Guía de Buenas Prácticas para el Desarrollo y Mantenimiento de Software

IMPaCT

Infraestructura de Medicina de Precisión asociada a la Ciencia y la Tecnología

Programa	IMPacT: Infraestructura de Medicina de Precisión asociada a la Ciencia y la Tecnología				
Nombre Proyecto	IMPaCT-Data: Programa de Ciencia de Datos de IMPaCT				
Expediente	IMP/00019				
Duración	Enero 2021 – Diciembre 2023				
Página web	impact-data.bsc.es				
Paquete Trabajo	WP2 – INFRAESTRUCTURA COMPUTACIONAL EN LA NUBE PARA LA GESTION E INTEGRACION DE DATOS.				
Tarea	T2.2 - Adopción e implantación de protocolos para el uso de contenedores software y flujos de trabajo siguiendo las recomendaciones de ELIXIR, EOSC-Life y GA4GH				
Entregable	E2.4. Guía de Buenas Prácticas para el Desarrollo y Mantenimiento de Software				
Versión	1.1.1				
Fecha Entrega	30/06/2022			***Fecha Aprobación***	17/05/2023
Responsable	BSC (Organización)				
Nivel Diseminación	X	PU	Público		
		CO-IMP	Confidencial, sólo participantes de los pilares de IMPaCT, incluyendo la comisión de evaluación de IMPaCT.		
		CO-DATA	Confidencial, sólo participantes de IMPaCT-Data, incluyendo la comisión de evaluación de IMPaCT.		

Autores		
Organización	**Nombre**	**Rol**
BSC	Lidia López	Autora
BSC	Salvador Capella-Gutierrez	Autor
CNB	Laura del Cano	Revisora
UPV	Ignacio Blanquer	Revisor

Historial de versiones			
Nro.	**Fecha**	**Descripción**	**Autor**
v 0.0	12/04/2022	Documento creado	L.López (BSC-CNS)
v 0.1	04/05/2022	primera versión índice	L.López (BSC-CNS)
v 0.2	13/05/2022	Revisada v 0.1	L. del Caño (CNB-CSIC)
v 0.3	23/05/2022	Revisada v 0.2	I. Blanquer (UPV)
v 0.4	31/05/2022	Contenido sección 2 (recomendaciones)	L.López (BSC-CNS)
v 0.5	02/06/2022	Contenido sección 3 (implementación) y anexos	L.López (BSC-CNS)
v 0.6	03/06/2022	Conclusiones	L.López (BSC-CNS)
v 0.7	10/06/2022	Comentarios sobre EDAM	L.López (BSC-CNS)
v 0.8	20/06/2022	Revisados algunos números de la sección 4	L.López (BSC-CNS)
v 0.9	24/06/2022	Revisada	L. del Caño (CNB-CSIC)
v 0.10	29/06/2022	Revisada	I. Blanquer (UPV)
v 0.11	10/07/2022	Cambios relacionados con los comentarios de los revisores	L.López (BSC-CNS)
V 0.12	14/07/2022	Revisada	Laura del Caño (CNB-CSIC)
V 0.13	21/07/2022	Lista herramientas identificadas incluida como anexo	L. López (BSC-CNS)
v 1.0	29/07/2022	Revisión general	S. Capella-Gutierrez (BSC-CNS)
v 1.1	17/05/2023	Cambio visibilidad a público y aprobado	Comité Dirección
v 1.1.1	14/06/2023	Cambio de formato para publicar en la Web de IMPaCT	David Velasco (ISCIII)

Contenido

Contenido 4
Tablas 4
Figuras 5
Resumen Ejecutivo 6
Introducción 8
Audiencia 8
Ámbito 8
Relación con otros Entregables 8
Estructura Entregable 8
1 Motivación 9
2 Buenas prácticas para el desarrollo de software de investigación de código abierto (4OSS) 9
2.1 Recomendación: Código Fuente Abierto (OSS) 10
2.2 Recomendación: Software Localizable 11
2.3 Recomendación: Escoger Licencia Apropiada 14
2.4 Recomendación: Definir Comunicación, Gobernanza y Contribución 15
3 Calidad del Software 16
4 Implementación de 4OSS en IMPaCT-Data 17
5 Conclusiones 21
Referencias 22
Anexo A. Relación de herramientas seleccionadas para Infrastructura IMPaCT-Data 23
Anexo B. Seminario bio.tools and EDAM 26
Anexo C. Seminario de encapsulado de componentes software en contenedores 32
Anexo D. Seminario Workflows - Uso de gestores de workflows, registro y compartición 38

Tablas

Tabla 1. Combinación de licencias OSS....15
Tabla 2. Repositorios para el catálogo de software de IMPaCT-Data....19
Tabla 3. Comentarios sobre la ontología EDAM para anotación de herramientas software .20

Figuras

Figura 1. Registro bio.tools 12
Figura 2. Clasificación concepto Relation extraction en la ontología EDAM 13
Figura 3. Seminarios para aplicar 4OSS en IMPaCT-Data 18
Figura 4. Asistencia a seminarios/tutoriales 18
Figura 5. Uso de licencias OSS para las herramientas software 19
Figura 6. Uso de la ontología EDAM en El registro en bio.tools 20

Resumen Ejecutivo

El Software de investigación es un componente esencial en disciplinas científicas con un gran volumen de datos. En el campo de Ciencias de la Vida es difícil imaginar llevar a cabo la mayoría de las actividades científicas sin contar con el uso de software para el procesamiento, análisis, visualización e interpretación de grandes cantidades de datos. En este documento se presentan las cuatro recomendaciones incluidas en la guía de buenas prácticas para el desarrollo de software de código abierto producidas por ELIXIR. Estas recomendaciones, están basadas en los valores del código fuente abierto, y se han escrito para que el software desarrollado por investigadores sea más localizable (fácil de encontrar), reusable y transparente, en resumen, para que el software sea más fácil de mantener y extender en el tiempo. Estas cuatro recomendaciones son: (1) hacer público el código fuente en sistemas de control de versiones, p. ej. GitHub, GitLab, desde el primer día; (2) hacer el software localizable a través de su publicación en registros utilizados por la comunidad para tal fin; (3) escoger una licencia reconocida por la comunidad, con preferencia por licencias de software libre, teniendo en cuenta las licencias de los componentes del que depende el software en cuestión; y (4) definir los mecanismos de comunicación, gobernanza y colaboración más allá de los propios desarrolladores del software de investigación.

En este documento también se ha incluido las acciones que se han llevado a cabo en el marco de IMPaCT-Data para facilitar la adopción de las cuatro recomendaciones anteriores. Estas acciones se refieren a una serie de sesiones de trabajo y seminarios para facilitar acceso a las plataformas, buenas prácticas y guías a los miembros de IMPaCT-Data que les permitan implementar las recomendaciones aquí descritas. Los seminarios que se han impartido han incluido los siguientes temas: (1) registro de herramientas en el registro bio.tools y el uso de la ontología EDAM para describir el software de investigación; (2) guía de buenas prácticas para el desarrollo de software recogiendo ejemplos prácticos de cada una de las recomendaciones; (3) encapsulado de software de investigación de forma modular en contenedores; y (4) uso de gestores de flujos de trabajo, popularmente conocidos como workflows o pipelines; el registro de dichos workflows y su publicación para el potencial uso por parte de la comunidad científica. Estos seminarios han sido impartidos por ponentes nacionales e internacionales, incluyendo el Centro de Regulación Genómica y el Barcelona Supercomputing Center – Centro Nacional de Supercomputación asi como la Universidad de Bergen (Noruega), la Universidad Técnica de Dinamarca (Dinamarca), la Universidad Libre de Freiburg (Alemania), todos ellos miembros de la plataforma de herramientas de ELIXIR y otras comunidades relevantes como Galaxy y Bioconda. A estos seminarios han asistido representantes de 34 de las 47 instituciones miembros del proyecto.

Como resultado de las acciones realizadas para la implantación de las recomendaciones, al cierre de este documento, se han identificado 75 herramientas dentro del consorcio, de las cuales 57 se han clasificado como herramientas software, 15 como workflows y 3 como bases de datos. De las herramientas identificadas, 51 tienen licencia OSS y 58 han sido incluidas en el registro de ELIXIR bio.tools.

Introducción

Audiencia

Este documento está destinado a todos los participantes del proyecto IMPaCT-Data, para que puedan seguir las recomendaciones para el desarrollo y mantenimiento de software de investigación. El cumplimiento de las recomendaciones es opcional dado que existe software de investigación que existe con antelación a IMPaCT-Data el cual puede utilizarse en el contexto del proyecto y cuya autoría puede ser de miembros ajenos al consorcio. En el caso de software desarrollado desde cero en el contexto de este proyecto, se espera el cumplimiento de estas recomendaciones para contribuir a su uso, difusión y sostenibilidad.

Ámbito

Las recomendaciones que se describen en este documento se aplicaran a lo largo de todo el proyecto.

Relación con otros Entregables

Este entregable no tiene relación directa con ningún otro, aunque es esperable que estas recomendaciones se sigan para el software desarrollado en el contexto de IMPaCT-Data.

Estructura Entregable

En la Sección 1 se explica la motivación para el seguimiento de las recomendaciones que forman parte de esta guía de buenas prácticas. Las recomendaciones se explican en la Sección 2. La Sección 3 incluye material complementario referente a la mejora de la calidad del código fuente de los componentes software. En la Sección 4 se han incluido las acciones realizadas hasta el momento para fomentar la implementación de las recomendaciones. Finalmente, en la Sección 5 se incluyen las conclusiones.

En los anexos de este documento se incluye la lista de herramientas identificadas para el catálogo de IMPaCT-Data (Anexo A) y el material de soporte para implementar algunas de las recomendaciones. El Anexo B incluye detalles sobre el registro de herramientas software en el registro de ELIXIR bio.tools. El Anexo C incluye detalles sobre el encapsulado de software en contenedores para la instalación de herramientas software en distintos entornos computacionales. Finalmente, el Anexo D incluye detalles sobre la gestión de flujos de trabajo (*workflows* o *pipelines*), en concreto sobre distintos gestores de workflows, su registro y publicación para el potencial uso por parte de la comunidad, ya sea de IMPaCT-Data o ajena al proyecto.

1 Motivación

El Software es una parte importante en la investigación en el campo de ciencias de la vida. De hecho, ciertas investigaciones no se podrían realizar si no existieran las herramientas software que lo permitieran. Cuando el software se desarrolla como parte de las herramientas que dan soporte a la investigación, habitualmente no se considera como un resultado de la investigación, por lo que no se le aplican los mismos procesos para asegurar su calidad, referencia, reproducibilidad y reusabilidad.

La producción y uso de software de código abierto (OSS, siglas en inglés para Open Source Software) es uno de los mecanismos que permiten aplicar los principios de ciencia abierta a los componentes software. OSS es software para el cual el código fuente es públicamente accesible para cualquier persona que quiera inspeccionar, usar, modificar y mejorar dicho código. Tener el código de las herramientas software que se utilizan para la investigación contribuye al reconocimiento de los desarrolladores del mismo, a crear comunidad alrededor de dichos desarrollos, así como a aumentar la confianza sobre el mismo al permitir la posibilidad de inspeccionar dicho código. Es importante señalar que, dependiendo de las licencias de los componentes utilizados en el desarrollo de software, puede ser necesario utilizar el mismo tipo de licencias, p. ej. familia de licencias GPL.

En este entregable se exponen recomendaciones para el desarrollo de software de código abierto en el contexto de ciencia abierta.

2 Buenas prácticas para el desarrollo de software de investigación de código abierto (4OSS)

Estas recomendaciones están propuestas en la guía de buenas prácticas definida por el grupo de trabajo *Software development best practices for Life Sciences*[1] de la organización ELIXIR[2] [1]. El objetivo de este grupo de trabajo es mejorar la calidad y sostenibilidad del software desarrollado por investigadores en el dominio de ciencias de la vida.

Estas recomendaciones se resumen en:

- Hacer público el código fuente en sistemas de control de versiones, p. ej. GitHub, GitLab, desde el primer día.
- Hacer el software localizable a través de su inclusión en registros generales o específicos de la comunidad en cuestión.
- Escoger la licencia más adecuada.
- Definir Comunicación, Gobernanza y Contribución.

[1] https://elixir-europe.org/platforms/tools/software-best-practices
[2] https://elixir-europe.org/

Estas recomendaciones no pretenden substituir las guías de buenas prácticas existentes para el desarrollo de software, sino complementarlas. Estas recomendaciones están basadas en los valores del código fuente abierto, y se han escrito para mejorar el software desarrollado por investigadores permitiendo que sea más localizable (fácil de encontrar), re-usable y transparente.

De hecho, estas recomendaciones están alineadas con los principios FAIR (siglas en inglés para los términos **F**indable, **A**ccesible, **I**nteroperable y **R**e-usable) definidos en el contexto de gestión de datos en el ámbito científico, y en la actualidad trasladados a distintos tipos de objetos digitales, incluido el software de investigación.

Además de la definición y publicación de estas recomendaciones, el grupo de trabajo *Software development best practices for Life Science*, en colaboración con *the Carpentries*[3], ha producido el material de formación *4 Simple recommendations for Open Source Software (4OSS)*[4].

2.1 Recomendación: Código Fuente Abierto (OSS)

Para maximizar la reproducibilidad, reusabilidad y la colaboración se recomienda tener el código fuente en un repositorio abierto (de acceso público) con control de versiones, por ejemplo, GitHub o GitLab. Tener el código fuente disponible públicamente también ayuda a mejorar la visibilidad y la confianza en el componente software.

Los beneficios de esta recomendación son:

- Promueve la confianza del componente software en concreto y del proyecto en el que se está desarrollando en general.
- Facilita encontrar proyectos existentes en los que se está desarrollando software.
- El uso de un sistema de control de versiones proporciona acceso al historial de las contribuciones, lo cual ayuda a dar reconocimiento a los contribuidores de código.
- Incentiva la contribución de la comunidad.
- Incrementa las oportunidades de colaboración y re-uso.
- Expone el trabajo a la evaluación de la comunidad, posibilitando sugerencias y validaciones del código.
- Incrementa la transparencia para el escrutinio de la comunidad.
- Incentiva a los desarrolladores a pensar e implementar buenas prácticas al desarrollar código.
- Facilita la reproducción de los resultados científicos generados por las versiones anteriores del software.
- Incentiva a los desarrolladores a generar documentación, incluyendo un manual de usuario y comentarios en el código fuente.

[3] https://carpentries.org/
[4] https://softdev4research.github.io/4OSS-lesson/

- Garantiza el almacenamiento a largo plazo del código fuente.

Una buena práctica al crear un repositorio de código en un repositorio abierto es incluir el archivo *README.md,* en el directorio raíz, incluyendo una descripción del componente para que los usuarios o potenciales contribuidores puedan entender su propósito y/o funcionalidad, así como las políticas de contribución al código.

