territorial y Función Pública. Los contenidos y los puntos de vista reflejados en esta publicación son responsabilidad exclusiva de su autor. El equipo Aporta no garantiza la exactitud de los datos incluidos en el estudio. El uso de este documento implica la expresa y plena aceptación de las condiciones generales de reutilización referidas en el aviso legal que se muestra en: <http://datos.gob.es/es/aviso-legal>

1. INTRODUCCIÓN:

¿Por qué este informe?

El contenido digital en Internet aumenta de forma exponencial. De la misma forma, aumenta la complejidad para encontrar información veraz y rigurosa, así como los derechos de los ciudadanos en lo que a privacidad y derechos de propiedad digital se refiere.

En la superficie de ese océano de Internet, la mar parece en calma permanente. Tan solo tenemos que escribir la palabra que nos interese en Google y recibiremos cientos de miles de entradas de las cuales solo inspeccionamos las 5 primeras. Navegamos, copiamos, pegamos y asumimos como cierta la mayor parte de la información a la que accedemos en Internet. Pero la aparente facilidad y accesibilidad de entrada oculta un iceberg de complejidad.

El volumen de los datos abiertos almacenados en repositorios públicos no deja de crecer. La plataforma europea [Europeana](#) cuenta con un catálogo de más de 58 millones de recursos registrados mientras que la [Biblioteca Digital Pública de América](#) sobrepasa los 29 millones. En el ámbito científico, [Academic Torrents](#), acumula más de 33TB de datos distribuibles por y para científicos de todo el mundo.

Para regular el uso y distribución de datos abiertos existen más de 100 tipos de licencias distintas. El ecosistema de tecnologías (open-source) desarrolladas para la clasificación, almacenado, búsqueda e intercambio de datos abiertos es un conjunto abrumador de nombres y familias técnicas (Lucene, Elasticsearch, MongoDB, Solr, Jenkins, Git, etc.). Bajo este panorama de explosión descontrolada de datos, parece razonable pensar que una de las competencias técnicas del trabajador del futuro será la habilidad para buscar, encontrar, procesar y comunicar historias apasionantes sustentadas en los datos.

El mundo empresarial trata de alinearse con esta tendencia tan rápido puede. Las principales compañías del mundo se afanan en crear nuevos departamentos de data products, así como

las principales tecnológicas se encargan de convencer a sus clientes para que apuesten por convertirse en una data-driven company.

En este contexto, cada vez más se necesitan perfiles profesionales con, lo que hemos venido a denominar, competencias digitales avanzadas. Cualquiera que aspire a alcanzar un nivel de desempeño avanzado en el espacio digital ha de conocer las herramientas de acceso a los datos, así como los lenguajes de programación que permitirán al trabajador transformar y construir nuevos productos y servicios basados en datos. El profesional competente ha de comprender cómo manejar APIs diversas para consultar información de forma escalable y segura. Ha de ser capaz de recepcionar grandes cantidades de datos en formatos diversos, almacenarlos, procesarlos y volver a compartirlos de acuerdo con las licencias de uso y distribución.

¿Qué vas a encontrar en este informe?

Este informe está recomendado para aquellos que deseen obtener un mayor nivel de detalle sobre las competencias digitales destacadas anteriormente. En él se exploran diferentes técnicas para la extracción y el análisis descriptivo de los datos contenidos en los repositorios de datos abiertos.

El informe está orientado hacia un público general no especialista, sin embargo, aquellos lectores familiarizados con el tratamiento e intercambio de datos en el mundo web se encontrarán con una lectura familiar y reconocible.

El informe se estructura de la siguiente manera:



Formatos de datos. Se introduce al lector en los distintos tipos de formatos de datos que se encontrará, de forma más habitual, al navegar por los repositorios de datos abiertos. Al final de esta sección se introducen, además, algunos formatos de datos más específicos de utilidad en el mundo científico e ingenieril.



Mecanismos de intercambio de datos a través de La Web. En esta sección se explican varios ejemplos prácticos que ilustran la forma de extraer datos de interés de algunos de los repositorios con más presencia en Internet. Siempre que se consumen datos de algún repositorio de datos abiertos, se hace bajo alguna licencia de uso y reutilización. Pese a que estas licencias, en su mayoría, persiguen fomentar la reutilización y buen uso de los datos, es importante conocer los principales tipos y sus características.



Principales licencias, orientando al lector hacia su identificación y reconocimiento.



Herramientas y tecnologías para el análisis de datos. Esta sección se vuelve ligeramente más técnica. En ella, se muestran diferentes ejemplos de extracción de información útil de los repositorios de datos abiertos, haciendo uso de algunos fragmentos cortos de código en diferentes lenguajes de programación.



Conclusiones, donde se ofrece una visión tecnológica de futuro, con la mirada puesta en los más jóvenes que constituirán la fuerza de trabajo del futuro.

2. TIPOS DE FORMATOS DE DATOS Y CONVENIENCIA DE USO

Los datos toman forma a través de ficheros de datos. Técnicamente, el proceso de materializar datos en forma de ficheros se denomina serialización. Quizás, uno de los aspectos fundamentales a comprender de este universo sea la diversidad en los ficheros de datos existentes.

Un buen ejemplo de esta variedad puede verse en el [sitio web de la Biblioteca Británica](#). En el apartado correspondiente al [servicio de descarga de los metadatos](#), los diferentes conjuntos de datos pueden descargarse en formato N-triples, RDF/XML y CSV, todos ellos comprimidos en paquetes ZIP. Por su lado, el catálogo de datos del [repositorio de datos abiertos del ayuntamiento de Madrid](#) dispone de conjuntos de datos abiertos disponibles para su descarga en formatos tan diversos como JSON, GEOJSON, TXT, MS Excel, KML, SHP, etc. La [iniciativa estatal de datos abiertos](#) dispone de un catálogo de más de 20.000 conjuntos de datos que no para de crecer.

Para tratar de clasificar¹ los diferentes tipos de ficheros de datos se puede hablar de ficheros diseñados para ser leídos e interpretados por humanos frente a aquellos cuyo objetivo es ser, eficientemente, procesados por máquinas. Algunos formatos cumplen conjugan ambos propósitos, de forma que pueden clasificarse como formatos híbridos.



Datos para humanos: En el contexto de este documento, se consideran datos diseñados para humanos aquellos que pueden ser abiertos y visualizados, con tan solo uno o dos clicks, a través de herramientas digitales estándar - como un PC, smartphone o tablet-. Los datos diseñados para humanos, han de ser legibles por cualquiera, sin necesidad de contar con competencias específicas para el análisis de datos como por ejemplo, tablas de datos en formato de texto plano. Además, mantienen una estructura muy simple y permiten representar información de baja dimensionalidad, es decir, información prácticamente plana.

¹ Existen diferentes formas de clasificar los ficheros de datos. En muchas ocasiones el lector se encontrará con clasificaciones que hacen referencia al nivel de estructuración que presentan los ficheros, hablándose entonces de datos estructurados, semi-estructurados o desestructurados. En este informe, sin embargo, hemos optado por clasificar los datos según su uso. Para más información sobre el primer tipo de clasificación recomendamos consultar el [siguiente artículo](#).



Datos para máquinas: *Por el contrario, los formatos de datos diseñados para ser procesados por las máquinas, disponen de una estructura compleja, que puede llegar a representar varias dimensiones de los datos (como en el caso de imágenes o videos). Además, los datos diseñados para máquinas pueden distribuirse y almacenarse de forma comprimida para ahorrar espacio de almacenamiento. Los datos para máquinas, pueden contener etiquetas intercaladas entre los propios datos, que sirven a la máquina para detectar secuencias o determinados contenidos.*



Formatos de datos híbridos. *El mundo de los datos no es muy diferente a la vida real, y como en la vida real, no todo es blanco o negro. Por ello, se está imponiendo un formato de dato híbrido, que si bien mantiene una estructura legible por cualquier humano, contiene elementos intercalados que ayudan a las máquinas a procesar estos formatos de forma eficiente, permitiendo mayores niveles de dimensionalidad en las estructuras de datos.*



A continuación, se describen los principales formatos utilizados en los repositorios de datos, teniendo en cuenta esta clasificación.

2.1 Datos para humanos

Los formatos de datos diseñados para ser consumidos por humanos añaden elementos de diseño para facilitar la visualización y presentación de datos en forma de resultados. Colores, separadores de celdas, texto y datos intercalados o figuras, son sólo algunos de los elementos que incorporan estos formatos para facilitar la comprensión por usuarios no especialistas. Sin embargo, estos formatos que ofrecen tanta libertad de edición al usuario final, no son adecuados para su intercambio ni su procesamiento automático, pues las máquinas han de estar programadas para eliminar la parte estética y visual del verdadero contenido.

