Guía de localización y traducción de rOpenSci

Yanina Bellini Saibene, Paola Corrales, Elio Campiteli, Maëlle Salmon

Tabla de contenidos

Prefacio

Te damos la bienvenida!

Este libro explica todos los pasos y herramientas que intervienen en la localización de nuestros materiales, incluidas las traducciones. Diseñamos, construimos y probamos un proceso, para mantener versiones en diferentes idiomas de nuestros materiales, que toma ideas de nuestro sistema de revisión por pares de software y del mantenimiento de proyectos de código abierto.

Sabemos que la revisión por pares de código garantiza la calidad de nuestro software; por lo tanto, con el mismo objetivo de garantizar la calidad de nuestros contenidos en distintos idiomas, implementamos un sistema para revisar y mantener la localización de nuestras guías, libros, articulos de blog y otros contenidos.

También sabemos que utilizando roles (como personas que revisan o mantienen) y herramientas (como GitHub y *pull request*) conocidas por nuestra comunidad, hacemos que este proceso sea más fácil de comprender, aplicar y, por tanto, también sea más fácil para contribuir.

Público al que va dirigido

Este libro es una guía para personas a las que les gustaría *localizar materiales de rOpenSci* o *contribuir a mantener el material localizado*. También está dirigido a personas que tienen que localizar material en otra comunidad o grupo. No necesitas dedicarte profesionalmente a traducir o desarrollar software para contribuir a este esfuerzo. Las siguientes personas tipo son ejemplos de los tipos de personas que constituyen nuestro público objetivo.

Ana es una estudiante colombiana que realiza su doctorado en una universidad de Argentina. Trabaja con datos geoespaciales. No es experta en programación, pero asistió a uno de los talleres de The Carpentries para científicas y utilizó algunos de los paquetes de la suite rOpenSci para manejar y acceder a datos geoespaciales para su tesis. El idioma fue una de las barreras a las que se enfrentó cuando aprendió a programar. Ana quiere contribuir traduciendo material de rOpenSci a su lengua materna para que otras personas puedan acceder a este conocimiento con menos esfuerzo. Este libro le mostrará cómo hacer una nueva traducción y cómo revisar una ya existente.

Francesco enviará sus paquetes al proceso de revisión de rOpenSci en español. Sabe que esto garantizará la calidad de su software y también facilitará su publicación en JOSS, lo que les dará créditos académicos por su trabajo en su universidad. Lee la versión en castellano del libro "Paquetes rOpenSci: Desarrollo, mantenimiento y revisión por pares" para preparar el envio de su paquete a revisión. Si su experiencia es buena, se ofrecerá para revisar de forma voluntaria en su lengua materna y contribuir al libro. Esta guía le explicará cómo convertirse en una de las personas que colabora y mantiene material traducido y la infraestructura que permite hacer las traducciones.

Bauti es cofundador de un grupo de usuarios en su ciudad de Brasil. Con un par de compañeros de universidad, enseña y aprende juntos a programar y hacer análisis reproducibles utilizando diferentes lenguajes de programación. Utiliza el material de rOpenSci en portugués durante sus talleres y le gustaría tener el material de otros lenguajes de programación, en portugués. Se pregunta cómo gestiona rOpenSci las traducciones. Este libro le mostrará cómo organizamos la traducción y qué herramientas utilizamos para que pueda aplicarlas en su comunidad.

Lo que aprenderás

La guía contiene:

- Por qué localizamos, traducimos y usamos publicaciones multilingües.
- Guía general con la infraestructura y las funciones, el proceso técnico paso a paso y las consideraciones sobre la localización, incluidos los términos técnicos que debemos traducir y los términos técnicos que no traduciremos.
- Guías específicas de cada idioma con consideraciones sobre la localización.
- Glosarios específicos de cada idioma.