2.2 Recomendación: Software Localizable

Consiste en registrar el software en un registro que sea popular en el dominio, al registrarlo incluir los suficientes metadatos para maximizar las posibilidades que sea encontrado en las búsquedas.

Los beneficios de esta recomendación son:

- Incrementa la visibilidad del proyecto, del software, su uso, sus éxitos, referencias y quienes contribuyen.
- Incentiva a los desarrolladores de software a pensar que metadatos describen mejor el software y cómo exponerlo.
- Ayuda a exponer los metadatos en un formato entendible por las máquinas (p. ej. Buscadores) a través del registro.
- Incrementa las opciones de colaboración, re-uso y mejora.

Dependiendo del tipo de software, hay que buscar el registro adecuado para darle la máxima visibilidad en el dominio más adecuado. Por ejemplo, en el contexto de IMPaCT-Data, para las herramientas software hemos escogido el registro bio.tools y para los flujos de trabajo, el registro WorkflowHub. Estos dos recursos forman parte del ecosistema de ELIXIR, lo cual se alinea con las directivas fijadas por IMPaCT. Cada registro tiene sus normas para la anotación de sus entradas, por ejemplo en el registro bio.tools se utiliza la ontología EDAM[5] para la anotación de los recursos que contiene (ver Anexo B), que contiene más de 3.500 conceptos clasificados y relacionados entre ellos. De hecho, en EDAM los conceptos están clasificados como:

- Tema (traducción de *Topic*[6]): Una categoría que denota un dominio o campo de interés, de estudio, aplicación, trabajo, datos o tecnología. Los temas no tienen fronteras claramente definidas entre sí.
- Operación (*Operation*[7]): Una función que procesa un conjunto de entradas y da como resultado un conjunto de salidas, o asocia los argumentos (entradas) con los valores

[5] https://edamontology.org

[6] https://bioportal.bioontology.org/ontologies/EDAM?p=classes&conceptid=topic_0003

[7] https://bioportal.bioontology.org/ontologies/EDAM/?p=classes&conceptid=http%3A%2F%2Fedamontology.org%2Foperation_0004

(salidas).Datos (*Data*[8]): Información, representada en un artefacto de información (registro de datos) que es "comprensible" por herramientas computacionales dedicadas que pueden usar los datos como entrada o producirlos como salida.

- Formato (*Format*[9]): Una forma definida o diseño de representar y estructurar datos en un archivo de ordenador, blob, string, mensaje o en cualquier otro lugar.

Las anotaciones en el registro bio.tools utilizan los conceptos asociados a las categorías *Topic* y *Operation*. En la Figura 1 se muestran dos de las herramientas de IMPaCT-Data registradas en bio.tools, después de la descripción se ven los conceptos relacionados con el *Topic* en la primera línea y con el *Operation* en la segunda, las etiquetas de la tercera línea son otras clasificaciones fuera de EDAM (tipo herramienta, licencia, colección, etc.).

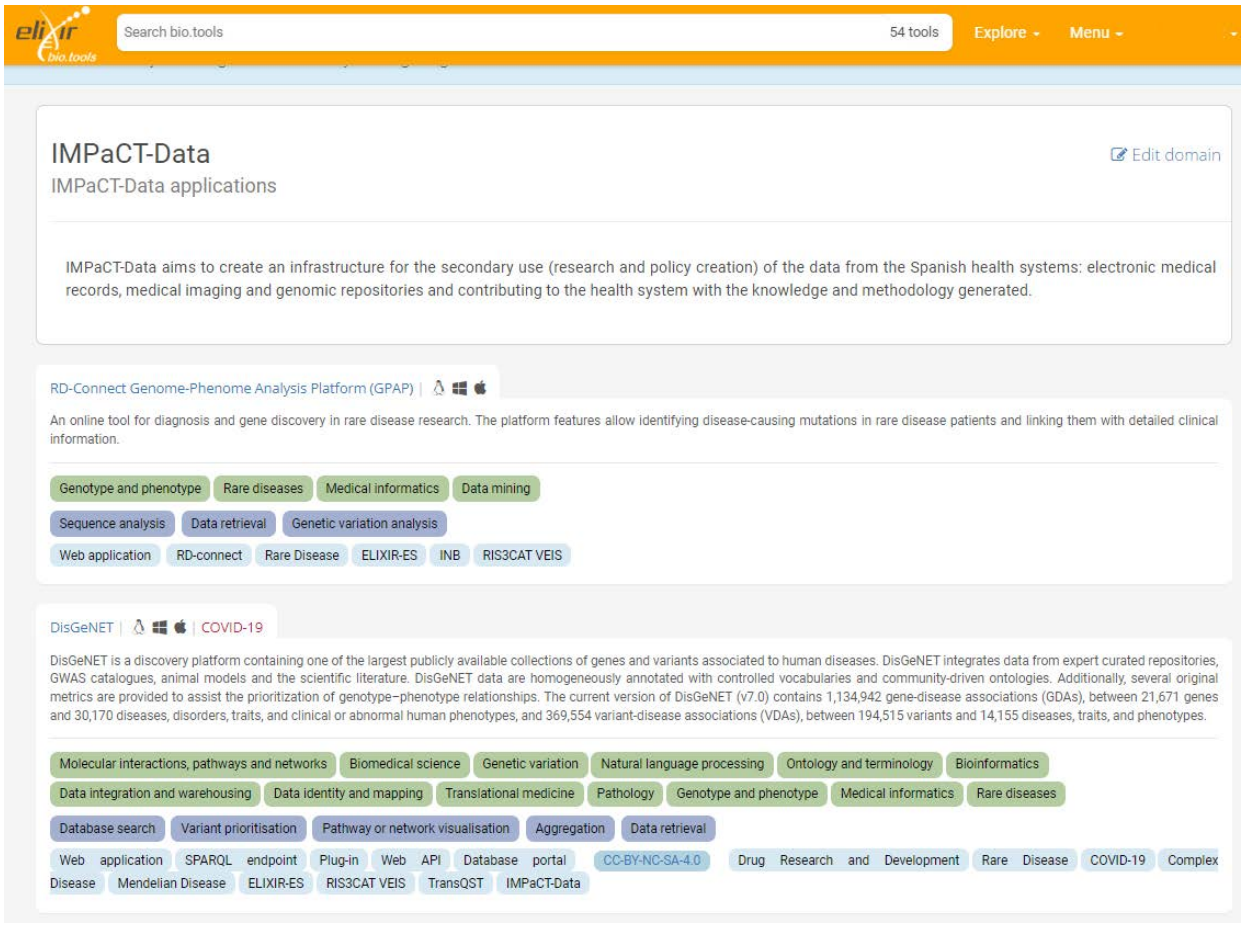


***Figura 1**. Registro bio.tools*

8 https://bioportal.bioontology.org/ontologies/EDAM/?p=classes&conceptid=http%3A%2F%2Fedamontology.org%2Fdata_0006

9 https://bioportal.bioontology.org/ontologies/EDAM/?p=classes&conceptid=http%3A%2F%2Fedamontology.org%2Fformat_1915

Para la herramienta *RD-Connect Genome-Phenome Analysis Platform (GPAP)* de la Figura 1, los conceptos de EDAM son:

- Topic: *Genotype and phenotype* (de la clasificación *Biology*/*Genetics*), Rare diseases (*Medicine*/*Pathology*), Medical Informatics (*Informatics* classification), y *Data Mining* (*Computer Science*)
- Operation: *Data Retrieval* (clasificaciones[10] *Analysis/Text mining/Information Retrieval* and *Data Handling/Query and Retreival*), Information Extraction (*Analysis/Text mining* y *Prediction and recognition/Text mining*), y Relation Extraction (*Analysis/Text mining* y *Prediction and recognition/Text mining*).

En la Figura 2 se puede ver la clasificación del concepto *Relation extraction* que pertenece a dos clasificaciones dentro de *Operation*.

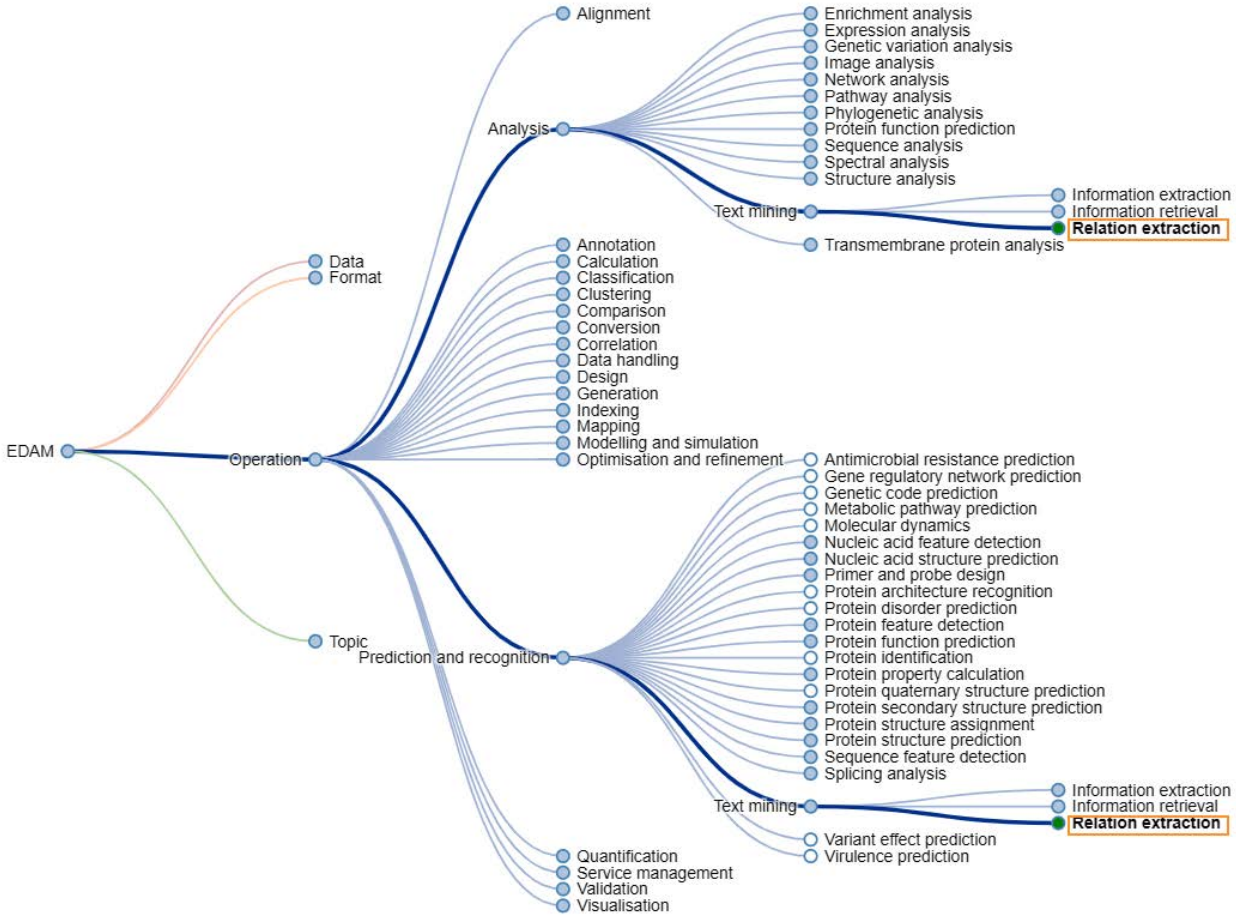


Figura 2. Clasificación concepto Relation extraction en la ontología EDAM

[10] Un concepto puede estar clasificado en varias clasificaciones

2.3 Recomendación: Escoger Licencia Apropiada

A la hora de desarrollar software de investigación, es importante asignar una licencia a dicho software. Siguiendo las directrices marcadas por IMPaCT, el software desarrollado debe estar disponible bajo licencias de código abierto (Open Source). Escoger la licencia Open Source adecuada permite clarificar los términos y condiciones de uso, modificación y redistribución del código fuente, así como fija responsabilidades de los autores del software. En el caso que el software utilice otros componentes disponibles bajo licencias de código abierto, se debe comprobar que se están cumpliendo los términos y condiciones de la licencia de cada componente y si la licencia es compatible con la que hemos escogido para el nuestro.

Los beneficios de esta recomendación son:

- Clarifica las responsabilidades y derecho de terceros que quieran utilizar, copiar, redistribuir, modificar o re-usar el código fuente.
- Permite utilizar el código fuente en jurisdicciones en las que "código sin licencia" significa que no se puede utilizar de ninguna manera.
- Protege la propiedad intelectual del software.
- Proporciona sostenibilidad a largo plazo permitiendo contribuciones y reutilización en el contexto de organizaciones legales con financiación.

Una buena práctica al crear un repositorio en un repositorio abierto es incluir el archivo *LICENSE.md,* en el directorio raíz, incluyendo el texto de la licencia escogida para el componente.

En cuanto al uso de licencias de código abierto, se recomienda escoger una licencia aprobada por la Open Source Initiative[11] (OSI), excepto que se necesiten condiciones especiales. En la web de OSI se pueden encontrar la lista de las licencias aprobadas y los criterios a cumplir por las licencias.

Las licencias OSS se distinguen entre *copyleft* y permisivas. Las licencias *copyleft* se caracterizan porque los trabajos derivados deben ser puestos a disposición de la comunidad obligatoriamente, lo que significa que el software se debe redistribuir utilizando la misma licencia o una más restrictiva. Las licencias permisivas no incluyen restricciones en los trabajos derivados, siempre que se de crédito a los autores originales, lo cual permite su potencial uso comercial. Hay un tercer grupo de licencias que está entre los dos tipos anteriores, el código en sí está disponible bajo una licencia *copyleft* pero los componentes (normalmente librerías) se pueden combinar con otro tipo de licencias sin la obligación de distribución bajo el mismo tipo de licencia (las llamaremos copyleft permisivas).

Si queremos distribuir un componente OSS, la licencia que se puede escoger para la distribución depende de dos factores: (1) la licencia original del componente y (2) el tipo de distribución que se quiere hacer. En la siguiente tabla se definen los tipos de licencia que se pueden aplicar (celdas) a la distribución de un software dependiendo de la licencia original

[11] https://opensource.org/licenses

(columnas) y de su tipo de distribución (filas). Se ha considerado la distribución como trabajo derivado (se incluyen modificaciones en el código fuente), si se distribuye combinado con otros componentes y ambos.