En este dominio, el estándar de mercado absoluto es el formato (no abierto) Microsoft (MS) Excel (XSLX). Pese a ser una interfaz muy amigable para el usuario final, el formato de datos de MS Excel no es un buen formato cuando se trata de ser fácilmente intercambiable, ligero y reproducible. En este sentido, su alternativa natural es el tradicional formato CSV o valores separados por comas (ver apartado de formatos de datos híbridos). Si bien el formato CSV ha perdurado en los años gracias a su ligereza y fácil manejo e interpretación, en los últimos años, con el auge de Internet y la conectividad, otro formato (del que también hablaremos en la sección datos híbridos) ha llegado para imponerse. Es el formato JSON (Java Script Object Notation), que fue originariamente el formato de datos nativo del lenguaje de programación JavaScript.

Existen más formatos de datos diseñados especialmente para humanos, aunque en esta sección hemos preferido mencionar los de uso más frecuente para el análisis de datos. El formato PDF por ejemplo, es un formato especialmente indicado para intercambiar documentos con independencia del programa o sistema con el que haya sido generado dicho documento. El formato PDF embebe en su interior toda la información necesaria para mostrar e imprimir el documento (fuentes, figuras, imágenes, etc.).

También cabe destacar que el lenguaje natural se considera en la actualidad un tipo de dato para humanos con una enorme potencialidad cuando se consigue ejecutar su análisis de forma eficiente. No debemos olvidar que, aproximadamente, el 80% de la información que

se genera en la actualidad, se considera *información (datos) sin estructurar* y buena parte de esta tiene forma de lenguaje natural (correo electrónico, redes sociales, video y audio streaming, etc.). Los datos en forma de lenguaje natural que se generan y propagan por Internet exceden con mucho el conjunto de datos estructurados o semi-estructurados (los que se abordan en este informe) orientados a humanos. La investigación en tecnologías del lenguaje desde el punto de vista de extracción y análisis de información es un campo de ferviente actividad en la actualidad dado el valor de negocio que encierran los datos en lenguaje natural. Para más información se recomienda consultar el [siguiente artículo](#).

2.2 Datos para máquinas

El objetivo de los formatos de datos diseñados para las máquinas es la eficiencia cuando los encargados de intercambiarlos y procesarlos son programas informáticos. Para este propósito destacan los formatos *XML* (Extensible Markup Language) y *RDF* (Resource Description Framework).

Hasta la llegada de JSON, XML ha sido el estándar de intercambio de datos en Internet. Básicamente, XML puede considerarse como el lenguaje de la Web. Este es un formato (de texto) en el que los datos se organizan formando estructuras dominadas por etiquetas que no han sido previamente definidas. Dado que, las etiquetas definen un patrón estructurado, los formatos XML alcanzaron su máximo de popularidad como fichero de intercambio en Internet. El exceso de complejidad de algunas estructuras de datos creadas sobre XML unido al excesivo volumen en bytes para el intercambio, han motivado su progresiva sustitución por otros formatos de serialización como JSON.

RDF no es formalmente un formato de serialización de datos sino un modelo de datos especialmente pensado para la *web semántica* donde la estructura básica de datos se denomina “triple”, *tripleta* o *3-tupla* y se define como sujeto-predicado-objeto. Dada la

popularidad del modelo de datos RDF en los repositorios de datos abiertos² merece la pena detenerse ligeramente en algunos ejemplos prácticos sobre su uso.

Ejemplo 1.

Por ejemplo, para la caracterización completa de una obra literaria será necesario referenciar, entre otros, el título de la obra, su autor, la fecha de publicación, la editorial, etc. Así, la siguiente afirmación “Lewis Carroll es el autor de la obra Alicia en el País de las Maravillas” tiene su reflejo según la estructura de datos definida por RDF de la siguiente forma:

sujeto	predicado	objeto
Alicia en el País de las Maravillas	autor	Lewis Carroll

Además de definir este modelo formal para organizar la información, RDF ha sido diseñado para intercambiar y enriquecer información a medida que ésta viaja por la Web. Por esto, RDF utiliza URIs (Uniform Resource Identifiers) para identificar objetos de forma única en la Web. Es decir, el modelo de datos RDF no se conforma con estructurar datos (similares al lenguaje natural) en un formato procesable por máquinas, sino que, además identifica unívocamente los objetos a través de la Web.

² La práctica totalidad de las instituciones dedicadas a proteger el patrimonio cultural de una nación utiliza alguna implementación de RDF para almacenar e intercambiar sus datos.

Ejemplo 2.

Siguiendo con el ejemplo anterior, un ejemplar original de Alicia en el País de las Maravillas registrado en la Universidad de Harvard en EEUU queda identificado como <https://hdl.handle.net/2027/hvd.32044020076089> y su sintaxis según RDF quedaría como:

sujeto	predicado	objeto
Alicia en el País de las Maravillas	autor	Lewis Carroll
https://hdl.handle.net/2027/hvd.32044020076089	autor	Lewis Carroll

La combinación de varias tripletas caracteriza, por ejemplo, una obra literaria en su totalidad, con todas aquellas propiedades relevantes relacionadas con dicha obra. El conjunto de tripletas se suele denominar grafo y es la forma más sencilla de imaginar un modelo *RDF completo*.



GOBIERNO DE ESPAÑA
 MINISTERIO DE POLÍTICA TERRITORIAL Y FUNCIÓN PÚBLICA
 MINISTERIO DE ECONOMÍA Y EMPRESA

Ejemplo 3.

Así por ejemplo, esta fotografía de Albert Einstein localizada a través del buscador del sitio de [Europeana.eu](http://www.europeana.eu) https://www.europeana.eu/portal/es/record/2021642/Atlantispubliek_Default_aspx_det ail_52770968.html?q=Albert+einstein referencia al proveedor original de la fotografía <http://www.schouwen-duiveland.nl/>. El ejemplo anterior puede parecer naive comparado con la complejidad del modelo de datos RDF. Sin embargo, si se multiplica este ejemplo por los más de 50 millones de recursos web alojados en Europeana.eu que no pertenecen a esta plataforma, la existencia de un modelo de datos y sus correspondientes implementaciones y herramientas asociadas están totalmente justificados.

Pese a que formalmente RDF no es un formato concreto, sino un modelo de organización de datos, es habitual encontrar repositorios de datos abiertos que ofrecen conjuntos de datos en formato RDF. En realidad, el tipo más general de serializar datos bajo el esquema RDF es el RDF/XML. Esto es un fichero XML que organiza los datos en su interior siguiendo un esquema RDF. Por supuesto, dada la popularidad y estandarización de RDF, especialmente en el campo de los datos abiertos, existen otros muchos formatos de serialización bajo modelo RDF, como por ejemplo, [N3](#), [Turtle](#), [JSON-LD](#), [RDF/JSON](#), etc.

2.3 Formatos de datos híbridos

Tanto CSV como JSON son dos formatos de ficheros de datos muy populares especialmente indicados cuando es un humano el que realiza análisis o explotación de datos. También es cierto que JSON se ha convertido en un estándar de serialización para intercambiar datos entre aplicaciones de Internet. Además, no es necesario adaptar JSON para realizar análisis de datos puesto que la gran mayoría de lenguajes de programación, así como utilidades de análisis para usuarios finales integran herramientas para leer ficheros JSON y transformarlos en otros formatos más aptos para el análisis.

JSON es un formato, de nuevo, en texto plano, por lo que es fácilmente legible por un humano. Su estructura se basa en describir objetos utilizando para ello un par conocido como clave-valor. De esta forma, para caracterizar un conjunto de coches, se utilizarán los pares clave-valor para describir las propiedades de los diferentes coches como: color:rojo; tipo:deportivo; marca:Ferrari; etc.

Ejemplo:

Cuando queremos describir y distribuir información geográfica, cómo los elementos que se muestran sobre un mapa (ciudades, ríos, fronteras, etc.) el formato JSON se vuelve especialmente potente. El formato JSON permite a un humano identificar rápidamente los elementos del mapa a los que se hace mención a la vez que éstos son fácilmente procesables por un programa (por ejemplo, una app móvil) para mostrar información de interés sobre un mapa.

Ejemplo de formato JSON

```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [125.6, 10.1]
  },
  "properties": {
    "name": "Dinagat Islands"
  }
}
```

Por su lado el clásico formato CSV se encuentra prácticamente en la totalidad de los repositorios de datos abiertos, ya que es un formato rápidamente explorable una vez descargado del repositorio. Cualquier editor de textos, por sencillo que sea, permite su

apertura y formateado, permitiendo así al usuario final acceder a los datos de su interior sin gran esfuerzo.

Ejemplo: la Biblioteca Británica pone a disposición un paquete comprimido **ZIP** con todos los metadatos relacionados con la obra *Alicia en el País de las Maravillas* de Lewis Carroll. **El paquete** consta de 5 ficheros de datos, todos ellos en formato **CSV**, en los que se enumeran todos los títulos, nombres, temas, etc. relacionados con la obra.

En resumen, podemos decir que los formatos de datos más populares para el intercambio de datos en internet son:



2.4 Otros formatos de datos

Como se ha comentado previamente, los formatos de datos repasados hasta el momento en esta sección son los más populares del momento para el intercambio de datos en internet. Sin embargo, existen multitud de formatos diferentes cuyo uso está normalmente reservado para ámbitos más científicos o técnicos. No es el propósito de este informe,

repasar de forma exhaustiva todos los formatos existentes, pero sería conveniente que el lector se conociera algunos de esos formatos más específicos.