En rOpenSci, utilizamos R como lenguaje de programación principal y GitHub para alojar nuestro código, incluidos nuestros paquetes, libros, página web y proceso de revisión. El proceso técnico paso a paso se refiere al trabajo con GitHub, como hacer revisar y comentar *pull requests* y crear *issues*. También explica cómo utilizar el paquete {babeldown} que facilita el trabajo con traducciones.

Las consideraciones sobre la localización y los glosarios son específicos de cada idioma y detallan los acuerdos alcanzados por la comunidad sobre cómo localizar y traducir el material. Este capítulo es diferente para cada idioma presente en esta guía.

Esperamos que la guía te resulte útil y clara, y agradecemos tus sugerencias en la lista de *issues* de este libro. Consulta Sección Contribución y Reutilización para saber más sobre cómo contribuir.

¡Feliz localización!

Contribuir y reutilizar

Este libro es un documento vivo. Si quieres contribuir a este libro, las correcciones, adiciones y sugerencias son muy bienvenidas. Todas las personas cuyo trabajo se incluya serán acreditadas en los agradecimientos. Consulta el repositorio de GitHub, en particular las directrices de contribución y nuestro Código de Conducta, para obtener más información sobre cómo contribuir.

Puedes citar este libro:

Guía de localización y traducción de rOpenSci v1.0. Yanina Bellini Saibene, Paola Corrales, Elio Campiteli, Maëlle Salmon (Noviembre, 2023)

¡Muchas gracias!

También puedes leer el versión PDF (en Inglés) de este libro.

Atribuciones

Estas directrices de traducción se basan en la experiencia y los documentos generados durante la traducción de Enseñar Tecnologia en Comunidad.

1 Motivación

En rOpenSci trabajamos para transformar la ciencia mediante datos abiertos, software y reproducibilidad. Proporcionamos apoyo mediante nuestra comunidad, normas e infraestructura para que las personas que investigan, hacen ciencia y desarrollan nuestro software de investigación y trabajan en R desarrollen, mantengan y publiquen software científico de código abierto de alta calidad. Además, desarrollamos y mantenemos documentación y recursos de alta calidad para apoyar estas actividades.

También creemos que la ciencia y el software de investigación que creamos para apoyarla, deben servir a todas las personas en nuestras comunidades, lo que significa que tiene que ser sostenible, abierto y construido por y para todos los grupos. Sin embargo, en la actualidad existe una consternadora falta de diversidad en las comunidades científicas y de código abierto en general. Esa falta de diversidad es potencialmente perjudicial para la sostenibilidad, utilidad y productividad de los proyectos.

rOpenSci está llevando a cabo una serie de actividades y proyectos para transformar la ciencia y garantizar que sirva a todas las personas que son parte de nuestra comunidad. Uno de esos proyectos es nuestro proyecto de publicación multilingüe.

1.1 El lenguaje como barrera

El inglés es la *lingua franca* para la ciencia, lo que supone una barrera importante para quienes no hablan inglés y que desean incorporarse a este campo. También es la lengua predominante del movimiento de software libre, el contenido y las interacciones de la comunidad, lo que hace que *acceso a [contenido] en tu lenguage* una de las *barreras medioambientales a la equidad en el software libre*[1].

La recomendación de la UNESCO sobre ciencia abierta subraya la necesidad de superar las barreras lingüísticas para alcanzar los valores fundamentales de la ciencia abierta y los principios rectores de Equidad e imparcialidad, Diversidad e inclusión, Igualdad de oportunidades y Colaboración, participación e inclusión[2].

[1] Hilary Carter and Jessica Groopman, "Diversity, Equity, and Inclusion in Open Source: Exploring the Challenges and Opportunities to Create Equity and Agency Across Open Source Ecosystems", foreword by Jim Zemlin, The Linux Foundation, December, 2021, https://www.linuxfoundation.org/research/the-2021-linux-foundation-report-on-diversity-equity-and-inclusion-in-open-source

[2] UNESCO. Recomendación de la UNESCO sobre la ciencia abierta. París, Francia; 2021.