Tabla 1. Combinación de licencias OSS

	Copyleft	Copyleft permisiva	Permisiva
Derivado	Sólo si la licencia es copyleft	Dependiendo de cómo se enlace (linking) el componente dentro del proyecto	Cualquier tipo de licencia
Combinado	Solo si la licencia es copyleft	Cualquier tipo de licencia	Cualquier tipo de licencia
Derivado y combinado	Solo si la licencia es copyleft	Dependiendo de cómo se enlace (linking) el componente dentro del proyecto	Cualquier tipo de licencia

2.4 Recomendación: Definir Comunicación, Gobernanza y Contribución

Definir los procesos de comunicación, gobernanza y contribución no significa que el software deba ser desarrollado colaborativamente. Sin embargo, hay que definir de manera clara la estrategia de colaboración y contribución teniendo un modelo transparente de gobernanza y canales de comunicación.

Los beneficios de esta recomendación son:

- Incrementa la transparencia en cómo el proyecto y el desarrollo del software se gestiona.
- Ayuda a definir responsabilidades y procesos de toma de decisiones.
- Ayuda a la comunidad a colaborar, contribuir y comunicarse con el proyecto.

La comunicación se refiere a establecer los mecanismos necesarios para el trabajo colaborativo dentro del proyecto, incluyendo la puesta a punto de distintos canales de comunicación. De hecho, los canales de comunicación dependerán de diversos factores teniendo en la cantidad de peticiones de mejora esperadas y del tamaño del equipo de desarrollo. Estos mecanismos se deben implantar desde el inicio del proyecto. Por ejemplo, el uso de listas de distribución, plataformas de mensajería como Slack[12] o rocket chat[13] y los *issues* en los propios repositorios de código. Para comunicaciones importantes, como por ejemplo la publicación de una nueva versión, estas se pueden hacer a través de un blog asociado al proyecto u otro espacio web y a través de los canales de los contribuidores del código.

[12] https://slack.com/intl/es-es/
[13] https://www.rocket.chat/

La gobernanza se refiere a los procesos de toma de decisiones, idealmente se debe definir las diferentes responsabilidades (roles) y reconocimiento. Es importante no dejar responsabilidades sin definir dado que esto puede crear incertidumbre a medida que el proyecto software avanza y crece.

Para incentivar la colaboración de agentes externos, la inclusión de las guías de contribución es muy útil, por ejemplo, añadir el archivo *CONTRIBUTING.md* en el directorio raíiz repositorio. Esta guía debe incluir como mínimo:

- Reconocimiento/bienvenida a las personas que pretenden contribuir.
- Descripción de las diferentes maneras de contribución (secciones "*How-to*"). Por ejemplo, como reportar un error (*bug*), como contribuir con código, como sugerir mejoras. En el caso de permitir la contribución con código, habría que incluir un enlace a las convenciones de codificación y la guía de estilo.

- La manera de contactar.

En el curso producido por el grupo de trabajo de ELIXIR *Software development best practices for Life Sciences*[14] (sección 4) se incluye una lista de verificación que incluye aspectos que se deben tener en cuenta para hacer las contribuciones más fáciles, claras y transparentes.

También es importante definir un código de conducta, que se puede incluir en el repositorio raíz como un archivo separado (*CODE_OF_CONDUCT.md*) o en una sección del archivo *CONTRIBUTING.md*. Las comunidades Open Source no solo son espacios en los que se comparte tecnología, tienen un componente social de comunidad. El código de conducta define lo que se espera de los miembros de la comunidad en sus interacciones, lo que ayuda a mantener el bienestar de la comunidad. En el código de conducta se espera encontrar:

- Comportamiento esperado
- Comportamiento no aceptado
- Consecuencias en caso de comportamiento no aceptado
- Protocolo para reportar un comportamiento no aceptado por parte de alguno de los miembros

3 Calidad del Software

Las recomendaciones incluidas en la sección anterior no se refieren a la calidad del código fuente, son guías para mejorar la sostenibilidad del software desarrollado en entornos de investigación. Para facilitar el re-uso y el mantenimiento de estos componentes software, el código fuente debe alcanzar niveles de calidad aceptables.

Las actividades relacionadas con la gestión del proyecto también pueden influir en la calidad del software resultante. En el grupo de trabajo *Software development best practices for Life Sciences* en ELIXIR se está trabajando en un *Software Management Plan*[15] que puede ser

[14] https://softdev4research.github.io/4OSS-lesson/
[15] https://elixir-europe.org/sites/default/files/documents/software-management-plan.pdf

utilizado como una lista de comprobación (checklist) para comprobar si se están haciendo las actividades recomendadas para maximizar la calidad del software en desarrollo. Las actividades incluidas en este plan se refieren a documentación; pruebas (*testing*); interoperabilidad; comunidad, contribución y gobernanza, reproducibilidad y reconocimiento.

En el contexto del European Open Science Cloud[16] (EOSC), el proyecto EOSC-Synergy[17] tiene como objetivo el proporcionar servicios con calidad certificada. EOSC-Synergy ha definido un conjunto de criterios que se pueden utilizar como marco de referencia para maximizar la calidad de los desarrollos de software en los proyectos de investigación [2]. Estos criterios se refieren a la accesibilidad del código, selección de licencia, estilo del código, meta-datos del código, pruebas unitarias, herramientas para la automatización de las pruebas (*test harness*), desarrollo dirigido por pruebas (TDD por sus siglas en inglés *Test-Driven Development*), documentación, seguridad, modelos de ramas en el código (*code workflow/branching model*) de tal forma que correcciones y nuevas funcionalidades se desarrollan y prueban en ramas independientes, y cuando se ha alcanzado el nivel de calidad deseado se integran en la rama principal de desarrollo, versiones semánticas, gestión del código, revisión de código, entrega automática y despliegue automático.

4 Implementación de 4OSS en IMPaCT-Data

En el contexto del proyecto, se han organizado cuatro seminarios para facilitar la adopción de las recomendaciones descritas en este documento (ver Tabla 3):

- bio.tools and EDAM: How to publish information about your tools. Matúš Kalaš (Universitet of Bergen, Noruega) and Hans Ienasescu (Danmarks Tekniske Universitet, Dinamarca).
- Best practices for software development. Salvador Capella-Gutierrez (Barcelona Supercomputing Center - Centro Nacional de Supercomputación, España).
- Conteinarización componentes software. Björn Grüning (Bioconda/BioContainers community, Germany).
- Workflows: Uso de gestores de workflows, registro y compartición. José Espinosa-Carrasco, Center for Genomic Regulation (Centre for Genomic Regulation), Ignacio Eguinoa (Ghent University, ELIXIR-BE, Bélgica), Salvador Capella-Gutierrez, Barcelona Supercomputing Center – Centro Nacional de Supercomputación (BSC-CNS) .

[16] https://eosc-portal.eu/

[17] https://www.eosc-synergy.eu/

Figura 3. *Seminarios para aplicar 4OSS en IMPaCT-Data*

De las 47 instituciones que forman parte de IMPaCT-Data, han asistido representantes de 34 instituciones (72%) a al menos una de las sesiones. En la Figura 4 se indica el número de personas que asistieron a cada sesión (serie azul y con trama de rayas) y el número de instituciones identificadas (serie naranja con trama de cuadros). Aunque todas las sesiones han tenido un número elevado de asistentes, las que has suscitado más interés han sido la de buenas prácticas y encapsulado de software en contenedores para su instalación, ambas con una asistencia de 48 personas.

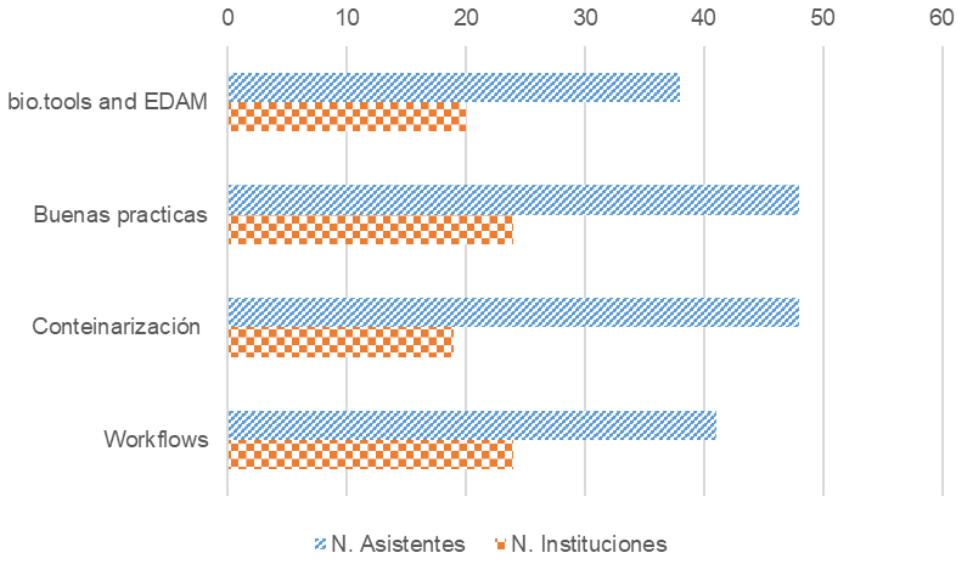


Figura 4. *Asistencia a seminarios/tutoriales*

Se ha pedido a los miembros del proyecto que nos indiquen las herramientas software/workflows que utilizan en el proyecto. Hasta el momento se han identificado 75 herramientas de 19 instituciones, 57 se han clasificado como herramientas software, 15 como workflows y 3 como bases de datos (listadas en el Anexo A).

Referente a la recomendación de tener el código fuente abierto (Sección 2.1), se han identificado las herramientas que se están desarrollando bajo una licencia OSS. De las 75 herramientas identificadas, se ha recogido esta información para 66 (9 herramientas no han proporcionado información sobre si la licencia es OSS o no), con el resultado de que la mayoría tienen licencia OSS (77%).

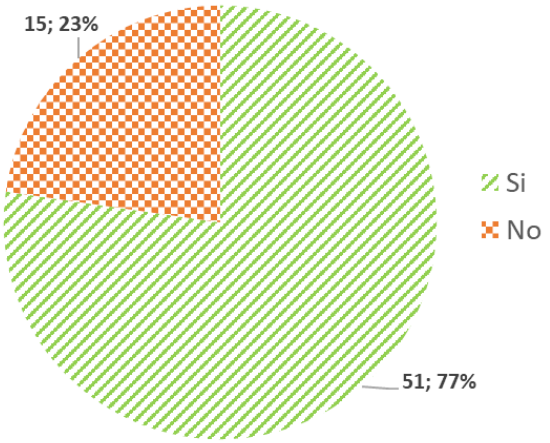


Figura 5. *Uso de licencias OSS para las herramientas software*

Para seguir la recomendación de hacer el software localizable (Sección 2.2, se han seleccionado diferentes repositorios que se utilizaran en el proyecto para mantener el catálogo de software. Las diferentes herramientas software pueden ser de diferente tipo y ser accesibles de diferente manera, la Tabla 2 incluye los detalles de los repositorios seleccionados.

Tabla 2. Repositorios para el catálogo de software de IMPaCT-Data

	Repositorio	Metadata	Detalles
Software	bio.tools[18]	EDAM (Anexo A)	IMPaCT-Data collection y domain
Containers	BIOCONDA[19]	--	--
Workflows	WorkflowHub[20]	EDAM (Topic, Operation)	--

De las 75 herramientas identificadas, a la finalización de este entregable, 58 se han registrado en el registro bio.tools. De las 58 registradas, 54 se han incluido en el dominio IMPaCT-Data[21], las 4 herramientas que no se han podido incluir en el dominio es debido a algún problema técnico que se está solucionando.

Se ha consultado a los miembros del proyecto por el uso que han hecho de la ontología EDAM para añadir los metadatos a sus herramientas al registrarlas, en concreto, se les ha preguntado si los términos definidos eran suficientes. De las 58 herramientas registradas, se

[18] https://bio.tools/
[19] https://bioconda.github.io/
[20] https://workflowhub.eu/
[21] https://bio.tools/t?domain=impact-data

han recibido 55 respuestas, en las que para la mayoría (73%) los términos definidos en la ontología han sido suficientes.

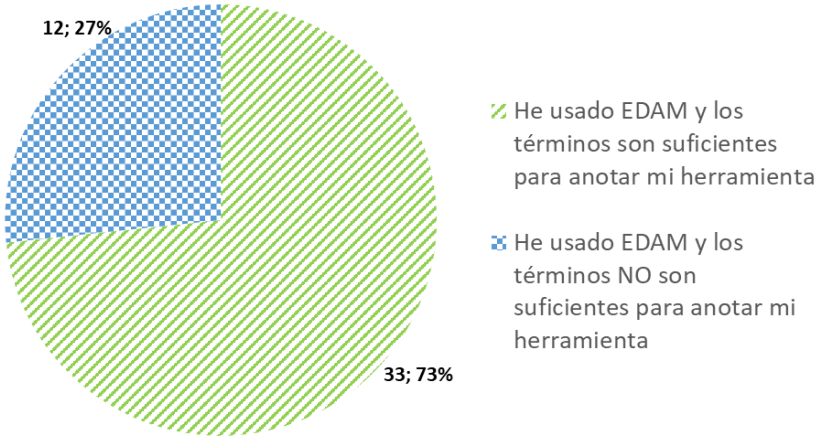


Figura 6. *Uso de la ontología EDAM en El registro en bio.tools*

Se han identificado a las personas de contacto de las herramientas que han reportado insuficiencias en la ontología EDAM, los comentarios recibidos se incluyen en la Tabla 3.

Tabla 3. Comentarios sobre la ontología EDAM para anotación de herramientas software

Herramienta	URL en bio.tools	Comentario sobre EDAM
mCSEA	https://bio.tools/mcsea	Han faltado términos sobre datos de metilación como input
ImaGEO	https://bio.tools/ImaGEO	Han faltado términos sobre meta-análisis
MetaGenyo	https://bio.tools/metagenyo	Han faltado términos sobre meta-análisis
DatAC	https://bio.tools/datac	Han faltado términos sobre epidemiología y factores ambientales
DExMA	https://bio.tools/dexma	Han faltado términos sobre meta-análisis
DomFun	https://bio.tools/domfun	Han faltado términos relacionados con transcriptómica (RNA-seq, miRNA-seq)
ExpHunterSuite	https://bio.tools/exphuntersuite	Han faltado términos relacionados con transcriptómica (RNA-seq, miRNA-seq)
iSkyLIMS	https://bio.tools/iskylims	Han faltado términos relacionados con la gestión de datos de laboratorio, o plataforma web de gestión de datos
Taranis	https://bio.tools/taranis	Han faltado términos relacionados con wg/cgMLST, MLST

5 Conclusiones

En este documento se presentan las cuatro recomendaciones incluidas en la guía de buenas prácticas para el desarrollo de software de código abierto producidas por ELIXIR. Estas recomendaciones están basadas en los valores del código fuente abierto, se han escrito para que el software desarrollado por investigadores sea más localizable (fácil de encontrar), reusable y transparente. Estas cuatro recomendaciones son: (1) Tener el código fuente abierto desde el primer día, (2) Hacer el software localizable; (3) Escoger la licencia más adecuada y (4) definir comunicación, gobernanza y colaboración.