La importancia del sector del diseño y la fabricación de productos en la actualidad es innegable. Con una sociedad cada vez más orientada al consumo, el número de nuevos productos en el mercado es cada vez mayor. Por ello, en esta sección, introducimos algunos formatos de datos específicos que se utilizan en la ingeniería de producto. Otro campo de importancia fundamental en la actualidad es la simulación científica. Desde las predicciones meteorológicas hasta la investigación en medicina y farmacología, se valen de simulaciones y cálculos científicos para anticipar desastres naturales y el desarrollo de nuevos fármacos. Al igual que en el caso de la ingeniería de producto, este campo de la investigación y ciencia utiliza sus propios formatos de datos optimizados para este propósito. A continuación describimos brevemente sus características más diferenciales.

2.4.1 Formatos de datos para ingeniería de producto

Para el sector de la ingeniería de producto (barcos, aviones, coches, etc.) la fase de diseño de ingeniería es vital a lo largo de todo el ciclo de vida del producto (PLM). Existen grandes productos software para el diseño de ingeniería tales como (AutoCAD®, CATIA®; SOLIDWORKS®, NX®, etc.) Todos ellos utilizan sus propios formatos de datos donde se almacenan, tanto las geometrías 3D como las entidades y relaciones asociadas a esas geometrías. Ni que decir tiene que estos son formatos de datos muy complejos, únicamente interpretables por máquinas. Para facilitar la interoperabilidad entre estas soluciones en el mercado (y permitir que una misma empresa que trabaja con dos soluciones diferentes pueda reutilizar sus diseños) existen algunos formatos estándar para el intercambio de diseños³. Este es el caso del formato STEP, que se ampara bajo una norma ISO por propósitos de estandarización.

³ Existen otros formatos e intercambio como el IGES.

2.4.2 Formatos de datos en simulación científica

Otro campo donde los formatos de datos son de vital importancia es el de la simulación científica. Por ejemplo, la predicción meteorológica es una de las disciplinas donde mayor cantidad de datos se generan. Los datos que producen los modelos de simulación son, además, muy variados, incluyendo geografía, topología, variables discretas escalares (como temperatura, presión, etc.) y variables complejas vectoriales (como vectores y líneas de viento). Dentro de este grupo se encuentran los formatos de referencia como [netCDF](#), [HDF](#) y [GRIB](#). Prácticamente, todas las agencias y organismos públicos de investigación ponen a disposición del público general datos en estos formatos estándar.

Todos estos formatos de datos más específicos son más difíciles de encontrar en los repositorios de datos abiertos, aunque hay excepciones. Por ejemplo, en [AEMET](#), encontramos datos diarios de las series históricas de precipitación (también temperatura y otras variables) en formato netCDF. La [NASA](#) dispone información similar, además de algunas utilidades de software para visualizar los datos contenidos en estos ficheros. La complejidad de procesar estos ficheros de datos es tal, que aquellos usuarios no avanzados requieren de estas utilidades para la simple apertura y visualización de estos datos.

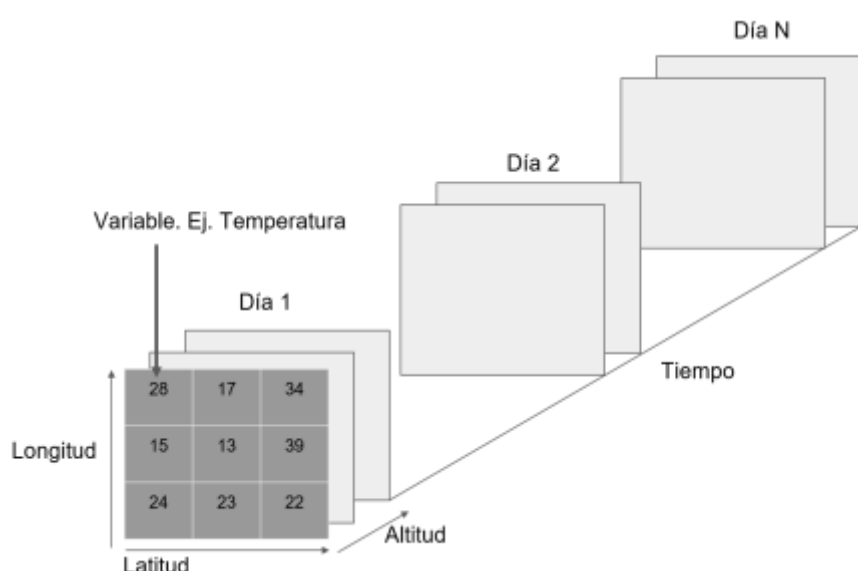
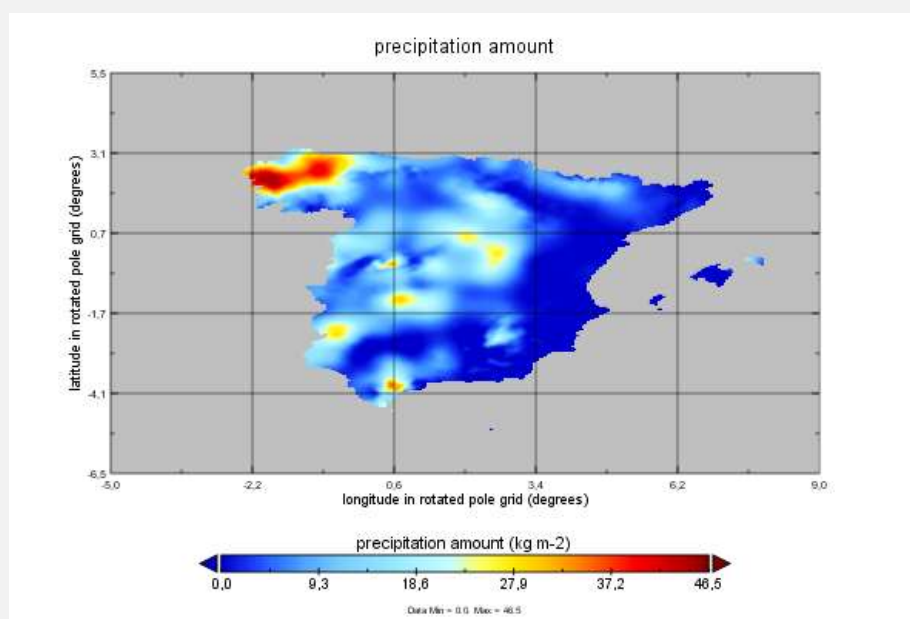


Diagrama de estructura simplificada de un fichero de datos en formato netCDF.

Ejemplo: datos de precipitación para la Península Ibérica y Baleares. Visualización generada desde un fichero de datos tipo netCDF extraído de AEMET. La aplicación utilizada para abrir y representar gráficamente es Panoply. Una utilidad gratuita descargable desde el [sitio web](#) de la NASA.



Ahora que ya conocemos los formatos de datos más populares tanto para el consumo directo por humanos como aquellos indicados para ser procesados directamente por máquinas, debemos entender cómo se propagan y distribuyen dichos datos. A continuación, entenderemos cómo extraer y consultar datos de interés almacenados en repositorios de datos abiertos.

3. EXTRACCIÓN Y CONSULTA DE DATOS ABIERTOS.

De forma natural, podemos preguntarnos cómo los organismos encargados de gestionar los repositorios de datos intercambian y enriquecen sus registros de recursos. Aquí es donde las tecnologías digitales de Internet cobran un protagonismo absoluto. Pero antes de entrar de

llo en tecnología, nos vamos a plantear la siguiente pregunta: ¿cómo extraer y clasificar todo el conocimiento encerrado en antiguos documentos dedicados a la medicina natural a lo largo del mundo? Antes de analizar el contenido de esas obras es necesario localizarlas a lo largo de los diferentes repositorios en el mundo, clasificarlas, resumirlas y unificarlas en algo que se denomina una colección. Sin embargo, aunque pudiéramos disponer de una colección así, sería un trabajo inabordable para un humano que no contara con las herramientas digitales necesarias. No obstante, para alguien que cuente con la ayuda de las **API (Application Programming Interfaces)** y las tecnologías de la **Web Semántica**, el desafío es alto pero asumible.

En esencia, una **API define el “idioma universal” que habla un sistema software en Internet**. Como un idioma, éste puede usarse para hacer preguntas complejas y masivas y comprender sus respuestas. Y como un idioma, con un poco de práctica, puede ser una herramienta poderosa y extensible para la comunicación entre máquinas y en un futuro entre máquinas y humanos. Una de las tecnologías más usadas para construir API se denomina **REST (Representational State Transfer)**. Sin entrar en demasiados detalles técnicos, la tecnología REST define una serie de métodos de tal forma que un cliente (el ordenador que hace una consulta) envía una petición - utilizando alguno de estos métodos - a un servidor remoto en Internet y éste le contesta con la información disponible.