Un estudio reciente[3] cuantificó las consecuencias de las barreras lingüísticas en el desarrollo de la carrera de las personas que investigan. El articulo presenta que para aquellas personas que investigan y cuya lengua materna no es el inglés necesitan

- hasta **91% más de tiempo para leer** publicaciones científicas en inglés,
- hasta un **51% más de tiempo para escribirlos** y sus trabajos serán **rechazados 2,6 veces más**. Si son aceptados, tendrán que **revisarlo 12,5 veces más**,
- cuando presentan en inglés, necesitan hasta **94% más de tiempo** para preparar sus presentaciones. Además **suelen evitar** conferencias y presentaciones orales en inglés.

Además, el informe de la Fundación Linux afirma que "el dominio del inglés es una métrica con la que se puede juzgar el rendimiento y la personalidad"[4].

Estos estudios también destacan que "la magnitud de la desventaja parece estar muy por encima del nivel que puede superarse con un esfuerzo individual"[5].

Al desarrollar la infraestructura técnica y social para publicar recursos multilingües y publicar nuestros propios recursos en varios idiomas, podemos reducir estas barreras por *aumentar el acceso al conocimiento* y *democratizar el acceso a recursos de calidad* y de este modo *aumentar el potencial de contribución de los individuos* a proyectos de software y ciencia abierta.

1.2 Traducciones y localizaciones impulsadas por la comunidad

Las traducciones y localizaciones impulsadas por la comunidad son esfuerzos realizados porlas personas que pertenecen a una comunidad para disponer de un recurso en la lengua de su elección. La comunidad organiza y acuerda distintos aspectos del proyecto de localización[6].

La comunidad hispanohablante de R ha sido muy activa y ha crecido en los últimos años, y ha emprendido diversas actividades de traducción de materiales técnicos como libros, hojas de trucos, guías y conjuntos de datos.

---

[3] Amano T, Ramírez-Castañeda V, Berdejo-Espinola V, Borokini I, Chowdhury S, Golivets M, et al. (2023) The manifold costs of being a non-native English speaker in science. PLoS Biol 21(7): e3002184. https://doi.org/10.1371/journal.pbio.3002184

[4] Hilary Carter and Jessica Groopman, "Diversity, Equity, and Inclusion in Open Source: Exploring the Challenges and Opportunities to Create Equity and Agency Across Open Source Ecosystems", foreword by Jim Zemlin, The Linux Foundation, December, 2021, https://www.linuxfoundation.org/research/the-2021-linux-foundation-report-on-diversity-equity-and-inclusion-in-open-source

[5] Amano T, Ramírez-Castañeda V, Berdejo-Espinola V, Borokini I, Chowdhury S, Golivets M, et al. (2023) The manifold costs of being a non-native English speaker in science. PLoS Biol 21(7): e3002184. https://doi.org/10.1371/journal.pbio.3002184

[6] Yanina Bellini Saibene y Natalia Soledad Morandeira. Ciencia de Datos Multilingüe: Diez consejos para traducir contenido de ciencia y la tecnología Chapter at Our Environment.A collection of work by data designers, artists, and scientists. ISBN:979-8-218-20191-3. http://datasciencebydesign.org/blog/multilingual-data-science

En 2017, varias R-Ladies latinoamericanas comenzaron a traducir el Código de Conducta de R-Ladies y sus Normas y Directrices al español.

En 2018, la comunidad R de América Latina tradujo colectivamente el libro R para la Ciencia de Datos al español. Esto incluyó la traducción de todos los conjuntos de datos utilizados en el libro, que se recopilaron en el paquetes datos convirtiéndolo en una excelente herramienta para la enseñanza.

La comunidad continuó con la traducción de Enseñar tecnología en comunidad y contribuyó a las traducciones al español de las Hojas de trucos de Posit, lecciones de The Carpentries y de The Programming Historian.

Impulsado por esta activa y creciente comunidad hispanohablante, rOpenSci pilotó con éxito nuestra primera revisión por pares en español en la que el envío, las revisiones y las respuestas editoriales estaban todas en español.