Estas recomendaciones se han complementado con herramientas que se pueden utilizar para mejorar la calidad el código fuente.

En este documento también se ha incluido las acciones que se han llevado a cabo en el proyecto para facilitar la adopción de las cuatro recomendaciones. Estas acciones se refieren a una serie de sesiones de trabajo y seminarios para dar las herramientas a los miembros del proyecto para que las puedan implementar.

Como resultado de estas sesiones, al cierre de este documento, se han identificado 75 herramientas, de las cuales 57 se han clasificado como herramientas software, 15 como workflows y 3 como bases de datos. De las herramientas identificadas, 51 tienen licencia OSS y 58 han sido registradas en el portal bio.tools.

Referencias

[1] Jiménez, R. C., Kuzak, M., Alhamdoosh, M., Barker, M., Batut, B., Borg, M., ... & Crouch, S. (2017). Four simple recommendations to encourage best practices in research software. *F1000Research*, *6*.

[2] Orviz, P., Lopez, A., Duma, D. C., David, M., Gomes, J., & Donvito, G. (2021). *A set of Common Software Quality Assurance Baseline Criteria for Research Projects*. Manubot.

Anexo A. Relación de herramientas seleccionadas para Infrastructura IMPaCT-Data

En este anexo se incluye la lista de 75 herramientas identificada en el contexto de IMPaCT-Data, de esta lista 58 están registradas en el registro bio.tools siguiendo las recomendaciones incluidas en este documento (ver sección 2.2

Nombre	Tipo	OSS	URL bio.tools	Institución
ADEx	Software	Si	https://bio.tools/ADEx	FPS
APID Interactomes	Software	Si	https://bio.tools/apid	CSIC
APPRIS	Software		https://bio.tools/appris	CNIO
Automatic segmentation tool	Software	Si	https://bio.tools/automatic_segmentation_tool	INIBICA
Beyondcell	Software		https://bio.tools/beyondcell	CNIO
CohortAnalyzer	Workflow	Si		UMA
CoV-hipathia	Software	Si	https://bio.tools/cov-hipathia	FPS
CSVS	Software	Si	https://bio.tools/csvs	FPS
cypathia	Software	Si	https://bio.tools/cypathia	FPS
DatAC	Software	Si	https://bio.tools/datac	FPS
DExMA	Software	Si	https://bio.tools/dexma	FPS
DisGeNET	Software	Si	https://bio.tools/disgenet	IMIM
DiSMed	Software	Si	https://bio.tools/dismed	FISABIO
DomFun	Software	Si	https://bio.tools/domfun	UMA
DREIMT	Software		https://bio.tools/dreimt	CNIO
EvolClust	Database	No	https://bio.tools/EvolClust	BSC
ExpHunterSuite	Software	Si	https://bio.tools/exphuntersuite	UMA
FAIR4Health Data Curation Tool	Software	Si	https://bio.tools/fair4health_data_curation_tool	SAS-HUVR
FAIR4Health Data Privacy Tool	Software	Si	https://bio.tools/fair4health_data_privacy_tool	SAS-HUVR
FireDB	Software		https://bio.tools/firedb	CNIO
FJD-pipeline	Software	Si	https://bio.tools/fjd-pipeline	IIS-FJD
GeneCodis	Software	No	https://bio.tools/genecodis	FPS
GLOWgenes	Software	Si	https://bio.tools/glowgenes	IIS-FJD
GPAP	Software	No	https://bio.tools/rd-connect_platform	CRG
hipathia	Software	Si	https://bio.tools/hipathia-gemomics	FPS
Hipathia	Software	Si	https://bio.tools/hipathia	FPS
ImaGEO	Software	Si	https://bio.tools/ImaGEO	FPS
ImpuSARS	Software	Si	https://bio.tools/impusars	FPS
IonGAP	Software	Si	https://bio.tools/iongap	FIISC

iSkyLIMS	Software	Si	https://bio.tools/iskylims	ISCIII
Jupyter Hub	Software			FPS

Relación de herramientas seleccionadas para Infrastructura IMPaCT-Data (continuación)

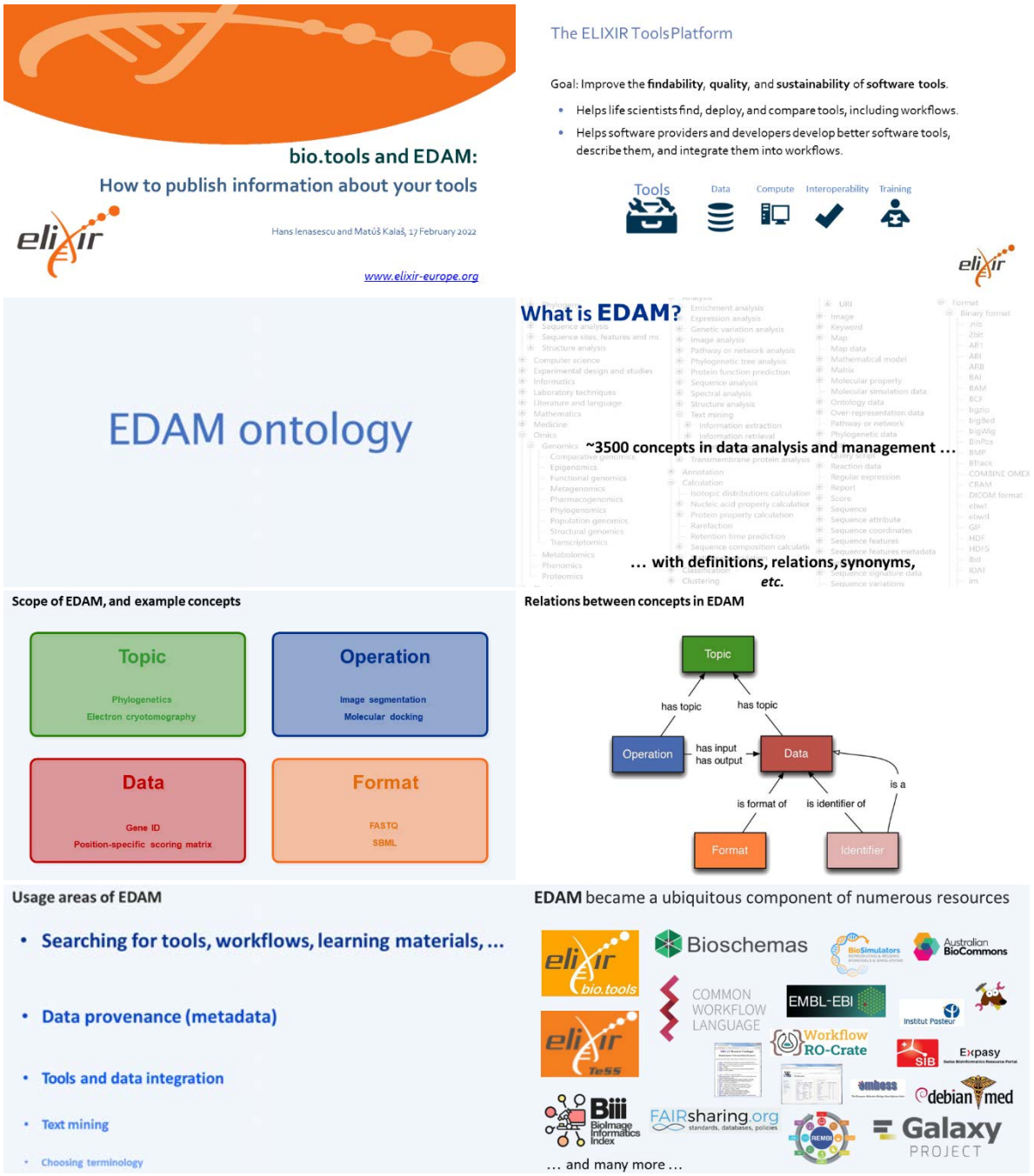
Nombre	Tipo	OSS	URL bio.tools	Institución
Liferay	Software	Si	https://bio.tools/liferay	HCB
LinkEHR	Software	Si	https://bio.tools/linkehr	HCB
mCSEA	Software	Si	https://bio.tools/mcsea	FPS
Metabolizer	Software	Si	https://bio.tools/metabolizer	FPS
MetaFun	Software	Si	https://bio.tools/metafun	CIPF
MetaGenyo	Software	No	https://bio.tools/metagenyo	FPS
meTAline	Workflow	Si		BSC
MetaPhors	Database	No	https://bio.tools/metaphors	BSC
MIDS	Software	Si		FISABIO
MIGNON	Software	Si	https://bio.tools/mignon	FPS
Mini-IsoQLR	Software	Si	https://bio.tools/mini-isoqlr	IIS-FJD
MyPROSLE	Software	Si	https://bio.tools/myprosle	FPS
NanoCLUST	Software	Si	https://bio.tools/nanoclust	FIISC
NanoDJ	Software	Si	https://bio.tools/NanoDJ	FIISC
NanoRtax	Software	Si	https://bio.tools/nanortax	FIISC
NetAnalyzer	Software	Si		UMA
nf-core-viralrecon	Workflow	Si	https://bio.tools/nf-core-viralrecon	ISCIII
ngsCAT	Software	Si	https://bio.tools/ngscat	FPS
OpenEBench	Software	Si	https://bio.tools/openebench	BSC
PanDrugs	Software		https://bio.tools/pandrugs	CNIO
PhenCo	Workflow	Si		UMA
PhenFun	Workflow	Si		UMA
PhylomeDB	Database	No	https://bio.tools/PhylomeDB	BSC
Pipeline CNVs germinal	Workflow	No		IdiPaz
Pipeline Germinal v1	Workflow	No		IdiPaz
Pipeline Germinal v2	Workflow	No		IdiPaz
Pipeline identificación de fusiones génicas en RNASeq	Workflow	No		INCLIVA
Pipeline Mosaico	Workflow	No		IdiPaz

Pipeline paneles/exomas en muestras tumorales	Workflow	No		INCLIVA
Pipeline RNASeq	Workflow	No		INCLIVA
Pipeline RNA-Seq	Workflow	No		IdiPaz
Pipeline Somáticas v1	Workflow	No		IdiPaz
PlasmidID	Software	Si	https://bio.tools/plasmidid	ISCIII

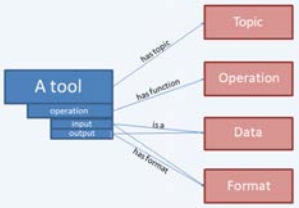
Relación de herramientas seleccionadas para Infrastructura IMPaCT-Data (continuación)

Nombre	Tipo	OSS	URL bio.tools	Institución
PriorR	Software	Si	https://bio.tools/priorr	IIS-FJD
PTMCode	Software	Si	https://bio.tools/ptmcode	IIS-FJD
RStudio Workbench	Software			FPS
Slides Viewer	Software	Si	https://bio.tools/slides_viewer	INIBICA
SMAca	Software	Si	https://bio.tools/smaca	FPS
SPACNACS	Software	Si	https://bio.tools/spacnacs	FPS
Taranis	Software	Si	https://bio.tools/taranis	ISCIII
TFTEA	Software	Si	https://bio.tools/tfta	FPS
TRIFID	Software		https://bio.tools/trifid	CNIO
vulcanSpot	Software		https://bio.tools/vulcanSpot	CNIO
XICRA	Workflow	Si	https://bio.tools/xicra	IGTP

Anexo B. Seminario bio.tools and EDAM



Annotation of computational tools with EDAM



bio.tools

The registry's motivation

- Large number of tools and databases created to support (life science) research
- Fragmented documentation and access of resources:
  - Hard to find, understand, compare and use
  - Lack formalized descriptions of scientific and technical functions
  - Don't ensure persistence of identifiers
  - Make reproducibility hard

- **Need for a registry / portal / database of life science software tools**
- Examples of tools registries / portals
  - EMBOSS
  - EMBRACE
  - BioCatalogue
  - SEQanswers
  - Omics Tools
  - BioMedBridges
  - Debian Med
  - Bioconductor

What is bio.tools?

- **bio.tools** strives to provide a comprehensive registry of software and data services facilitating researchers from across the spectrum of biological and biomedical science to find, understand, utilise and cite the resources they need in their day-to-day work

Created in the context of the ELIXIR Europe life science infrastructure project

From simple command-line tools and online services, through to databases and complex, multi-functional analysis workflows

Contains tool descriptions (information, annotations about tools), not the actual tools themselves

https://bio.tools

bio.tools principles

- Open data
  - Content is freely available to all under CC BY 4.0 license
- Open source
  - Source code is freely available to all under GPL-3.0
- Built by the community
  - 4800+ (and growing!) contributors
- Persistent IDs
  - Unique, persistent, human-readable resourceidentifiers
- Standard semantics
  - Scientific function of bio.tools resources can be precisely annotated in defined terms from the EDAM ontology, including common topics, operations, types of data and data formats
- Standard syntax
  - Resources adhere to a rigorous syntax
  - ~50 key scientific, technical and administrative attributes (4 required)
- Community-driven
- Backed by ELIXIR
  - bio.tools will remain free, open and maintained in the long-term

Data model behind: biotoolsSchema

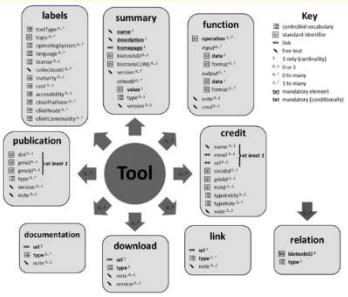
A simplified model which defined the attributes, information and scope provided to best describe software tools. A formal machine readable (and human understandable) schema to allow for data interoperability

- XML (JSON coming soon) schema
  - ~50 key scientific, technical and administrative attributes, uniform & rigorous syntax and semantics
- controlled vocabularies (18 in total)
  - e.g. tool type, software license, software maturity
- community-defined standard (v2.0, mature)
  - from multiple workshops /iterations
- compatible with related initiatives
  - shema.org / Bioschemas
- https://doi.org/10.1093/gigascience/giaa157

Data model: examples

- Required *
  - Name
  - bioToolsID
  - Description
  - Homepage
- Labels:
  - Topics
  - Tool Type
  - OS
  - License
  - Language
  - Cost
- Publication
- Download links
- Documentation links
- Other links
- Credits
- Function

bio.tools provides curation guidelines for each attribute to help curators and regular users describe tools in a standard manner

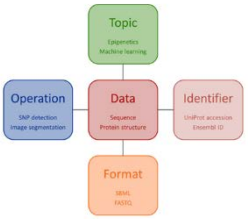


Scientific tool descriptions: EDAM ontology

Scientific function of bio.tools resources can be precisely annotated by concepts from the EDAM ontology