Las API (los idiomas) ayudan a crear una capa lógica de servicios sobre la Web (la comunicación). Es decir, una vez que conseguimos hablar el mismo idioma, podemos entablar una conversación y con un poco de práctica negociar e intercambiar bienes (en Internet, datos). Sin embargo, en Internet también hay reglas que marcan cómo han de intercambiarse esos datos, para preservar y respetar la propiedad y autoría del autor de esos datos en origen.

3.1 Licencias y buenas prácticas de uso.

En esta sección introducimos brevemente el concepto de licencia como herramienta útil para conocer nuestros derechos y obligaciones cuando publicamos e intercambiamos

datos en Internet. El alcance de este informe no contempla una revisión exhaustiva de los diferentes tipos de licencias y sus términos particulares. Sin duda, abarcar de forma amplia esta materia daría lugar a la realización de un monográfico extenso⁴.

Cada vez más, la propiedad intelectual de activos inmateriales, como es el caso de las obras digitales (al fin y al cabo, datos), tiene una mayor presencia y relevancia en nuestra sociedad digital. Casi cualquier proceso de creación digital comienza construyendo sobre activos ya existentes, ya sean repositorios de código abierto, de conjuntos de datos abiertos o bibliotecas de recursos web abiertos y reutilizables.

Una mala práctica reconocida en el uso de Internet es la reutilización de todo tipo de recursos (imágenes, documentos, videos, audios, etc.) accesibles a través de la Web⁵ sin aceptar (ni tan siquiera revisar) los términos de la licencia bajo la cual se ponen a disposición de los usuarios esos recursos.

Fundamentalmente las licencias establecen tres tipos de condiciones de uso sobre los datos a los que afectan. Estas condiciones son:



Obligación. Las licencias establecen que obligaciones ha de cumplir la persona⁶ que utilice esos datos. Por ejemplo, la mayoría de las licencias, obligan a mantener un aviso visible sobre el tipo de licencia que afecta a esos datos así como a atribuir la autoría del conjunto de datos a su autor original.

⁴ Para obtener una información más específica sobre licencias de uso y reutilización de datos el lector puede consultar [el portal de datos abiertos de la Unión Europea](#) donde encontrará diferente material didáctico sobre este tema.

⁵ Formalmente WWW (World Wide Web).

⁶ Hablamos de persona por sencillez, sin embargo, las licencias aplican igual a una persona física o jurídica, así como a los sistemas software que utilicen los datos.



Permiso. Las licencias establecen qué permisos otorgan a la persona que utiliza esos datos. Por ejemplo, la licencia puede establecer que esos datos son redistribuibles o incluso, la licencia puede permitir que se conceda la patente a la persona que utiliza dichos datos.



Prohibición. Al igual que con los permisos, las licencias pueden establecer explícitamente qué prohibiciones imponen a la persona que utiliza esos datos. Un caso común es el de aquellas licencias que prohíben hacer un uso comercial de los datos que protegen.



3.2 Ejemplo de utilización de API para extracción y análisis de datos.

Veamos el siguiente ejemplo. Supongamos que estamos muy interesados en un tema muy candente en la actualidad: *el bitcoin*, y nos preguntamos cuántos y cuáles son los títulos registrados en la Biblioteca Digital Pública de America (DPLA).

Dado que la DPLA expone un API público para consulta, puedo ejecutar la llamada https://api.dp.la/v2/items?q='bitcoin'&api_key=e0b949s23d6de3c1af96433f88e2 donde específico el término de búsqueda (**q='bitcoin'**) y el API me devolverá en menos de un

segundo la información de la [tabla 1](#). Dada la relativa novedad del término⁷, solamente aparecen 12 títulos en la búsqueda. En la sección [herramientas y tecnologías para el análisis de datos abiertos](#) se explica con detalle la forma de utilizar la API para obtener estos resultados. Por motivos de claridad, los resultados finales que se muestran aquí solamente contienen el título de los libros y su fecha de publicación.

La realidad es que la API nos devuelve un conjunto vasto de información que caracteriza completamente a cada uno de los títulos. La estructura que retorna la API está de acuerdo con el modelo de datos (RDF) definido por la DPLA⁸. Técnicamente, el fichero que devuelve la API, que contiene la información de cada título, [está serializado en JSON-LD](#).

Tabla 1. Obras que contienen la palabra Bitcoin en su título

item	date
<i>Bitcoin basics</i>	2018
<i>Bitcoin Bling Necklace, United States, 2018</i>	2018
<i>Regulation of Bitcoin in selected jurisdictions /</i>	2014
<i>Forecasting exchange rate volatility with high-frequency Bitcoin data : Is digital currency really that different?</i>	2014-05
<i>Forecasting exchange rate volatility with high-frequency Bitcoin data : Is digital currency really that different?</i>	2014-05

⁷ El término se acuña por primera vez en el año 2008 en el misterioso artículo publicado por el autor cuyo pseudónimo es Satoshi Nakamoto Nakamoto, Satoshi (31 October 2008). "[Bitcoin: A Peer-to-Peer Electronic Cash System](#)" (PDF). bitcoin.org. [Archived](#) (PDF) from the original on 20 March 2014. Retrieved 28 April 2014.

⁸ Para más información se recomienda consultar [la documentación de la API de la DPLA](#). La DPLA mantiene su repositorio de datos abiertos ampliamente documentado. De igual forma ocurre con el resto de las organizaciones que preservan patrimonio, prueba de la importancia que tiene el uso de estas tecnologías en este tipo de organización.

<i>Forecasting exchange rate volatility with high-frequency Bitcoin data : is digital currency really that different?</i>	2014-05
<i>Bitcoin : examining the benefits and risks for small business : hearing before the Committee on Small Business, United States House of Representatives, One Hundred Thirteenth Congress, second session, hearing held April 2, 2014</i>	2014
<i>Utah Law Review 2016 Number 2</i>	2016-03-01
<i>Regulation of cryptocurrency around the world</i>	2018
<i>Regulation of cryptocurrency in selected jurisdictions</i>	2018
<i>The disrupter series : digital currency and blockchain technology : hearing before the Subcommittee on Commerce, Manufacturing, and Trade of the Committee on Energy and Commerce, House of Representatives, One Hundred Fourteenth Congress, second session, March 16, 2016</i>	2016
<i>Beyond bitcoin : emerging applications for blockchain technology : joint hearing before the Subcommittee on Oversight & Subcommittee on Research and Technology, Committee on Science, Space, and Technology, House of Representatives, One Hundred Fifteenth Congress, second session, February 14, 2018</i>	2018

4. HERRAMIENTAS Y TECNOLOGÍAS PARA EL ANÁLISIS DE DATOS ABIERTOS.

En la sección anterior, [3.1 Ejemplo de utilización de API para extracción y análisis de datos](#), se introdujo la herramienta tecnológica por excelencia para compartir y distribuir datos abiertos a través de la Web. Las API son poderosas armas para expandir el conocimiento digital que durante años ha vivido en silos aislados repartidos por Internet. Como la mayoría de las herramientas en Internet, existen formas muy diferentes de usarlas dependiendo del tipo de usuario y la aplicación buscada. Las diferentes formas de utilizar una API se pueden denominar implementaciones de las API. Por ejemplo, un usuario con competencias digitales intermedias en Internet puede hacer uso de una API a través de una implementación en consola de esa API. Esto es, una herramienta en forma de pantalla en una página web que nos permite utilizar la API rellenando campos en un formulario y efectuar la consulta en el servidor remoto presionando un botón para ejecutar la llamada a la API.

4.1. Usuarios intermedios. Ejemplos de uso de algunas API.

4.1.1. AEMET

Por ejemplo, en el siguiente enlace: <https://opendata.aemet.es/centrodedescargas/productosAEMET?> se presenta una aplicación web muy sencilla que permite consultar todos los datos de la Agencia Estatal de Meteorología (AEMET.) Estos datos están disponibles para cualquier usuario a través de la [página web estándar de AEMET](#). Sin embargo, localizar conjuntos de datos particulares puede ser tedioso y exigir muchos pasos en la navegación de la página web. Además, si se desea automatizar el proceso para obtener determinados datos periódicamente, el uso de la API es la mejor opción.

El conjunto de avisos meteorológicos adversos se obtiene haciendo uso de la siguiente llamada⁹:

`https://opendata.aemet.es/opendata/api/avisos_cap/ultimoelaborado/area/esp?api_key=123456789`

La respuesta del servidor es la siguiente:

```
{
  "descripcion" : "exito",
  "estado" : 200,
```

⁹ La mayoría de las API de servicios de referencia para consulta y extracción de datos implementan opciones de seguridad y privacidad. En algunos casos, se exigirá al usuario que se registre para obtener un nombre de usuario y contraseña como método de autenticación. En otros, tras el registro (habitualmente basta con un email válido) el usuario obtiene una clave con la que puede efectuar llamadas a la API. Esta clave se conoce como el `api_key` y ha de incluirse en todas las llamadas a la API. Por motivos de seguridad las `api_key` que aparecen en este documento son ficticias y no tienen validez.

```
"datos" :
  "https://opendata.aemet.es/opendata/sh/Z_CAP_C_LEMM_2018083014514
  2_AFAE",
  "metadatos" : "https://opendata.aemet.es/opendata/sh/738bbbfd"
}
```

Este fichero JSON (devuelto por la API) incluye dos enlaces a la información relevante.