Estos trabajos y las experiencias comunitarias previas crearon las condiciones para que iniciáramos nuestra publicación multilingüe en español. El español es la segunda lengua materna más hablada del mundo y una de las más extendidas geográficamente, siendo lengua oficial en muchos países[7].

1.3 Construir la comunidad

En rOpenSci, entendemos la revisión como una forma de construir comunidad y esperamos que tener nuestro proceso de revisión en español y portugués, nos permitirá seguir construyendo la comunidad en las regiones donde se habla estos lenguajes, aumentar el número de colaboradores y obtener devoluciones sobre cómo se pueden mejorar nuestras herramientas y procesos para servir mejor a estos participantes.

Además de utilizar nuestro material para aprender a ser una persona que puede desarrollar, mantener, revisar o editar, la gente también puede contribuir a través de la traducción. Este tipo de contribución suele ser un buen primer paso y la comunidad la reconoce como valiosa.

También esperamos que la documentación y las herramientas del proyecto sean útiles para ampliar este esfuerzo a otras lenguas y para otras comunidades y proyectos con esfuerzos de traducción.

1.4 Referencias

[7] Lista de lenguas por número de hablantes nativos. Consultado el 1 de diciembre de 2022. https://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers

2 Visión general

> "Encuentrate con las personas de tu comunidad donde están".

2.1 Introducción

rOpenSci proporciona apoyo comunitario, normas e infraestructura para que las personas que hacen ciencia y desarrollan software de investigación con R desarrollen, mantengan y publiquen software científico de código abierto de alta calidad. Además, desarrollamos y mantenemos documentación de alta calidad y recursos para apoyar estas actividades. Muchos de nuestros materiales se definen como *documentos vivos* lo que significa que se mejoran y actualizan constantemente.

La publicación multilingüe de documentación y recursos implica dos aspectos, internacionalización y localización[1]:

- La *internacionalización* se refiere a las soluciones tecnológicas que permite que el software se adapte a diferentes regiones sin requerir cambios de ingeniería en el código fuente. Es lo que nos *permite* localizar nuestros contenidos.
- La *localización* es el proceso de tomar un contenido en su forma original y convertirlo en algo accesible y adecuado para otra región, país o público. Esto incluye el idioma, los formatos de fecha, la moneda, las unidades de medida y la compatibilidad con distintos juegos de caracteres.

Además, la localización de documentos *vivos* tiene dos etapas bien definidas en las que intervienen distintos recursos:

Fase 1. En primer lugar, traducir el material.

Fase 2. mantener el material actualizado y sincronizado entre los diferentes idiomas[2].

[1] Internacionalización y localización. Accedido el 1 de noviembre de 2022. https://en.wikipedia.org/wiki/Internationalization_and_localization

[2] Yanina Bellini Saibene y Natalia Soledad Morandeira. Ciencia de Datos Multilingüe: Diez Consejos para Traducir Contenidos Científicos y Técnicos. Chapter at Our Environment.A collection of work by data designers, artists, and scientists. ISBN:979-8-218-20191-3.

La *traducción* suele ser el componente que más tiempo consume en estos esfuerzos[3].

2.2 Infraestructura técnica y organización del trabajo

Existen muchas soluciones y herramientas para internacionalizar y localizar contenidos y software. Por ejemplo, sistemas de gestión de traducción (Crowdin, Transifex, Weblate), traductores automáticos (Google Translate, DeepL), sistemas de control de versiones (GitHub, GitLab), lenguajes de marcado (LaTeX, Markdown) y herramientas para escribir estos lenguajes (Overleaf, Quarto).

Estas soluciones tecnológicas están en continua evolución. La elección de la tecnología influye en las posibilidades que creamos para las contribuciones, por lo que hay que procurar elegir una tecnología que reduzca al máximo las barreras a la participación.[4].

Desarrollamos nuestra infraestructura para nuestras localizaciones, utilizando las herramientas que mejor se adaptan a nuestro equipo, colaboradores y materiales y, por lo tanto, a nuestra comunidad.