EDAM has 4 sections:

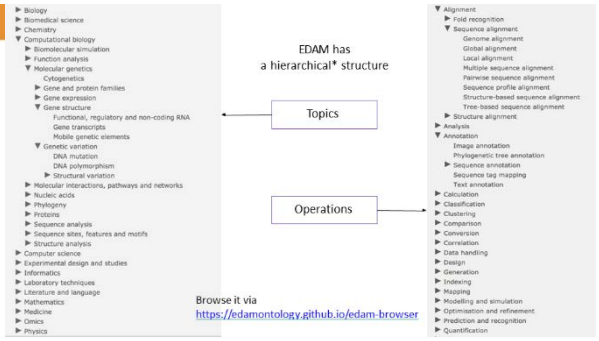
- Topic
- Operation
- Data (incl. Identifier)
- Format



EDAM ontology in bio.tools

- Topics in bio.tools define the scientific or technical domains the tool is developed for

  e.g. Transcription factors and regulatory sites , Genomics , Gene regulation
- Operations, Data and Formats are used in the context of functions
- Functions in bio.tools are represented as Input / Operation / Output triplets
  - Input (EDAM Data + EDAM Format)
  - Operation (EDAM Operation)
  - Output (EDAM Data + EDAM Format)



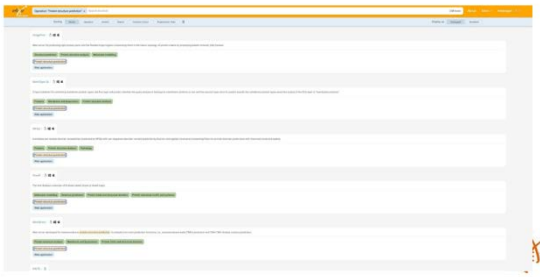
Collaboration with otherprojects

- EDAM Ontology: http://edamontology.org
- DebianMed: https://www.debian.org/devel/debian-med
- Galaxy: https://usegalaxy.eu
- BioContainers: http://biocontainers.pro
- Bioconda: https://bioconda.github.io
- SciCrunch: https://scicrunch.org
- ELIXIR TeSS: https://tess.elixir-europe.org
- FAIRSharing: https://fairsharing.org
- EuropePMC: https://europepmc.org
- ELIXIR Scientific Communities: https://elixir-europe.org/communities
- Australian BioCommons: https://www.biocommons.org.au
- MathWorks: https://www.mathworks.com
- OpenEBench: https://openebench.bsc.es
- Others

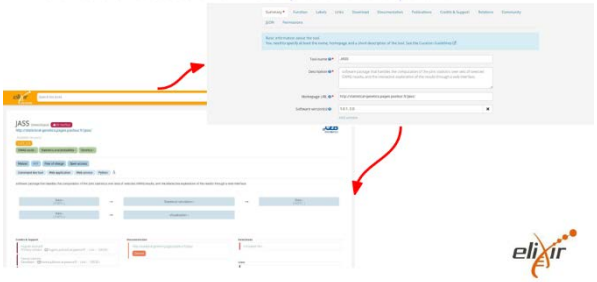
bio.tools usefullinks

- **APIs**
  - https://biotools.readthedocs.io/en/latest/api_reference.html
  - https://biotools.readthedocs.io/en/latest/api_usage_guide.html
- **Documentation**
  - https://biotools.readthedocs.io
  - https://biotoolsschema.readthedocs.io
- **Curation guidelines**
  - https://biotools.readthedocs.io/en/latest/curators_guide.html
- **Contact:**
  - registry-support@elixirmail.cbs.dtu.dk
  - https://github.com/bio-tools/biotoolsRegistry/issues
- **EDAM:**
  - https://edamontology.org
  - https://github.com/edamontology/edamontology

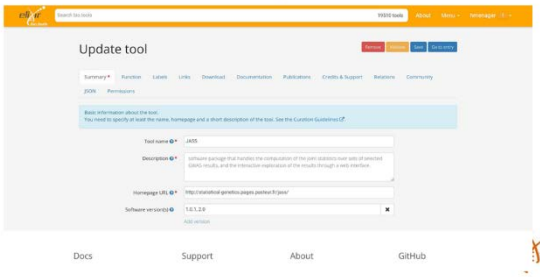
bio.tools: find tools



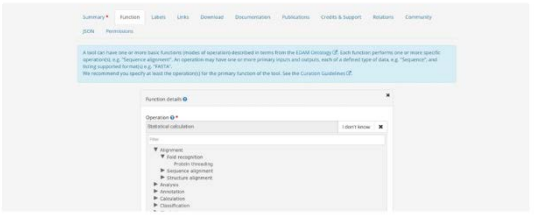
bio.tools: access and modify tool descriptions



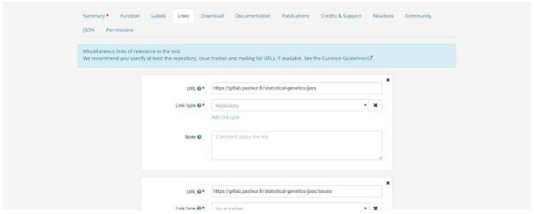
bio.tools curation: Summary



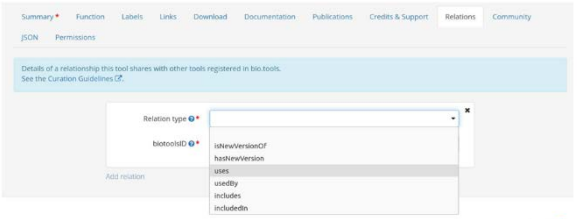
bio.tools curation: Function



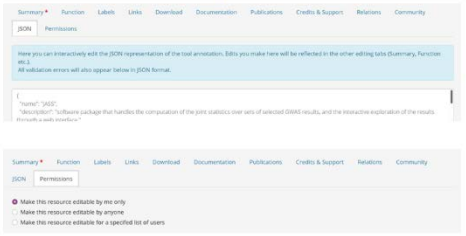
bio.tools curation: Labels, Links, etc.



bio.tools curation: Relations

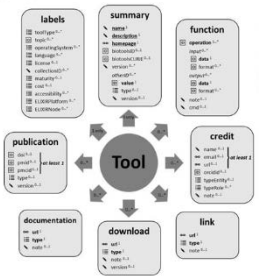


bio.tools curation: JSON, Permissions



bio.tools descriptions

biotoolsSchema



Add a new tool: required fields

Basic information about the tool. You need to specify at least the name, homepage and a short description of the tool. See the Curation Guidelines.

- Tool name *:
- Tool description *:
- Tool homepage URL *:
- Tool identifier (biotoolsID) *:
  - Suggested from name (URL safe)
  - Editable if user wants
- Tool version (here or somewhere else)

Tool properties links todocs

- Accessibility
- Collection
- Cost
- Credit & Support
- Documentation
- Download
- ELIXIR community
- ELIXIR node
- ELIXIR platform
- Input data type
- Input file format
- License
- Links
- Maturity
- Operating system
- Operation
- Other ID
- Output data type
- Output file format
- Programming language
- Publication
- Tag as COVID-19
- Tool type
- Topic

Labels

Miscellaneous scientific, technical and administrative details of the tool, expressed in terms from controlled vocabularies. We recommend you specify at least the tool type, license, and one or more topics. See the Curation Guidelines .

- Tag as COVID-19
- Tool type (Command-line, Web application, Desktop Application, Library)
- Operating system (Linux, Mac, Windows)
- Programming language (C++, Java, Python, R ...)
- Maturity (Emerging, Mature, Legacy)
- License (MIT, GPL-3.0, ... , Proprietary, Not licensed)
- Cost (Free, Free with restrictions, Commercial)
- Collection
- Accessibility (Open acces, Open access (with restrictions), Restricted access
- ELIXIR platform (Tools, Data, Interoperability, Compute, Training)
- ELIXIR node (Denmark, France, Germany, ...)
- ELIXIR community (Proteomics, Metabolomics, Rare diseases, ...)
- Tool confidence score (Tool, High, Medium, Low, Very Low)
- Other ID

Links

Links

- URL *
- Link type
- Note

Link types

- Discussion forum
- Galaxy service
- Helpdesk
- Issue tracker
- Mailing list
- Mirror
- Repository
- Service
- Social media
- Software catalogue
- Technical monitoring
- Other

Downloads

Downloads

- URL *
- Download type
- Version
- Note

Download types

- API specification
- Binaries
- Biological data
- Command-line specification
- Container file
- Downloads page
- Icon
- Screenshot
- Software package
- Source code
- Test data
- Test script
- Tool wrapper (CWL)
- Tool wrapper (Galaxy)
- Tool wrapper (Other)
- Tool wrapper (Taverna)
- VM image
- Other

Documentation

Documentation

- URL *
- Documentation type
- Note

Documentation types

- API documentation
- Citation instructions
- Code of conduct
- Command-line options
- Contributions policy
- FAQ
- General
- Governance
- Installation instructions
- Quick start guide
- Release notes
- Terms of use
- Training material
- User manual
- Other

Publication

- Digital Object ID (DOI)
- PubMed ID
- PubMed Central ID
- Publication type
  - Primary
  - Method
  - Usage
  - Benchmarking study
  - Review
  - Other
- Version
- Note

Credits & Support

- ORCID ID
- Name
- Email
- URL
- gridid
- rorid
- fundrefid
- Entity type
- Entity role
- Note

Entity types

- Person
- Project
- Division
- Institute
- Consortium
- Funding agency

Entity roles

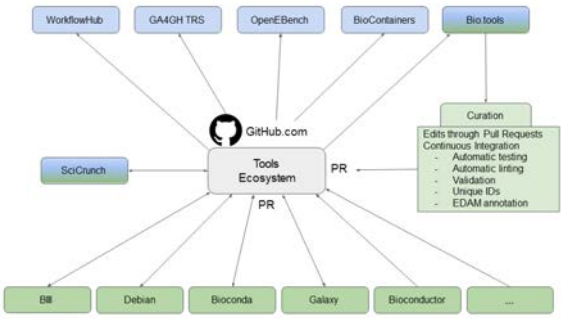
- Developer
- Maintainer
- Provider
- Documentor
- Contributor
- Support
- Primary contact

Tool properties links todocs

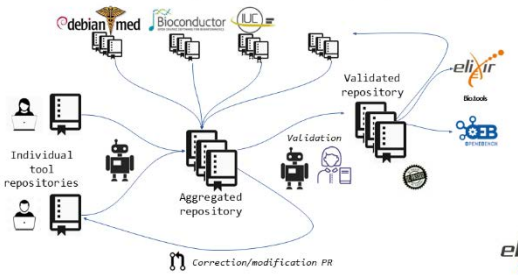
- Accessibility
- Collection
- Cost
- Credit & Support
- Documentation
- Download
- ELIXIR community
- ELIXIR node
- ELIXIR platform
- Input data type
- Input file format
- License
- Links
- Maturity
- Operating system
- Operation
- Other ID
- Output data type
- Output file format
- Programming language
- Publication
- Tag as COVID-19
- Tool type
- Topic

"Tools Ecosystem"

(a work in progress)



The **new** Tools Ecosystem architecture (WIP)



EDAM tools

EDAM Tool Annotator and ontology browsers for EDAM

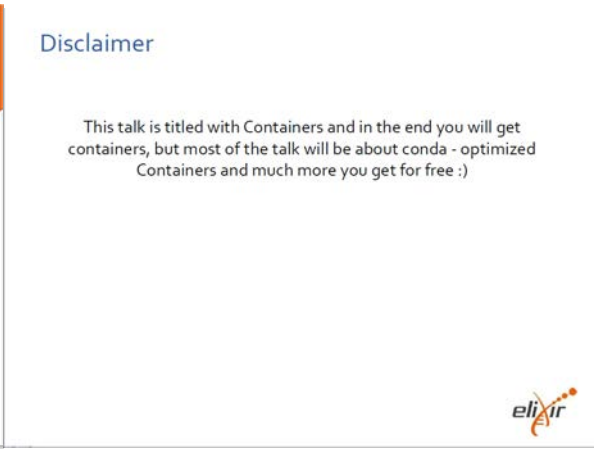
- https://bio.tools/static/eta (EDAM Tool Annotator)
- 3rd-party tooling to allow easier annotation of *bio.tools* entries with EDAM
- Provides an improved EDAM concept searching, and browsing over the *bio.tools*
- Steps:
  - Import tool metadata from *bio.tools* directly (via ID) or other sources
  - Annotate / edit EDAM-related tool metadata (topics, functions, ...)
  - Result is a *Bio.tools*-compatible JSON
  - Request EDAM concept/term if none match your preferences
- Ontology browsers for EDAM:
  - https://bioportal.bioontology.org/ontologies/EDAM (NCBO BioPortal)
  - https://edamotology.github.io/edam-browser (EDAM Browser)
  - https://www.ebi.ac.uk/ols/ontologies/edam (OLS ontology browser)

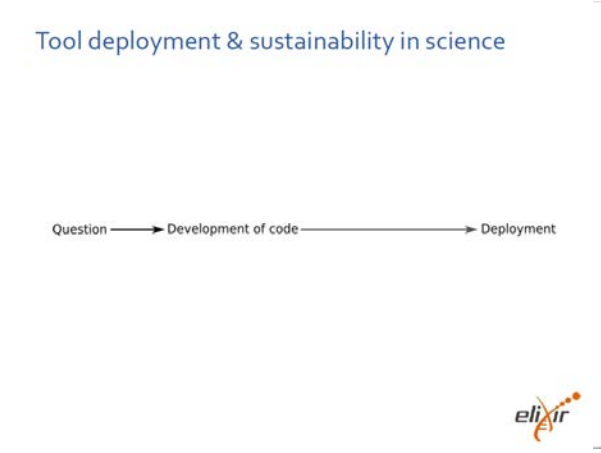
EDAM concepts from text mining

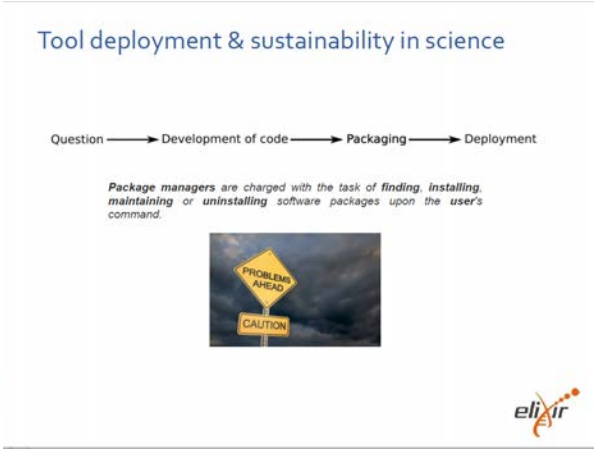
- EDAMmap
  - Obtain EDAM concepts from text mining (free text, publications, *etc.*)
  - https://biit.cs.ut.ee/edammap/
  - https://github.com/edamontology/edammap
- Pub2tools
  - Generate *bio.tools*-compatible JSON data (including EDAM) from text mining publications
  - https://github.com/bio-tools/pub2tools