- "datos" :
["https://opendata.aemet.es/opendata/sh/Z_CAP_C_LEMM_20180830145142_AFAE"](https://opendata.aemet.es/opendata/sh/Z_CAP_C_LEMM_20180830145142_AFAE)
- "metadatos" : "https://opendata.aemet.es/opendata/sh/738bbbfd"

Uno de ellos apunta directamente a los datos que se desea consultar. El otro, apunta a la información relativa a los metadatos asociados a los datos mismos.

La ejecución de esta llamada a la API de AEMET (a través de la consola web) puede hacerla cualquier usuario con unos conocimientos básicos y está pensada para un uso puntual de la API o bien para comprobar su correcto funcionamiento. Por ello, los enlaces anteriores dejarán de estar vigentes en un periodo de tiempo corto.

Ahora bien, si el uso de este API tiene por objetivo capturar estos u otros datos disponibles, de forma repetida, bien para su explotación y análisis o bien para ser mostrados en otra aplicación (por ejemplo, una aplicación móvil), la API ha de usarse de otra manera (bajo otra implementación). Aquí es donde entran en juego los lenguajes de programación y las librerías que implementan las API para estos lenguajes. Ampliaremos esta aproximación en la sección *Lenguajes de programación*.

4.1.2 EUROPEANA

Por ejemplo, volviendo al caso de la sección *Interoperabilidad entre organismos a través de los datos* para recuperar datos de la Biblioteca Pública Digital de América podemos utilizar un método (consola) similar al mostrado anteriormente para AEMET. En este caso, hemos cambiado la fuente para utilizar *la consola* de la plataforma Europea.eu.

```
{
  "apikey": "xxxxxxxx",
  "success": true,
  "requestNumber": 999,
  "itemsCount": 12,
  "totalResults": 25,
  "items": [
    {
      "id": "/0940436/_nnkRXVb",
      "completeness": 8,
      "country": [
        "poland"
      ]
    }
    .....
  ]
}
```

Retornamos 12 registros de un total de 25 obras que contienen el término bitcoin en su título. Por motivos de claridad el contenido total del fichero JSON que retorna los resultados se ha omitido. La llamada puede reproducirse si se cuenta con un `apy_key` válido efectuando la siguiente petición:

```
https://www.europeana.eu/api/v2/search.json?wskey=xxxxxxx&query=bitcoin&rows=100
```

Antes de alcanzar un mayor nivel de profundidad en el uso de las API mediante una introducción a los lenguajes de programación (sección [Lenguajes de programación](#)), merece la pena destacar una herramienta ampliamente usada entre la comunidad de desarrolladores en Internet para comprobar el funcionamiento de las API. [Postman](#) es una herramienta muy sencilla pero extremadamente potente que nos permite construir nuestras llamadas a las API mediante un interfaz agradable e intuitivo. Existen versiones gratuitas de este software y otras bajo suscripción que incrementan la funcionalidad. Buena parte de los ejemplos expuestos en este documento se pueden reproducir fácilmente con Postman. Como

material adicional a este documento se incluyen unas colecciones con los ejemplos aquí desarrollados que pueden importarse en Postman de forma sencilla siguiendo [este tutorial](#).

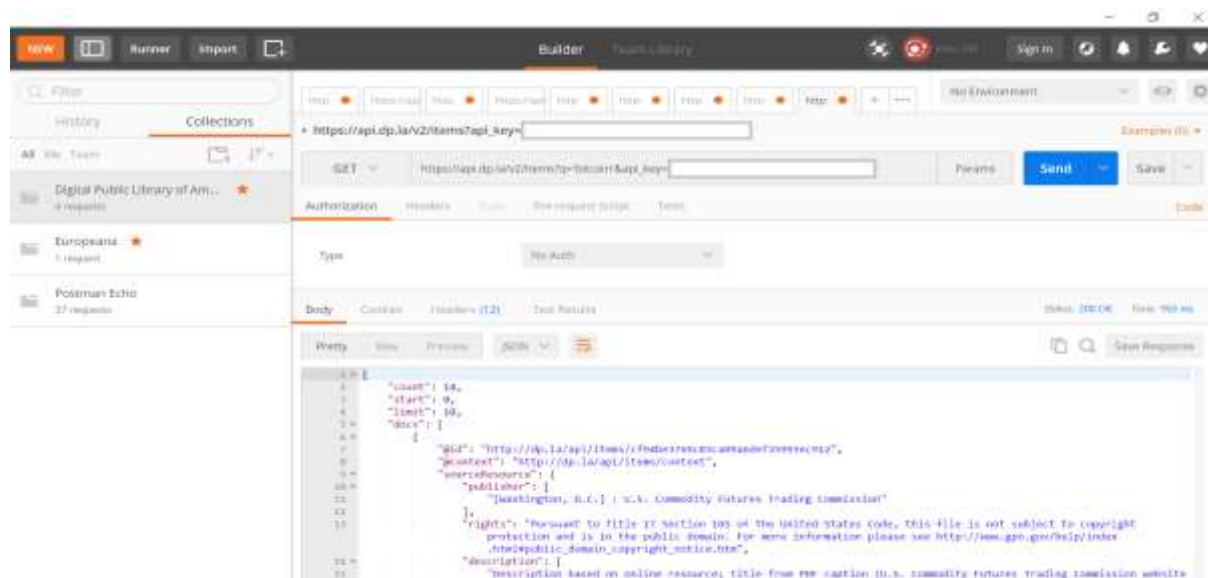


Imagen 4. Interfaz de Postman.

4.1.3 The Sports DB

Uno de los sectores que se digitalizan a mayor velocidad es el sector de los deportes. Por un lado, los deportes tradicionales se están convirtiendo en uno de los mayores generadores de datos digitales del momento. Las principales competiciones de deportes de masas tales la **NBA** en Baloncesto o la **NFL** en fútbol norteamericano, **La Liga** de fútbol española o **F1** a nivel mundial, invierten cantidades ingentes de dinero en sistemas y software para la recolección y explotación masiva de datos sobre jugadores, encuentros, estadísticas, etc.

Obviamente, existen empresas cuyos modelos de negocio se asientan a lo largo de toda la cadena de valor de este tipo de datos. Desde aquellas que habilitan con sus equipos técnicos (cámaras, sistemas de seguimiento, sistemas de accesos, etc.) la generación del dato, hasta los fabricantes de software que permite explotar y extraer decisiones sobre aquel jugador que tiene mayor probabilidad de anotar, lesionarse, etc.

Un ejemplo sencillo de utilización de estas API sería el siguiente. En el sitio web [The Sports DB](#) ponen a disposición de los usuarios un API gratuita para pruebas y otra para entornos de producción. Para demostrar la capacidad de este API, obtenemos la plantilla actual del “Real Madrid” con algunos datos relevantes¹⁰. La API proporciona hasta 48 parámetros o datos de cada jugador.

Con tan solo ejecutar la petición con método GET desde la interfaz de POSTMAN, vemos cómo se retorna el resultado de la lista de jugadores del Real Madrid caracterizados con 48 variables como por ejemplo, nombre, edad, altura, fecha de inicio del contrato, etc.

```
{
  "player": [
    {
      "idPlayer": "34145514",
      "idTeam": "133738",
      "idSoccerXML": "154",
      "idPlayerManager": "18026122",
      "strNationality": "Belgium",
      "strPlayer": "Thibaut Courtois",
      "strTeam": "Real Madrid",
      "strSport": "Soccer",
      "intSoccerXMLTeamID": "15",
      "dateBorn": "1992-05-11",
      "dateSigned": "2011-06-03",
      "strSigning": "35,00 Mill. €",
      "strWage": "",
      "strBirthLocation": "Bree, Belgium",
      "strDescriptionEN": "Thibaut Nicolas Marc Courtois (Dutch: ; French: ; born 11 May 1992) is a Belgian professional footballer who plays as a goalkeeper for La Liga club Real Madrid and the Belgium national team.\r\n\r\nCourtois graduated through the youth system of Genk, and aged 18, he played a key role in the team's Belgian Pro League victory. In July 2011 he joined Chelsea for a reported £8 million, and was immediately loaned to Atlético Madrid. In three seasons there, he won the Europa League in 2012, the Copa del Rey in 2013 and the La Liga title in 2014. He also won the Ricardo Zamora Trophy for best goalkeeper in La Liga, for his performances in his two latter seasons. Courtois returned to Chelsea in July 2014, and in his first season he helped them win the League Cup and the Premier League title.\r\n\r\nCourtois made his senior international debut in October 2011, becoming the youngest goalkeeper to represent Belgium. He has since earned over 50 caps and appeared at the 2014 FIFA World Cup, UEFA Euro 2016 and the 2018 FIFA World Cup where he was awarded the Golden Glove as best goalkeeper of the tournament.",
```