- Creamos el paquete babeldown para realizar una primera traducción automática de los documentos, incluyendo un proceso para actualizar traducciones existentes.
- Creamos el paquete babelquarto para configurar y generar libros o sitios web multilingues con Quarto .
- Utilizamos proyectos de GitHub para hacer un seguimiento del progreso de cada proyecto de localización y de las personas que tienen diferentes funciones de contribución, como *revisores*, *editoras,* y personas encargadas del *mantenimiento*.
- También desarrollamos, y documentamos en estas directrices, un flujo de trabajo que sigue la misma idea de utilizar las herramientas que nuestra comunidad ya conoce y utiliza en sus actividades.

2.3 Como empezar con git y GitHub

Si recien estas iniciando a usar git y GitHub, te recomendamos iniciar con los siguientes recursos:

[3]Internacionalización y localización. Accedido el 1 de noviembre de 2022. https://en.wikipedia.org/wiki/Internationalization_and_localization

[4]Yanina Bellini Saibene y Natalia Soledad Morandeira. Ciencia de Datos Multilingüe: Diez Consejos para Traducir Contenidos Científicos y Técnicos. Chapter at Our Environment.A collection of work by data designers, artists, and scientists. ISBN:979-8-218-20191-3.

- Developing Software Together por Paola Corrales, Elio Campitelli y Yanina Bellini Saibene. Este taller fue desarrollado y enseñado para el programa de Campeonas/es de rOpenSci.
- Happy Git and GitHub for the useR. Este libro es una gran recurso para aprender a usar git y GitHub con R.

En la seccion de tranduccion y revision explicamos en detalle el flujo de trabajo que usamos en rOpenSci usando conceptos de git y GitHub con otras herramientas.

2.4 Aspectos generales de la fase 1 del proceso de traducción

El proceso de traducción comienza con una primera traducción automática utilizando DeepL via el paquete babeldown. Esto proporciona un primer borrador que luego es revisado por una persona, que corrigen los errores e incorporan los acuerdos de localización y traducción lingüística detallados en las Directrices específicas de cada idioma.

Para minimizar los errores y promover una visión amplia de la traducción, en rOpenSci pedimos que cada capítulo o sección pase al menos por dos revisiones hechas en serie (la primera revisa la traducción automática y la segunda revisa la primera revisión), seguidas de una revisión general del libro o documento en su conjunto. Se buscará, de ser posible, que las personas que revisen tengan origen en diferentes países, para poder considerar las diferentes formas en que se habla el idioma en todo el mundo.

El proceso de traducción y revisión se realiza en GitHub utilizando *pull requests* (consulta la sección de pull request para más detalles). Elegimos este flujo de trabajo porque es la infraestructura que utilizamos en nuestra comunidad. Todos nuestros paquetes y libros están alojados en GitHub y utilizan *issues* y *pull requests* en su desarrollo. Nuestro proceso de revision de software por pares tambien se realiza en GitHub utilizando las mismas herramientas. Además, esto permite que el proceso sea abierto para que otras personas puedan contribuir y aportar devoluciones.

Por supuesto, como en todas las áreas de rOpenSci, este proceso está sujeto a nuestro código de conducta para crear un entorno amigable y seguro.