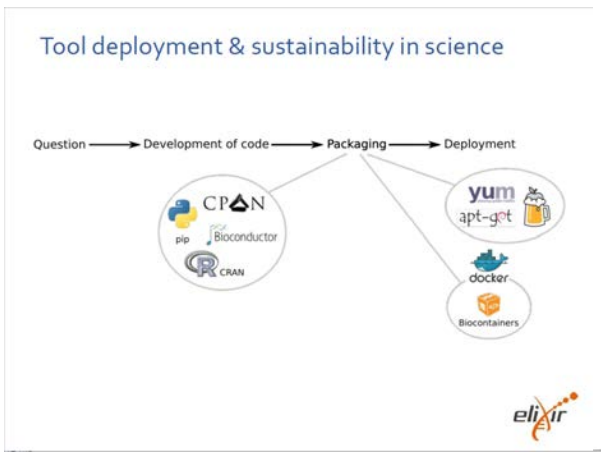
Demo 👩‍💻

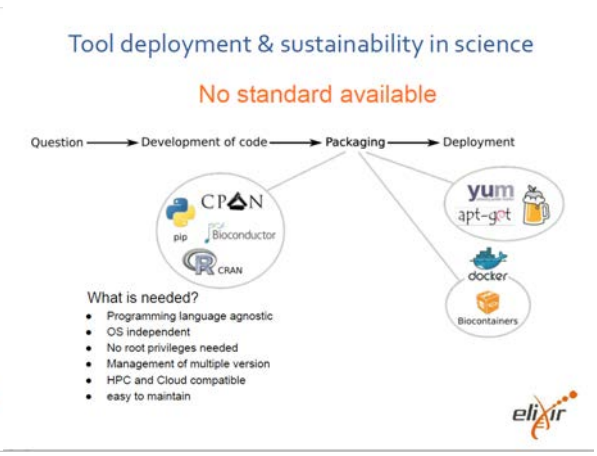
Anexo C. Seminario de encapsulado de componentes software en contenedores











CONDA

- Open Source package manager
- Independent of any programming language and OS
- Fast, robust and easy package installation
    - > conda install deeptools
- Installation and management of multiple versions

elixir

CONDA

elixir

CONDA

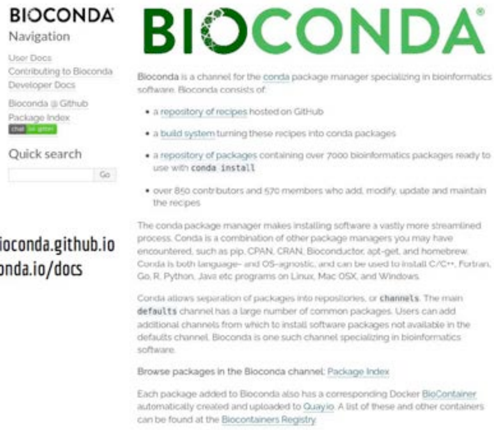
- Build packages
    - > conda build packages/deeptools
- building a scientific community
  BIOCONDA
- using a unified build environment
    - > circleci build
- joining other Conda communities CONDA-FORGE

https://bioconda.github.io | https://conda.io/docs

elixir

BIOCONDA

https://bioconda.github.io
https://conda.io/docs



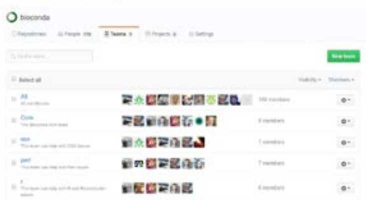
BIOCONDA

https://bioconda.github.io
https://conda.io/docs

elixir

BIOCONDA

- 9218 package (2022.04)
- contribute at *https://github.com/bioconda/bioconda-recipes*
- community driven

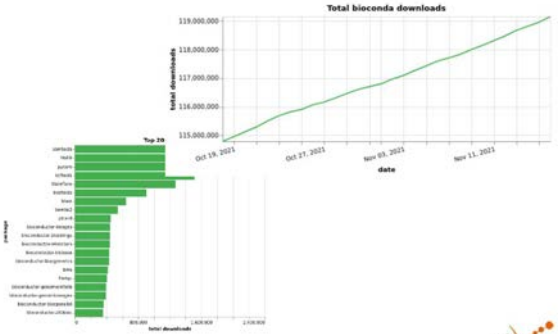


elixir

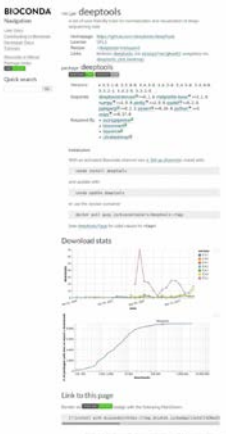
Community

1.419 Contributors ***~30.000 merged PR***

A few stats ...



Bioconda website:



But the new cool kid is called **Containers**

docker

rkt

But the new cool kid is called **Containers**

Question → Development of code → Packaging → Deployment

CONDA BIOCONDA

docker

rkt

But the new cool kid is called **Containers**

Question → Development of code → Packaging → Deployment

CONDA BIOCONDA

docker

Mulled

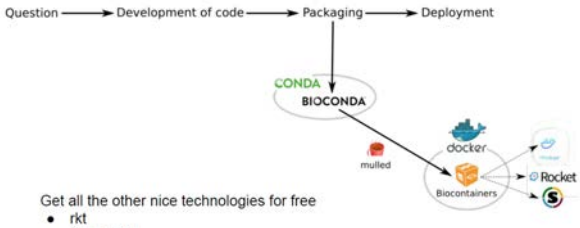
Biocontainers

Mulled

- A layer donning approach to build containers
- without Dockerfile
- completely automatised
- powered by CI

Sharing the same artefact automatically, across technologies.

Container build without Dockerfile



Get all the other nice technologies for free

- rkt
- Singularity
- Docker
- ~40.000 containers for ~8.000 tools

Automated Container builds

```
package:
    samtools: 1.3.1
```

vs.

```
process amberleap {
    container='quay.io/biocontainers/ambertools:20.4'
    input:
```

> mulled-build build-and-test 'samtools=1.3.1' --test 'samtools --help'

docker pull quay.io/biocontainers/**samtools:1.3.1**

Automated Container builds

```
package:
    samtools: 1.3.1
    bedtools: 2.26
```

> mulled-build build 'samtools=1.3.1,bedtools=2.26'

Automated Container builds

```
package:
    samtools: 1.3.1
    bedtools: 2.26
```

> mulled-build build 'samtools=1.3.1,bedtools=2.26'

Are multi-tool containers really needed? Decompose your workflows!

How do you name them?

No pets anymore, just cattles.

Findability!

Findability

- **no search** needed, just **retrieve or fail**
- build containers on the fly from packages
- build containers in-advance by monitoring GitHub repos

Automated Container builds

```
package:
  samtools: 1.3.1
  bedtools: 2.26
```

- predictable namespace for <samtools + bedtools> ?
- normalize package names, hash them
- normalize versions, hash them

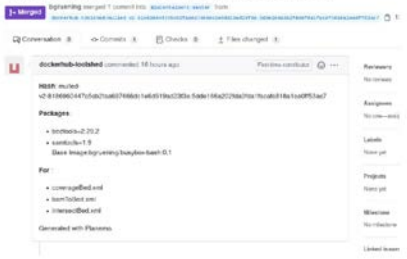
`mulled-v2-619c3451acc46ef686f36023:75e74fb16bc237935232e184b5187`

- https://github.com/BioContainers/multi-package-containers

Create containers via single line PRs

Let a bot do it for you!

Add container mulled-v2-8186960447c5cb2faa697666dc1e6d919ad23f3e:5dde166a202fda0fda1facafc818a1ea0ff53ac7. #1286



Bots

- new software is released
- **bot → conda**
- community review
- **bot → container**
- **(bot → missing container)**
- **(bot → missing singularity container)**
- **bot → Galaxy tool**
- community review
- **(bot → multi-package container)**
- **bot → workflow testing**

Who is using it?

Galaxy PROJECT
CYVERSE
Open Science Grid
COMMON WORKFLOW LANGUAGE
nextflow
elixir

Acknowledgements

Galaxy · Galaxy EUROPE · Galaxy AUSTRALIA · BIOCONDA · XSEDE · TACC · de.NBI · elixir · Pittsburgh Supercomputing Center · Indiana University · Galaxy Network · BioPlatforms Australia · ARDC · VIB · elixir Belgium · Vlaanderen · EOSC-Life · Datamonkey · ifb · Datamonkey · Bundesministerium für Bildung und Forschung · NSF · NIH National Institutes of Health

links

- cheat-sheet: https://raw.githubusercontent.com/bioconda/bioconda-outreach/master/ISMB2019/CheatSheet.pdf
- binder online bash: https://mybinder.org/v2/gh/gjbex/BinderBash/master
- https://drive.google.com/file/d/1R256MvMY_j_OGKzN7e8daVMPUbUEsiX6/view?usp=sharing

Anexo D. Seminario Workflows - Uso de gestores de workflows, registro y compartición

Nextflow and nf-core
IMPaCT-Data workshop
30th May 2022

Cedric Notredame's lab

Jose Espinosa-Carrasco

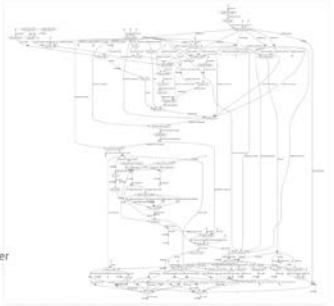
CRG Centre for Genomic Regulation

Bioinformatics workflows

- Data analysis applications to extract information from (large) genomic datasets.
- Embarrassingly parallelisation, can spawn 100s-100k jobs over a distributed cluster.
- Mash-up of many different tools and scripts (dependencies).
- Complex dependency trees and configuration

Very fragile ecosystem

An example

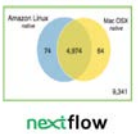


Steinbiss et al., Companion: a web server for annotation and analysis of parasite genomes, DOI: 10.1093/nar/gkw292

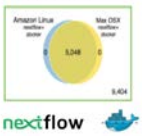
The same pipeline deployed in different environments produces different results (!)

Workflow Management System to the rescue

Nextflow enables reproducible computational workflows



nextflow



nextflow

Workflow managers enable reproducibility

Reproducible, scalable, and shareable analysis pipelines with bioinformatics workflow managers

Laura Wratten, Andreas Wilm & Jonathan Göke

Nature Methods 18, 1161–1168 (2021) | Cite this article

The rapid growth of high-throughput technologies has transformed biomedical research. With the increasing amount and complexity of data, scalability and reproducibility have become essential not just for experiments, but also for computational analysis. However, transforming data into information involves running a large number of tools, optimizing parameters, and integrating dynamically changing reference data. Workflow managers were developed in response to such challenges. They simplify pipeline development, optimize resource usage, handle software installation and versions, and run on different compute platforms, enabling workflow portability and sharing. In this Perspective, we highlight key features of workflow managers, compare commonly used approaches for bioinformatics workflows, and provide a guide for computational and noncomputational users. We outline community-curated pipeline initiatives that enable novice and experienced users to perform complex, best-practice analyses without having to manually assemble workflows. In sum, we illustrate how workflow managers contribute to making computational analysis in biomedical research shareable, scalable, and reproducible.

Comparison with other workflow managers

Perspective | Published: 23 September 2021

Reproducible, scalable, and shareable analysis pipelines with bioinformatics workflow managers

Laura Wratten, Andreas Wilm & Jonathan Göke

Nature Methods 18, 1161–1168 (2021) | Cite this article

nextflow main features

Polyglot

BASH

Dependencies sandboxed in containers

CONDA

Uses dataflow programming paradigm

Version control

Support of multiple platforms

kubernetes

aws

slurm

Platform Computing

Task example

```
bwa mem reference.fa sample.fq \
      | samtools sort -o sample.bam
```

Task example

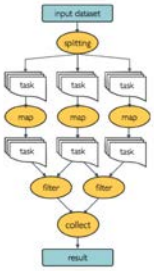
```
process align_sample {

    input:
    path ref_fasta
    path sample_fastq

    output:
    path 'sample.bam', emit: bam

    script:
    """
    bwa mem $ref_fasta $sample_fastq \\
          | samtools sort -o sample.bam
    """
}
```

Task composition

```
process align_sample {

    input:
    path ref_fasta
    path sample_fastq

    output:
    path 'sample.bam', emit: bam

    script:
    """
    bwa mem reference.fa $sample_fastq \\
          | samtools sort -o sample.bam
    """
}

process index_sample {

    input:
    path bam_to_idx

    output:
    path "${bam_to_idx}.bai", emit: bai

    script:
    """
    samtools index $bam_to_idx
    """
}

workflow {

    align_sample ( ref_fasta_ch, fastq_ch )
    index_sample ( align_sample.out.bam )

}
```

Dataflow paradigm

- Declarative computational model for parallel process executions
- Processes wait for data, when an input set is ready the process is executed
- They communicate by using dataflow variables i.e. async FIFO queues called channels
- Parallelisation and tasks dependencies are implicitly defined by process in/out declarations



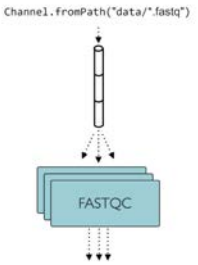
How parallelization works

```
samples_ch = Channel.fromPath("data/sample.fastq")

process FASTQC {

    input:
    path reads
    output:
    path 'fastqc_logs', emit: fastqc_ch

    """
    mkdir fastqc_logs
    fastqc -q ${reads} -f fastq -o fastqc_logs
    """
}

workflow {
    FASTQC ( samples_ch )
}
```

How parallelization works

```
samples_ch = Channel.fromPath("data/*.fastq")

process FASTQC {

   input:
   path reads
   output:
   path 'fastqc_logs', emit: fastqc_ch

   """
   mkdir fastqc_logs
   fastqc -q ${reads} -f fastq -o fastqc_logs
   """
}

workflow {
   FASTQC ( samples_ch )
}
```

Implicit parallelism



Supported platforms

Portability


```
nextflow run your-script.nf -with-docker your/image
```

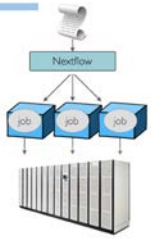
Portability

Portability

Portability

Portability

Containerisation



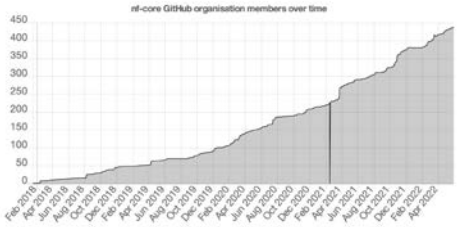
- Nextflow envisioned the use of software containers to fix computational reproducibility
- Mar 2014 (ver 0.7), support for Docker
- Dec 2016 (ver 0.23), support for Singularity

Nextflow turns modular

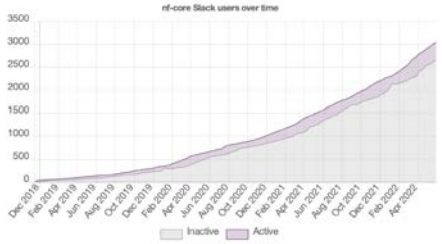
- DSL-2 is an extension of Nextflow syntax (released on version 20.07.1)
- Enables the definition of reusable modules and sub-workflows
- Pipeline can be now split in different files instead of having a huge script with all the logic
- Some changes to respect DSL-1 but don't change the core Nextflow concepts (e.g. channels can be reused without need of create multiple copies of the same channel)
- DSL2 became the default mode in the last stable release 22.04.1

What makes nextflow strong???