¹⁰ En el API de pruebas, el API Key es un número genérico para todos los usuarios, en este caso el “1”.


```

    "strDescriptionDE": null,
    "strDescriptionFR": null,
    "strDescriptionCN": null,
    "strDescriptionIT": null,
    "strDescriptionJP": null,
    "strDescriptionRU": null,
    "strDescriptionES": null,
    "strDescriptionPT": null,
    "strDescriptionSE": null,
    "strDescriptionNL": null,
    "strDescriptionHU": null,
    "strDescriptionNO": null,
    "strDescriptionIL": null,
    "strDescriptionPL": null,
    "strGender": "Male",
    "strPosition": "Goalkeeper",
    "strCollege": null,
    "strFacebook": "",
    "strWebsite": "",
    "strTwitter": "",
    "strInstagram": "",
    "strYoutube": "",
    "strHeight": "1.99",
    "strWeight": "",
    "intLoved": "1",
    "strThumb":
    "https://www.thesportsdb.com/images/media/player/thumb/b9cy1x1533894579.jpg",
    "strCutout":
    "https://www.thesportsdb.com/images/media/player/cutout/18026122.png",
    "strBanner": null,
    "strFanart1":
    "https://www.thesportsdb.com/images/media/player/fanart/yxxtpy1431626262.jpg",
    "strFanart2":
    "https://www.thesportsdb.com/images/media/player/fanart/uvuryx1431626272.jpg",
    "strFanart3":
    "https://www.thesportsdb.com/images/media/player/fanart/uvwuup1431626285.jpg",
    "strFanart4":
    "https://www.thesportsdb.com/images/media/player/fanart/xxusuw1431626297.jpg",
    "strLocked": "unlocked"
  },

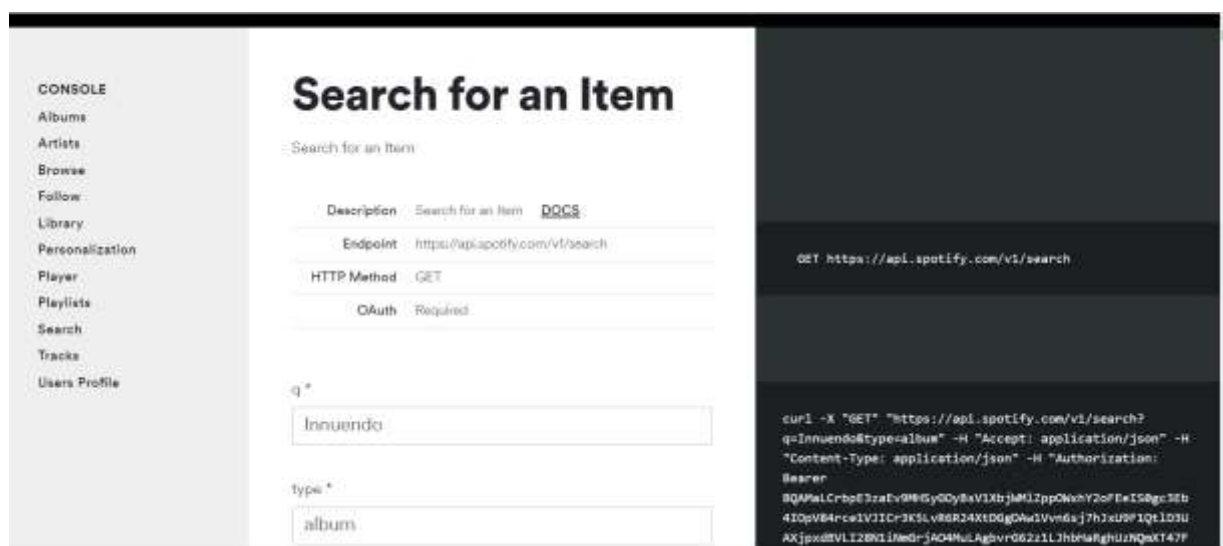
```

4.1.4 Spotify

El servicio de música en streaming más conocido del mundo dispone, como no, de su propia API. El objetivo de esta API es permitir a los desarrolladores construir aplicaciones que consuman datos del servicio. *Spotify* dispone de más de 30 millones de canciones en su

catálogo e incontable listas de reproducción de sus usuarios. Dados los objetivos de promoción comercial de esta API, su documentación es muy extensa y accesible. Existen multitud de proyectos e ideas que poner en práctica haciendo uso de la API de este servicio. Por simplicidad, vamos a demostrar la potencia de esta API haciendo uso de su propia consola web para retornar algunos resultados¹¹.

Para acceder a la consola del servicio solo necesitamos acceder a la siguiente URL <https://developer.spotify.com/console>. La URL base de la API es <https://api.spotify.com>. A partir de esta URL base, la API cuenta con varios endpoints para realizar peticiones dependiendo de la categoría en la que se pretende buscar (artistas, álbumes, canciones, playlists, etc.). En este ejemplo, vamos a realizar una búsqueda muy sencilla, sin ninguna categoría en especial, utilizando el endpoint de búsqueda de items en general.



The screenshot shows the Spotify API console interface. On the left is a sidebar with navigation links: CONSOLE, Albums, Artists, Browse, Follow, Library, Personalization, Player, Playlists, Search, Tracks, and Users Profile. The main area is titled 'Search for an Item' and contains a form with the following fields:

- Description:** Search for an item
- Endpoint:** `https://api.spotify.com/v1/search`
- HTTP Method:** GET
- OAuth:** Required
- q*:** Innuendo
- type*:** album

On the right side of the console, the following cURL command is displayed:

```
curl -X "GET" "https://api.spotify.com/v1/search?q=Innuendo&type=album" -H "Accept: application/json" -H "Content-Type: application/json" -H "Authorization: Bearer BQAPwLCrbpE3ratv9M6Gy0Cy8aV1XbJhM1ZpQWnY2oF8eIS0gc3Eb4IQpVB4rce1V1ICr3KSLv86R34X1DGdQWd1Vvmdaj7h1xU9P1qt103UAXjpxdV1L2BN1fmedr-jA04MuLAgbvr0G2z1LJhblwghUzNQxT47P"
```

Ejemplo de la consola web del API de Spotify.

Sin más, ejecutamos la consulta o llamada a este endpoint con los siguientes parámetros:

- **Término a buscar:** "Innuendo"
- **Tipo:** album

¹¹ Para poder utilizar la API del servicio es necesario registrarse (o iniciar sesión) como usuario y seguir algunos sencillos pasos de configuración para obtener un ID o Token con el que poder realizar solicitudes.

- Mercado: Español
- Máximo número de resultados: 1
- Offset: 1

La API nos devuelve el resultado de forma inmediata serializado en formato JSON con el siguiente contenido:

```
{
  "albums": {
    "href":
      "https://api.spotify.com/v1/search?query=Innuendo&type=album&market=ES&offset=1&limit=1",
    "items": [
      {
        "album_type": "album",
        "artists": [
          {
            "external_urls": {
              "spotify": "https://open.spotify.com/artist/1dfeR4HaWDbWqFHLkxsg1d"
            },
            "href": "https://api.spotify.com/v1/artists/1dfeR4HaWDbWqFHLkxsg1d",
            "id": "1dfeR4HaWDbWqFHLkxsg1d",
            "name": "Queen",
            "type": "artist",
            "uri": "spotify:artist:1dfeR4HaWDbWqFHLkxsg1d"
          }
        ],
        "external_urls": {
          "spotify": "https://open.spotify.com/album/6b7VmKVacoa7bbIHw1KfK7"
        },
        "href": "https://api.spotify.com/v1/albums/6b7VmKVacoa7bbIHw1KfK7",
        "id": "6b7VmKVacoa7bbIHw1KfK7",
        "images": [
          {
            "height": 640,
            "url": "https://i.scdn.co/image/7421ed7ea55607617c767ce311783b1a8603d860",
            "width": 640
          },
          {
            "height": 300,
            "url": "https://i.scdn.co/image/8799d5eb4aac15498ad7d389c4ff45952db0bd3c",
            "width": 300
          },
          {
            "height": 64,
            "url": "https://i.scdn.co/image/c679645fb6a698b3d9b8bd764afc3e270dd30ff0",
            "width": 64
          }
        ]
      }
    ]
  }
}
```

```

    }
  ],
  "name": "Innuendo (Deluxe Edition 2011 Remaster)",
  "release_date": "1991-02-05",
  "release_date_precision": "day",
  "total_tracks": 17,
  "type": "album",
  "uri": "spotify:album:6b7VmKVacoa7bbIHw1KfK7"
}
],
"limit": 1,
"next":
"https://api.spotify.com/v1/search?query=Innuendo&type=album&market=ES&offset=2&limit=1",
"offset": 1,
"previous":
"https://api.spotify.com/v1/search?query=Innuendo&type=album&market=ES&offset=0&limit=1",
"total": 84
}
}

```

Tras el resultado vemos que la información retornada es bastante completa pese a haber introducido tan solo el título del álbum. Vemos que Innuendo corresponde con un álbum de la banda de rock Queen, publicado el 5 de febrero de 1991 y que contenía un total de 17 canciones. La potencia de los datos abiertos enlazados se encuentra una vez más en la combinación de resultados de forma agregada para retornar mayor y más variada información.