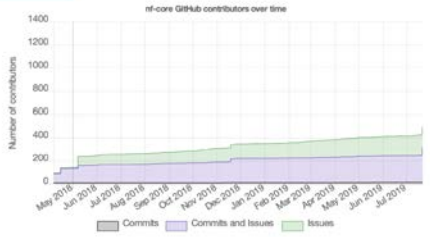
GitHub nf-core members over time



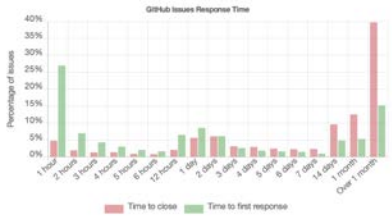
Slack members over time



nf-core GitHub contributors



Response time to pull requests and issues



nf-core community resources

https://nf-core.re/

https://github.com/nf-core/

https://nf-co.re/join/slack

https://twitter.com/nf_core

https://groups.google.com/forum/#!forum/nf-core

https://www.youtube.com/c/nf-core

nf-core/bytesize

All Tuesdays 13:00 CEST

Bytesize: resources to learn Nextflow

What nf-core is?

- A community of users and developers
- A curated set of analysis pipelines build using Nextflow
- A set of guidelines (standard)
- Helper tools

nf-core curated pipelines

36 RELEASED

24 UNDER DEVELOPMENT

6 ARCHIVED

nf-core guidelines

nf-core requirements

- Build using Nextflow
- Have a MIT license
- Software bundled using Docker/Singularity
- CI testing (include minimal test dataset)
- Pass nf-core lint tests
- Stable release tags (GitHub releases with associated changelog, and DOI using Zenodo)
- Common pipeline structure and usage
- Run in a single command (pipelines should not be split into multiple sub-pipelines)
- Comprehensive documentation
- A responsible point of contact

nf-core recommendations

- Software bundled with Bioconda (package the software using Bioconda if not already available)
- Explicit support for cloud environment
- Benchmarks from running on cloud environments
- Optimized output file formats (Standard format where possible, including CRAM)

Why having strict guidelines

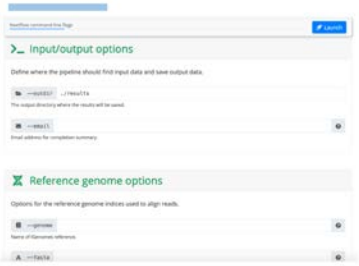
- Follow FAIR principles
- Adhere to current best practices in terms of computational reproducibility and interoperability
- Guarantee the portability between different computational infrastructures
- Enable a set of common features between pipelines (how they run, documentation, etc.)

Why having strict guidelines

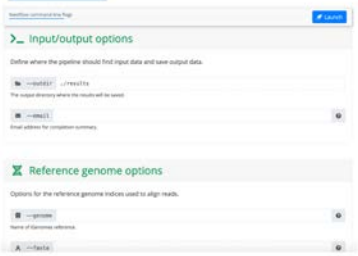
Pipelines work in a comparable manner

```
nextflow run <pipeline> -profile test,singularity
nextflow run <pipeline> -profile test,docker
nextflow run <pipeline> -profile test,conda
```

A common interface: e.g. web form



A common interface: e.g. web form



A package of helper tools

nf-core/tools

A python package with helper tools for the nf-core community

python — `pip install nf-core`

BIOCONDA — `conda install -c bioconda nf-core`

`docker pull nfcore/tools`

nf-core tools

nf-core turns DSL2

nf-core/rnaseq
nf-core/viralrecon
nf-core/fetchngs
nf-core/ampliseq
nf-core/bacass
nf-core/fetchngs
nf-core/bcellmagic
nf-core/cutandrun
nf-core/ampliseq
nf-core/rnaseq
nf-core/rnafusion
...

nf-core DSL2 concepts

MODULE: A process that can be used within different pipelines and is as a atomic as possible i.e. cannot be split into another module.
e.g. a module file containing the process definition for a single tool such as FastC

SUB-WORKFLOW: A chain of multiple modules that offer a higher-level functionality within the context of a pipeline.
e.g. a sub-workflow to sort, index and run some basic stats on a BAM file.

WORKFLOW: An end-to-end pipeline created by a combination of Nextflow DSL2 individual modules and sub-workflows.
e.g. from one or more inputs to a series of final inputs

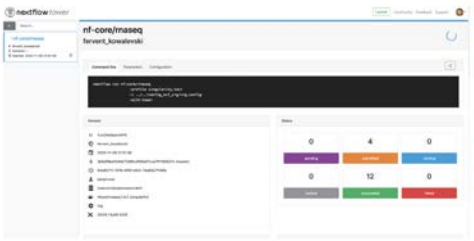
nf-core DSL2 modules

Nextflow tower

- Web user interface to interact with Nextflow
- An API to "talk" to pipelines
- Seamless configuration of cloud environments
- Enables to run pipelines in the cloud or HPC

nextflow tower
https://tower.nf
seqeralabs

Nextflow tower enables to launch and monitor your executions



Join the community

nextflow
https://nextflow.io

nf-core
https://nf-co.re/

https://nf-core/join slack

https://join.slack.com/t/nextflow/shared_invite/zt-11iwlxtw5-R6SNBpVksOJAx5sPOXNrZg slack

Demo

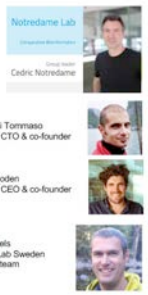
Thanks!!!

nf-core https://nf-core.re/

nextflow https://www.nextflow.io

https://www.nextflow.io/docs/latest/index.html

nextflow tower https://tower.nf



PRIORITY OF CONFIGS

1. Parameters specified on the command line (`--something value`)
2. Parameters provided using the `-params-file` option
3. Config file specified using the `-c my_config` option
4. The config file named `nextflow.config` in the current directory
5. The config file named `nextflow.config` in the workflow project directory
6. The config file `$HOME/.nextflow/config`
7. Values defined within the pipeline script itself (e.g. `main.nf`)

Example: use software bundle with Bioconda/Biocontainers

bioconda / packages / fastqc 0.11.9

A quality control tool for high throughput sequence data.

License: GPL >=3
Home: http://www.bioinformatics.babraham.ac.uk/projects/fastqc/
446619 total downloads
Last upload: 1 year and 13 days ago

BioContainers

docker

```
conda (params.enable_conda ? "bioconda::fastqc=0.11.9" : null)
container "${ workflow.containerEngine == 'singularity' && !task.ext.singularity_pull_docker_container ?
    'https://depot.galaxyproject.org/singularity/fastqc:0.11.9--0' :
    'quay.io/biocontainers/fastqc:0.11.9--0' }"
```

HOW TO CONTRIBUTE

1. Join the nf-core community

#new-pipelines

There should only be a single pipeline for a given data + analysis type

CREATE A PIPELINE FROM THE TEMPLATE

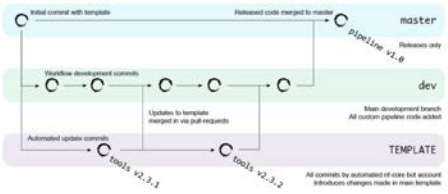
2. nf-core create

```
Workflow Name: awesomepipeline
Description: An awesome demo pipeline
Author: Jose Espinosa-Carrasco
INFO     Creating new nf-core pipeline: 'nf-core/awesomepipeline'
INFO     Initialising pipeline git repository
INFO     Done. Remember to add a remote and push to GitHub:
         cd /Users/joseespinosa/dev/nf-core-awesomepipeline
         git remote add origin git@github.com:USERNAME/REPO_NAME.git
         git push --all origin
INFO     This will also push your newly created dev branch and the TEMPLATE branch for syncing.
INFO     !!!!!! IMPORTANT !!!!!!

         If you are interested in adding your pipeline to the nf-core community,
         PLEASE COME AND TALK TO US IN THE NF-CORE SLACK BEFORE WRITING ANY CODE!

         Please read: https://nf-co.re/developers/adding_pipelines#join-the-community
```

Use the template!!

SOFTWARE DEVELOPMENT CYCLE

2.



CONFIGURATION FILES

- Default config (e.g. base.config)
  - Automatically loaded
  - Sensible default resources request
- Core profiles (e.g. docker, singularity, test) → **`-profile test,docker`**
  - Specify software packaging
  - Specify common presets
- Institutional profiles (nf-core/configs) → **`-profile CRG`**
  - Specify job submission for your institution cluster
  - Specify software packaging and other settings
  - Available for all nf-core pipelines
- Your local config files → -c flag e.g. **`-c my_local.config`**)
  - Custom resource requirements, user specific parameters

nextflow-schema.json

```
params.foo = "BovReg"
println ("Hello $params.foo !")
```

```
$ nextflow run main.nf --foo EuroFAANG
```

Hello EuroFAANG !

nextflow-schema.json

```
"foo": {
    "type": "string",
    "default": "BovReg",
    "description": "A grettings parameter"
}
```

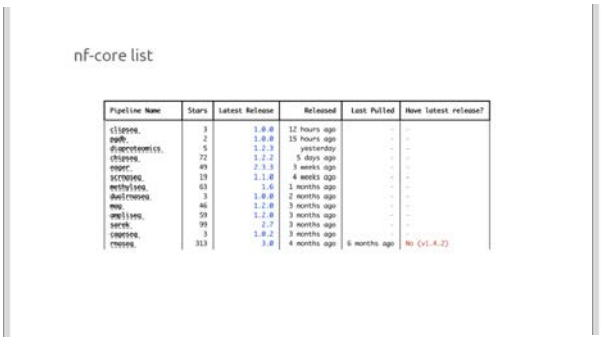
Automates
- User input validation
- Pipeline CLI help
- Documentation
- User interfaces

`nf-core schema build`

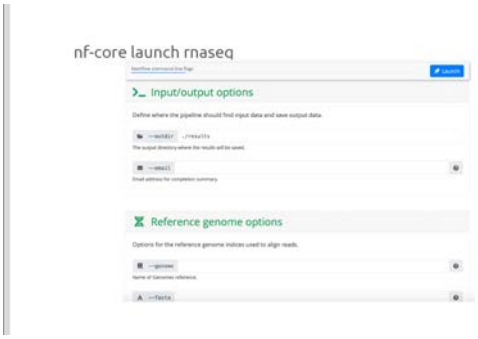

HOW TO CONTRIBUTE
Develop your code (Automatic CI tests + Community reviews)


HOW TO CONTRIBUTE
zenodo
Pipeline released with a DOI per release

7,8 cm


nf-core list
Pipeline Name
Stars
Latest Release
Released
Last Pulled
Have latest release?


nf-core launch rnaseq


nf-core launch rnaseq
Input/output options
Reference genome options

Galaxy + WorkflowHub

Ignacio Eguinoa (VIB - ELIXIR BE)
30/05/2022

Agenda

- Galaxy
  - Overview
  - Tools
    - Templates
    - Ejemplos
  - Galaxy workflows
    - Editor
    - Ejecución
- WorkflowHub
  - Registrar un workflow
  - RO-Crate

¿Cómo usar la plataforma?

- Instalación local:
  - Clone https://github.com/galaxyproject/galaxy
  - ./run.sh
    - Default configuration
- Versiones distribuidas en contenedores Docker:
  - https://hub.docker.com/r/bgruening/galaxy-stable/
- Servidores públicos:
  - Usegalaxy.org
  - Usegalaxy.be
  - Usegalaxy.eu
  - Usegalaxy.es
  - Usegalaxy.org.au
  - ...

Galaxy Project

Provee una capa de abstracción para acceder y hacer uso de la infraestructura de cómputo. Una simple interfaz gráfica provee una capa de abstracción para:

- Ejecutar comandos en entorno Linux.
- Definir y ejecutar un workflow.
- Resolver dependencias de ejecución.
- Job scheduling.
- Data management.
  - Metadata + tipos de datos.
- Manejo de usuarios.

Galaxy es toda una plataforma/ecosistema de abstracciones y herramientas, pero el core se basa en 2 entidades: Herramientas y Workflows.

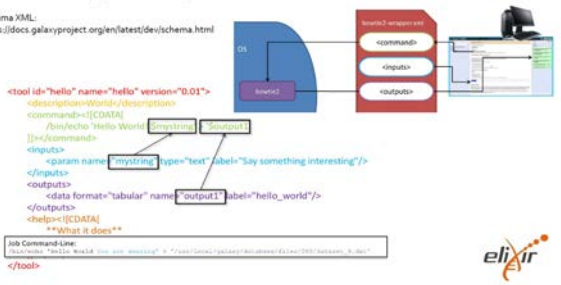
Tools: unidad atómica de ejecución.

- Listadas en el panel de herramientas (panel a la izquierda), organizadas en secciones.
- Módulos instalables: cada admin de un servidor decide qué herramientas instalar y poner a disponibilidad de usuarios.
- En Galaxy, "todo" es una herramienta: cargar datos locales, de repositorios remotos, enviar datos, procesamiento , linux commands, herramientas bioinformaticas, etc.
- Definidas mediante "tool wrappers" que proveen reproducibilidad.

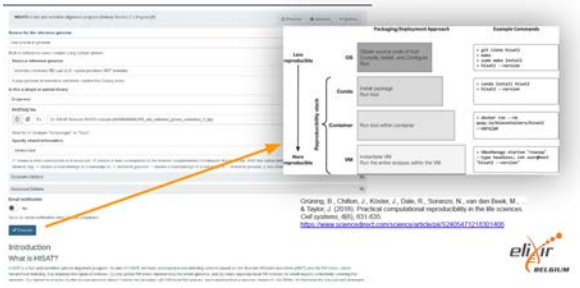
GUI de cada herramienta

Tool wrappers: templates para cada herramienta.