En este sentido, de la anterior consulta, obtenemos el id del álbum en concreto **6b7VmKVacoa7bbIHw1KfK7**. De ahí, podemos utilizar el endpoint de álbumes de la API <https://api.spotify.com/v1/albums/{id}> para obtener más información, por ejemplo, la foto de la portada del álbum.



Foto de la portada del álbum Innuendo de Queen publicado en 1991

4.2 Usuarios avanzados. Uso de API mediante lenguajes de programación.

Las consolas para explotar API nos proporcionan rapidez y sencillez de uso. Sin embargo, la máxima flexibilidad y versatilidad la obtendremos al utilizar un lenguaje de programación con el que podremos, no solo automatizar la búsqueda, sino además postprocesar el resultado que nos devuelva la API de la forma en que queramos. Así, una vez retornado el resultado de la búsqueda podremos almacenar, mostrar y enviar esos datos a múltiples destinos. Veamos a continuación cómo hacer uso de algunos lenguajes de programación populares para realizar algunas consultas a las API ya introducidas anteriormente.

5.2.1 Python

En la sección 3. [Extracción y consulta de datos abiertos](#), introdujimos REST como una de las tecnologías más populares en el universo de las API. En el ejemplo de la subsección 3.1 vimos cómo consultar los títulos registrados en la Biblioteca Digital Pública de América que contienen el término “bitcoin”. Veamos ahora cómo realizar la misma llamada mediante el lenguaje de programación [Python](#) y una de las muchas librerías que implementan la tecnología API REST¹² como [Requests](#). Para obtener el mismo resultado que con la llamada a través de la consola tan solo tenemos que escribir las siguientes 6 líneas de código en un entorno de programación preparado para Python:

```
pip install requests
import requests
import json
```

¹² La práctica totalidad de los lenguajes de programación disponen de librerías que implementan la tecnología REST.

```
payload = {'wskey': 'xxxxxx', 'query':'bitcoin'}
r =
requests.get('https://www.europeana.eu/api/v2/search.json',
params=payload)
r.json()
```

El servidor responde a la búsqueda con 25 resultados serializados en formato JSON¹³ con todos los campos referentes a la información que caracteriza a cada una de las obras:

```
{
  "apikey":"xxxxxxx",
  "success":true,
  "items":[
    {
      "timestamp_created_epoch":1512041002629,
      "timestamp_update_epoch":1512041002629,
      "dcLanguageLangAware":{
        "def":[
          "pl"
        ]
      },
      "previewNoDistribute":false,
      "dcTitleLangAware":{
        "def":[
          "Bitcoin a definicja i funkcje pieni\u0105dza"
        ]
      },
      "europeanaCompleteness":8,
      "europeanaCollectionName":[
        "0940436_Ag_PL_dlibra.umcs.lublin.pl"
      ],
      "year":[
```

¹³ De nuevo el fichero JSON retornado se corta por motivos de claridad.


```

    "2014"
  ],
  "dataProvider":[
    "Biblioteka Cyfrowa UMCS"
  ]
}

```

4.2.2 R

Python es un lenguaje de programación de obligada cita cuando se trata el tema de *lenguajes de programación orientados al análisis de datos*. Python cumple perfectamente su cometido como lenguaje versátil de propósito general y muy buenas condiciones para el análisis de datos. Sin embargo, un lenguaje de programación mucho más específico para el análisis de datos y el aprendizaje automático es *R*. *R* también tiene poderosas armas para acceder a datos abiertos a través de las API y es una de las lenguas francas en el mundo de la ciencia de datos y la inteligencia artificial.

El ejemplo de la sección 3.1 donde se retornaba la tabla correspondiente a los títulos (disponibles en la DPLA) que contienen la palabra bitcoin, se obtuvo, en realidad, haciendo uso de una implementación de API REST para *R*. Veamos el ejemplo, ahora con mayor grado de detalle. De igual forma que en Python necesitamos hacer uso de las librerías *requests* y *json*, en *R* (las librerías se denominan paquetes) necesitamos hacer uso de dos paquetes similares. Para manejar la petición a la API de la DPLA necesitamos *httr* y para controlar los ficheros JSON que el servidor nos devuelve cuando efectuamos la búsqueda necesitamos *jsonlite*. Para ello le pedimos al sistema (entorno de programación en *R*) que haga uso de esas dos librerías:

```

library(httr)
library(jsonlite)

```

En caso de no estar instaladas previamente en el entorno de programación se instalarán mediante:

```
install.packages(c("httr", "jsonlite"), dep = true)
```

Una vez tenemos el entorno listo, solo tenemos que escribir este sencillo programa para obtener los títulos de la DPLA que contienen la palabra bitcoin.

```
# Save API key as a variable
apikey <- 'xxxxxxxxxxxxxxxxxxxxxxxx'
# Save base endpoint as variable
url_api <- 'https://api.dp.la/v2/'
# Construct API request
resource <- "items?" #don't forget to add ? character after the
resource
keyword <- 'q="bitcoin"'
call <- paste0(url_api,resource, "q=", "'bitcoin'", "&",
"api_key=",apikey)
recall <- GET(url = call)
recall_content <- content(recall)
```

Finalmente retornamos el fichero JSON-LD que contiene toda la información de las obras encontradas. Dado que, la información que caracteriza a todas y cada una de las obras es extensa e intrincada (ver el modelo de datos de la DPLA [aquí](#)), con el siguiente programa extraemos el contenido que nos interesa relativo a los títulos de las obras y su fecha de publicación:

```
# Apply function across all list elements to extract the name and
address of each repo
repo_df <- lapply(recall_content$docs, function(x) {
```

```
df <- data_frame(item      = x$sourceResource$title,
                 date      = x$sourceResource$date[[1]],
                 #publisher =
unlist(x$sourceResource$contributor))
}) #>% bind_rows()
finaldf <- data.frame(Reduce(rbind, repo_df))
```

El objeto `finaldf` contiene la información de la tabla *Tabla 1. Obras que contienen la palabra Bitcoin en su título*. Fijémonos ahora en la potencia de un lenguaje de programación combinado con un potente API REST para datos abiertos. Además del título y la fecha de publicación, nos gustaría obtener un enlace directo a la publicación allí donde se encuentre. En algunos casos, este enlace nos llevará directamente a la obra en caso de que su licencia permita la descarga sin restricciones. En otros casos, el sitio web que aloje la obra nos indicará cómo proceder si deseamos descargar dicha obra. Sea como fuere, añadiendo una simple línea a nuestro programa en R retornamos una nueva tabla con la fuente de origen del documento. Veamos la sección del código donde debemos introducir esa línea adicional (`Source = x$isShownAt`)

```
# Apply function across all list elements to extract the name and
# address of each repo
repo_df <- lapply(recall_content$docs, function(x) {
  df <- data_frame(Item      = x$sourceResource$title,
                  Date      = x$sourceResource$date[[1]],
                  Source = x$isShownAt)
                  #publisher =
unlist(x$sourceResource$contributor))
}) #>% bind_rows()
finaldf <- data.frame(Reduce(rbind, repo_df))
```

La tabla que obtenemos a cambio es la siguiente:

Tabla 2. Obras que contienen la palabra Bitcoin en su título junto con el enlace a la fuente.

Item	Date	Source
<i>Bitcoin basics</i>	2018	http://catalog.gpo.gov/F/?func=direct&doc_number=001051611&format=999
<i>Bitcoin Bling Necklace, United States, 2018</i>	2018	http://collections.si.edu/search/results.htm?q=record_ID%3Anmah_1874521&repo=DPLA
<i>Regulation of Bitcoin in selected jurisdictions /</i>	2014	http://catalog.gpo.gov/F/?func=direct&doc_number=000938591&format=999
<i>Forecasting exchange rate volatility with high-frequency Bitcoin data: is digital currency really that different?</i>	2014-05	https://digitalcc.coloradocollege.edu/islandora/object/coccc:9783
<i>Forecasting exchange rate volatility with high-frequency Bitcoin data: is digital currency really that different?</i>	2014-05	https://digitalcc.coloradocollege.edu/islandora/object/coccc:9783
<i>Forecasting exchange rate volatility with high-frequency Bitcoin data: is digital currency really that different?</i>	2014-05	https://digitalcc.coloradocollege.edu/islandora/object/coccc:9783
<i>Bitcoin: examining the benefits and risks for small business: hearing before the Committee on Small Business, United States House of Representatives, One Hundred Thirteenth Congress, second session, hearing held April 2, 2014</i>	2014	http://catalog.gpo.gov/F/?func=direct&doc_number=000929686&format=999
<i>Utah Law Review 2016 Number 2</i>	2016-03-01	http://utah-primoprod.hosted.exlibrisgroup.com/primo_library/libweb/action/dlDisplay.do?vid=MWDL&afterPDS=true&docId=digcoll_uu_u_11uu_law_clp/1278867

<i>Regulation of cryptocurrency around the world</i>	2018	http://catalog.gpo.gov/F/?func=direct&doc_number=001062821&format=999
<i>Regulation of cryptocurrency in selected jurisdictions</i>	2018	http://catalog.gpo.gov/F/?func=direct&doc_number=001062819&format=999
<i>The disrupter series: digital currency and blockchain technology: hearing before the Subcommittee on Commerce, Manufacturing, and Trade of the Committee on Energy and Commerce, House of Representatives, One Hundred Fourteenth Congress, second session, March 16, 2016</i>	2016	http://catalog.gpo.gov/F/?func=direct&doc_number=000987452&format=999
<i>Beyond bitcoin: emerging applications for blockchain technology: joint hearing before the Subcommittee on Oversight & Subcommittee on Research and Technology, Committee on Science, Space, and Technology, House of Representatives, One Hundred Fifteenth Congress, second session, February 14, 2018</i>	2018	http://catalog.gpo.gov/F/?func=direct&doc_number=001061634&format=999

En la [sección anterior](#) veíamos un ejemplo de utilización del API pública de [The Sports DB](#) para retornar la lista de jugadores actuales del Real Madrid en formato JSON a través de la herramienta POSTMAN. En esta ocasión, con las siguientes líneas de código, se obtiene la misma información pero gracias a la versatilidad del lenguaje de programación (en este caso R) se puede transformar el JSON inicial en una [tabla](#) más legible y amigable para el usuario final.

```
#Getting Real Madrid PLayer list

call2 <-
"https://www.thesportsdb.com/api/v1/json/1/searchplayers.php?t=Real_Madrid"
recall <- GET(url = call2)
recall_content <- content(recall)

repo_df2 <- lapply(recall_content$player, function(x) {
  df <- data_frame(Player = x$strPlayer,
                    Height = x$strHeight,
                    Born = x$dateBorn,
                    Signed = as.list(x$dateSigned),
                    Amount = x$strSigning)
  #publisher =
  unlist(x$sourceResource$contributor))
}) #>% bind_rows()

finaldf2 <- data.frame(Reduce(rbind, repo_df2))
```

Tabla de jugadores actuales del Real Madrid incluyendo los datos de altura en metros, la fecha de nacimiento, la fecha de firma del contrato con el equipo y el importe de venta en millones de Euros.

Player	Height	Born	Signed	Amount
Thibaut Courtois	1.99	1992-05-11	2011-06-03	35,00 Mill. €
Toni Kroos	1.75	1990-01-04	2014-07-17	25,00 Mill. €
Raphael Varane	1.88	1993-04-25	2011-08-01	10,00 mill. €

Sergio Ramos	1.83	1986-03-30	2005-09-01	27,00 Mill. €
Marcelo	1.68	1988-05-12	2007-01-01	6,50 Mill. €
Daniel Carvajal	1.73m	1992-01-11	2013-07-01	6,50 Mill. €
Nacho Fernandez	1.79m	1990-01-18	2012-07-01	Youth
Gareth Bale	1.83	1989-07-16	2013-09-01	101,00 Mill. €
Luka Modric	1.73	1985-09-09	2012-08-27	30,00 Mill. €
Isco	1.7	1992-04-21	2013-06-27	30,00 Mill. €
Karim Benzema	1.83	1987-12-19	2009-07-10	35,00 Mill. €
Keylor Navas	1.85	1986-12-15	2014-07-17	10,00 Mill. €
Marcos Llorente	1.80	1995-01-30	2014-07-01	Youth
Casemiro	184 cm	1992-02-23	2014-08-14	6,00 Mill. €
Lucas Vazquez	1.73m	1991-07-01	2014-08-29	1,00 mill. €
Casilla	1.88	1986-10-02	2010-08-01	6,00 Mill. €
Raul de Tomas Gomez	0	1994-10-17	2013-07-01	
Sergio Aguza	0	1992-09-02	2014-09-26	

En los ejemplos anteriores hemos citado Python y R como lenguajes apropiados para consultar y explotar datos abiertos. Por supuesto existen muchos otros lenguajes de programación. Algunos de ellos son considerados lenguajes de propósito general como Java o C#. Lenguajes que van desde la creación de aplicaciones móviles hasta el desarrollo de

aplicaciones de gestión empresarial. Otros lenguajes tienen una mayor orientación al desarrollo de sitios web como PHP, JavaScript, Node.js o Ruby. Todos estos lenguajes disponen de implementaciones de API REST.

Además, algunos repositorios importantes como la DPLA o Europeana disponen de implementaciones particulares de sus API de búsqueda en estos lenguajes. Habitualmente, estas implementaciones se denominan clientes. Por ejemplo en la [sección de herramientas de Europeana Pro](#) encontramos algunos de estos clientes con sus correspondientes enlaces a repositorios de [GitHub](#).

5. CONCLUSIONES Y VISIÓN TECNOLÓGICA DE FUTURO

Las habilidades de programación siguen y seguirán siendo enormemente importantes en el futuro. Sin embargo, a medida que la tecnología avanza, la programación también se especializa y se vuelve una profesión de nicho. Cada vez hay menos espacios en la programación de aplicaciones de gestión generalistas. Por el contrario, se abren nuevos mundos donde la demanda de profesionales sufre intensos picos de necesidad que hace que se disparen los salarios y las condiciones laborales.

En un futuro no muy lejano, la inteligencia artificial será capaz de programar aplicaciones básicas en base a plantillas. En el medio plazo seremos capaz de decirle a nuestro asistente personal (Alexa, Siri, Google, Aura, etc.) que nos programe una nueva aplicación para que nuestra televisión inteligente nos muestre un panel de control para nuestro coche autónomo. Esto será posible gracias a que, por un lado, los protocolos de interoperabilidad (API, modelos de datos, protocolos de comunicación, etc.) serán estándar y por otro, se habrán programado y entrenado esos algoritmos de IA gracias a la ingente cantidad de datos generados en los próximos años.

Para ello, los jóvenes de hoy han de formarse y especializarse en tecnologías con capacidad de disrupción. Tecnologías que conectan el mundo físico con el virtual como gafas, cascos e interfaces de usuario que generen una realidad extendida. Lenguajes de programación como [Unity](#) o entornos de desarrollo de realidad aumentada como [Vuforia](#). Tecnologías abiertas que permiten la gestión de volúmenes ingentes de información con estructuras complejas como [MongoDB](#), [Apache Solr](#) o [Elasticsearch](#).

De especial importancia serán todas las tecnologías relacionadas con la automatización un futuro inminente. Automatización de la navegación por páginas web con objetivos de medir el rendimiento o detectar errores como [Selenium](#), [Scrapy](#) o [NodeRed](#) en el ámbito de las cosas conectadas. Sin olvidar algunas tecnologías importantes en el mundo de la robótica

colaborativa como *ROS*, el sistema operativo abierto para co-bots (robots colaborativos), así como tecnologías relacionadas con la aceleración y depuración de proyectos de software como *Git*, *Ansible*, *Maven*, *Redmine*, etc.

Por supuesto, algunos de los lenguajes de programación citados en este documento serán fundamentales como complemento a alguna de las anteriores tecnologías. Disponer de conocimientos básicos de programación en lenguajes como Python, R o JavaScript, serán una base imprescindible en el futuro de los jóvenes tecnólogos.

6. RECURSOS

Organismos citados

- *Organización Mundial de la Salud (OMS)*
- *Servicio Británico Nacional de Salud*
- *Agencia Estatal de Meteorología (AEMET)*
- *Observación Oceánico y Meteorológico (NOAA)*
- *Biblioteca Nacional de España*
- *Biblioteca Británica*
- *Museo de Historia Natural de Londres*
- *Europeana*
- *Europeana Pro*
- *API de Europeana*
- *Biblioteca Digital Pública de América*
- *repositorio de datos abiertos del ayuntamiento de Madrid*
- *La iniciativa estatal de datos abiertos*
- *FSF (Free Software Foundation)*

Datos y formatos

- *CSV*
- *ZIP*
- *JSON*
- *JavaScript*
- *XML*
- *RDF*
- *web semántica*
- *Modelo de datos RDF de la Biblioteca Británica*
- *N3*
- *Turtle*
- *JSON-LD*
- *API (Application Programming Interfaces)*
- *Web Services*

Licencias

- *la licencia Creative Commons CC0 1.0*
- *GNU Affero General Public License v3.0*
- *European Union Public License 1.1*

Lenguajes de programación y tecnologías

- *Python*
- *Requests*
- *lenguajes de programación orientados el análisis de datos*
- *R*
- *Unity*
- *Vuforia*
- *MongoDB*
- *Apache Solr*
- *Elasticsearch*
- *Selenium*
- *Scrapy*
- *NodeRed*
- *ROS*
- *Git*
- *Ansible*
- *Maven*
- *Redmine*

Recursos web

- *GitHub*
- *Underworld*
- *Lanes Group en Reino Unido*
- *CRAFT*