Schema XML:
https://docs.galaxyproject.org/en/latest/dev/schema.html



Reproducible job execution



¿Cómo definir las dependencias?

La sección de requirementos del wrapper es un indicador a la plataforma sobre qué dependencias deben estar disponibles en el entorno de ejecución

https://docs.galaxyproject.org/en/latest/dev/schema.html#tool-requirements

La plataforma puede estar configurada para resolver dependencias usando distintos sistemas y prioridades (mayor o menor reproducibilidad):

- Paquetes locales (deprecated)
- Conda (el admin debe configurar los canales y prioridades)
- Contenedores (e.g https://quay.io/biocontainers/fastqc:0.11.2--1)

Bioconda <-> Biocontainers

EDAM metadata

- https://edamontology.org/
- https://docs.galaxyproject.org/en/latest/dev/schema.html#tool-edam-topics

```
<!-- Example: this tool performs a 'Conversion' operation (http://edamontology.org/operation_3434) -->
<edam_operations>
    <edam_operation>operation_34344</edam_operation>
</edam_operations>
```

- Ayuda a enriquecer la metadata de la herramienta/proceso.
- Se usan tanto dentro (e.g para organizar el panel de herramientas) como en servicios externos (e.g workflowhub)

Repositorios + Cómo crear un wrapper para tu herramienta

Las herramientas son módulos instalables: wrapper + metadata

- Galaxy tiene su propio package manager interno que permite instalar paquetes desde https://toolshed.g2.bx.psu.edu/
  - Cada uno puede subir sus propios paquetes (también en https://testtoolshed.g2.bx.psu.edu/)
  - Se recomienda usar Github/Gitlab para mantener el código del paquete.
  - Galaxy tiene su propio repositorio (IUC -= nf-core) el cual sigue las mejores prácticas de desarrollo + CI test + updates automáticos, etc

Muchas herramientas ya tienen un wrapper en el ecosistema Galaxy

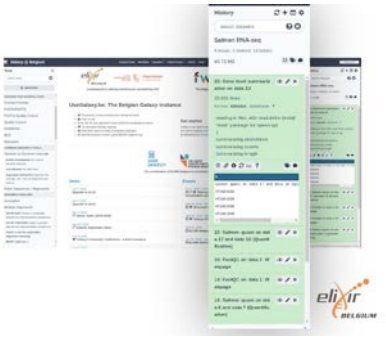
Para hacer un wrapper desde cero:

- https://training.galaxyproject.org/training-material/topics/dev/tutorials/tool-from-scratch/tutorial.html
- Crear las dependencias en bioconda-biocontainers
  https://github.com/bioconda/bioconda-recipes
- El schema del XML (sistema de tipos de datos, outputs, etc):
  https://docs.galaxyproject.org/en/latest/dev/schema.html
- Usar los repositorios publicos de Galaxy Project (IUC) como guia:
  https://github.com/galaxyproject/tools-iuc/tree/master/tools

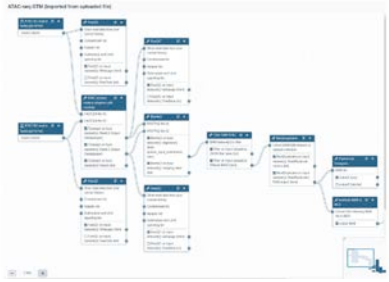
Communities sharing one coherent framework

- rna.usegalaxy.eu
- clipseq.usegalaxy.eu
- metagenomics.usegalaxy.eu
- hicexplorer.usegalaxy.eu
- cheminformatics.usegalaxy.eu
- proteomics.usegalaxy.eu
- imaging.usegalaxy.eu
- metabolomics.usegalaxy.eu
- ecology.usegalaxy.eu
- nanopore.usegalaxy.eu
- singlecellomics.usegalaxy.eu
- humancellatlas.usegalaxy.eu
- virology.usegalaxy.eu
- climate.usegalaxy.eu
- streetscience.usegalaxy.eu

Historial del análisis



Graphical workflow editor

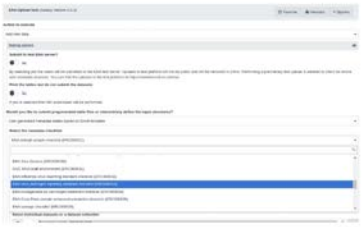


Compartir workflows dentro de una misma instancia

Formato de serialización de workflows

- Generado automáticamente a partir del editor de workflow.
- JSON based (formato v1)
  - YAML based (formato v2)
- Referencias al id de la herramienta -> no todos los detalles son exportados.
- Compartir inter-instancias de Galaxy: debe tener las mismas herramientas instaladas.

Submission to public repositories

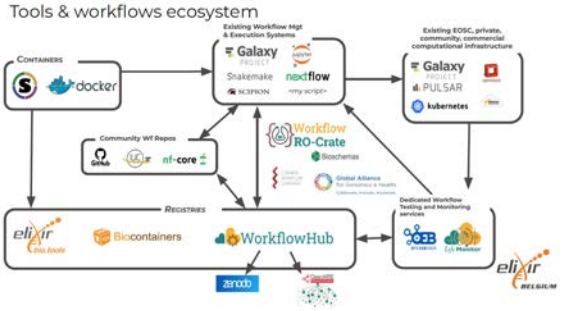


- Simplified submission interface.
- Multiple metadata input options.
- Include the submission automatically in your analysis.

Demo

- Workflow input datasets
- Parametros estáticos: https://training.galaxyproject.org/training-material/topics/galaxy-interface/tutorials/workflow-parameters/tutorial.html
- Descargar workflows
- **Ejecutar workflows**

Tools & workflows ecosystem



WorkflowHub.eu: a FAIR workflow registry

Beta release Sept 2020, sponsored by EOSC-Life

Perpetual Development in the open by an **open community**

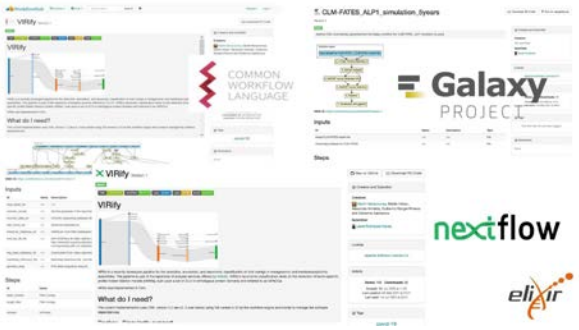
- Registering on behalf of makers
- Regular releases of new features
- Agile revisions of features

Workflows at the centre, as entry point for end-users

Open to any workflow and WfMS

- Open to workflows from **all disciplines** and **any country**, currently already spanning
  - Life Science
  - Biodiversity
  - Climate change
  - Industrial Biotech
- Workflow management system **agnostic**
- Workflows may remain in their **native repositories** in their **native form**
- Registry & repository functionality

SCIPION
jupyter Galaxy PROJECT
nextflow
Snakemake



Representación interna de un Workflow en Workflowhub

- Internamente WorkflowHub intenta llevar todos los workflows a un único estándar: CWL Abstract, con el objetivo de aumentar la interoperabilidad.
  - Lista los pasos y orden de ejecución
  - Describe inputs/outputs
  - No especifica cómo generar el comando que cada paso debe ejecutar (orden de parámetros, lógica, etc)
- Esta representación estándar permite(-iría):
  - Comparar la estructura de workflows independientemente de su lenguaje inicial.
  - Buscar un workflow usando una estructura abstracta o metadata (e.g procesos EDAM, inputs, tags, ..) y obtener la implementación en distintos lenguajes.

Community driven standards to describe workflows and metadata

COMMON WORKFLOW LANGUAGE
Canonical workflow description
Native or Abstract CWL
https://pypi.org/project/galaxy2cwl/

Bioschemas
Metadata for registration and discovery
ComputationalWorkflow and FormalParameter
ComputationalTool
Schema.org profile and types
https://bioschemas.org/profiles/ComputationalWorkflow/

Global Alliance for Genomics & Health
API Exchanging Tools and Workflows
GA4GH Tool Registry Service (TRS) API

Workflow RO-Crate

Workflow-RO-crate Profile : Specialisation of RO-Crate to package an executable workflow with all necessary documentation.

RO-Crate contains **files or references to files in native repository,** documentation, tests, examples

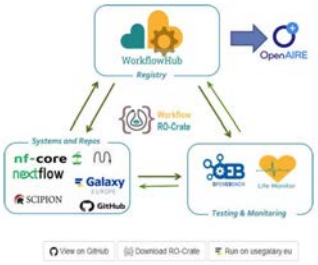
Exchange format for WorkflowHub

Integration with Other Services and WfMS platforms

- Register a workflow directly as uploaded RO-Crate
- Register a workflow file, abstract CWL, and diagram, provide additional metadata and license information, and have an RO-Crate made for you
- Download an RO-Crate

Coming soon

- API method for submitting RO-Crates to WorkflowHub.
- GitHub action to manage a repository contained workflows.



Representación externa: RO-Crate (Research Object Crate)

A method of organizing file-based data with associated metadata, using linked data principles, in both human and machine readable formats, with the ability to include additional domain-specific metadata.

- The core of RO-Crate is a JSON-LD file, the *RO-Crate Metadata File*, named ro-crate-metadata.json. This file contains structured metadata about the dataset as a whole (the *Root Data Entity*) and, optionally, about some or all of its files.
- Schema.org is the base metadata standard for RO-Crate.
  - Schema.org was chosen because it is widely used on the World Wide Web and supported by search engines, on the assumption that discovery is likely to be maximized if search engines index the content.
  - https://www.researchobject.org/ro-crate/1.1/context.jsonld

RO-Crate

Workflow RO-Crate

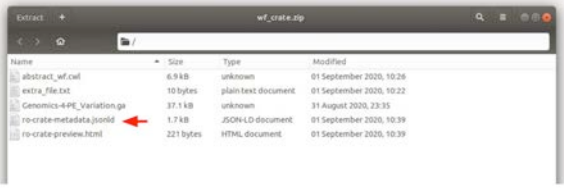
Workflow specific schema

Schema.org es extensible:

- Thing > CreativeWork > SoftwareSourceCode > ComputationalWorkflow
- https://bioschemas.org/profiles/ComputationalWorkflow/1.0-RELEASE
  - Minimally needs to describe: author, input, output, version...

Bioschemas

Ejemplo de workflow RO-Crate (*.zip)



What was done + work in progress

- Python implementation of RO-Crate:
  https://github.com/ResearchObject/ro-crate-py
- Entry point to extract workflow details from Galaxy:
  https://github.com/galaxyproject/galaxy/pull/9407
- Use the CWLProv profile (https://github.com/common-workflow-language/cwlprov/#) to export the provenance of a workflow run:
  - intermediate files + logs + generatedBy + executionTime + ...
  - Based on general provenance ontology (https://www.w3.org/TR/2013/REC-prov-o-20130430/)
  - Similar to: https://academic.oup.com/gigascience/article/8/11/giz095/5611001

¿Cómo registrar tu workflow?

Archivo principal del workflow:

- Upload de archivos individuales
- Link a repositorios Git
- Upload de RO-Crates

Metadata:

- Completar formulario de upload
- RO-Crate

Handling Workflow Lifecycles

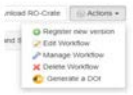
RO-Crates upload is a starting point

Enhancing

- Need to include references to **Datasets** and **Publications** associated with the workflow in WorkflowHub
- Update and enhance uploaded RO-Crates with metadata provided by the WorkflowHub
- RO-Crate download is enriched

Versioning of workflow files or RO-Crate

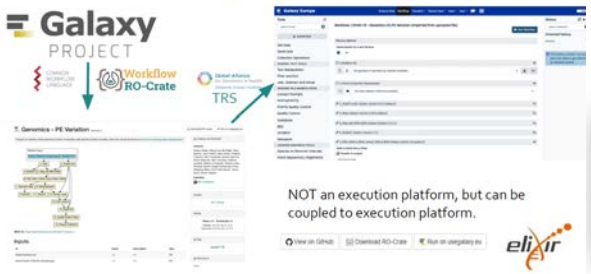
- Register a new version though the web interface
- Control over visibility of past versions (unless they have a DOI, in which case they are public)



Publishing

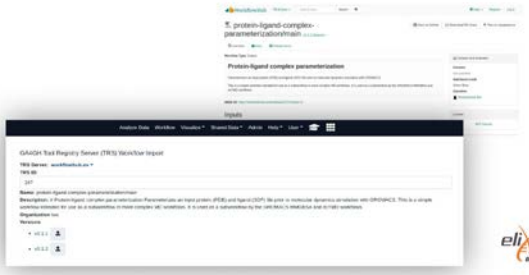
- Generate DOI for a public workflow entry
- Simple one-click to create
- DOI is tied and resolves to a specific version
- Citation is available to copy

WorkflowHub - usegalaxy.eu integration

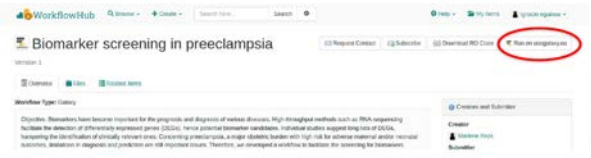


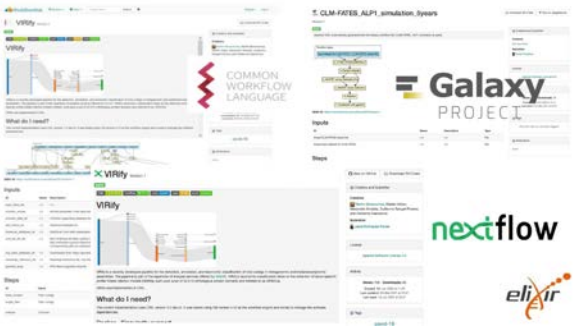
NOT an execution platform, but can be coupled to execution platform.

Ejecutar workflows de WorkflowHub en Galaxy.



Ejecutar workflows de WorkflowHub en Galaxy.





Join the WorkflowHub Club

We gratefully acknowledge the WorkflowHub Club, Bioschemas Group, RO-Crate Group, CWL Community and our WfMS partners in Galaxy, Snakemake, Nextflow, CWL, SCIPION.

- **WorkflowHub Club**
  https://about.workflowhub.eu
- **Bioschemas**
  https://bioschemas.org/groups/Workflow/
- **Research Object/ RO-Crate**
  https://www.researchobject.org/
- **Common Workflow Language**
  http://commonwl.org

https://workflowhub.eu

Acknowledgements

EOSC-Life receives funding from the European Union's Horizon 2020 Research and Innovation Programme under grant agreement number 824087.