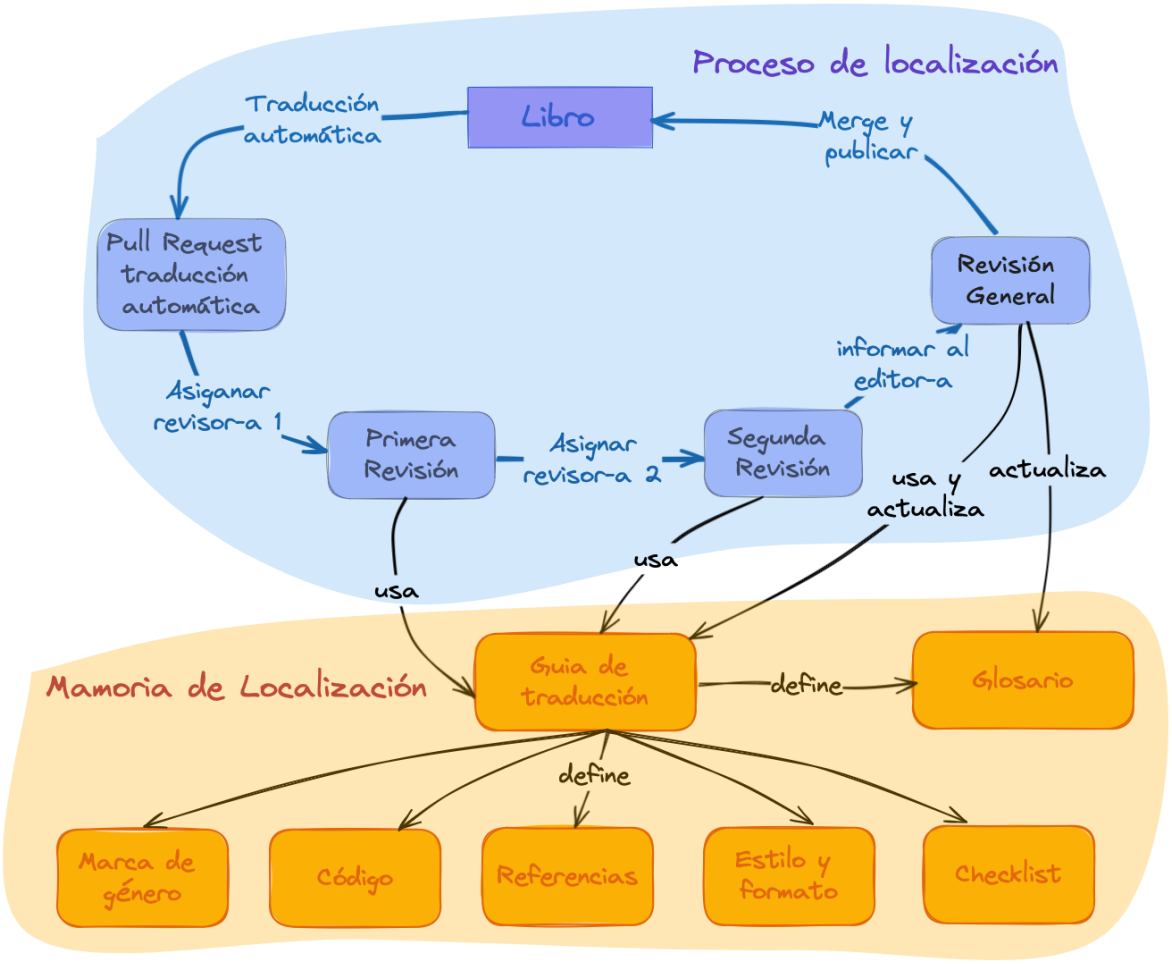


Figura 2.1: Proceso de localización de rOpenSci

2.5 Aspectos generales del proceso de traducción de la fase 2

En este caso estamos actualizando los cambios en el idioma original de un contenido que ya se ha traducido y tenemos que reflejar esos cambios en las diferentes traducciones.

El proceso es similar al de la etapa 1 de traducción, pero con algunas diferencias:

1. Tras el cambio de contenido en el idioma original, la persona que mantiene el proyecto de traducción crea una nueva traducción automática **solamente de los cambios** utilizando el paquete babeldown.
2. Luego revisa la *pull request* con la traducción automática y aplica los cambios necesarios.

3. Finalmente, hace un *merge* de la *pull request* con la traducción revisada.
4. La persona responsble actualiza las directrices de traducción específicas del idioma y el glosario si es necesario.

La entrada del blog de Maëlle Salmon Cómo actualizar una traducción con Babeldown explica este proceso con más detalle.

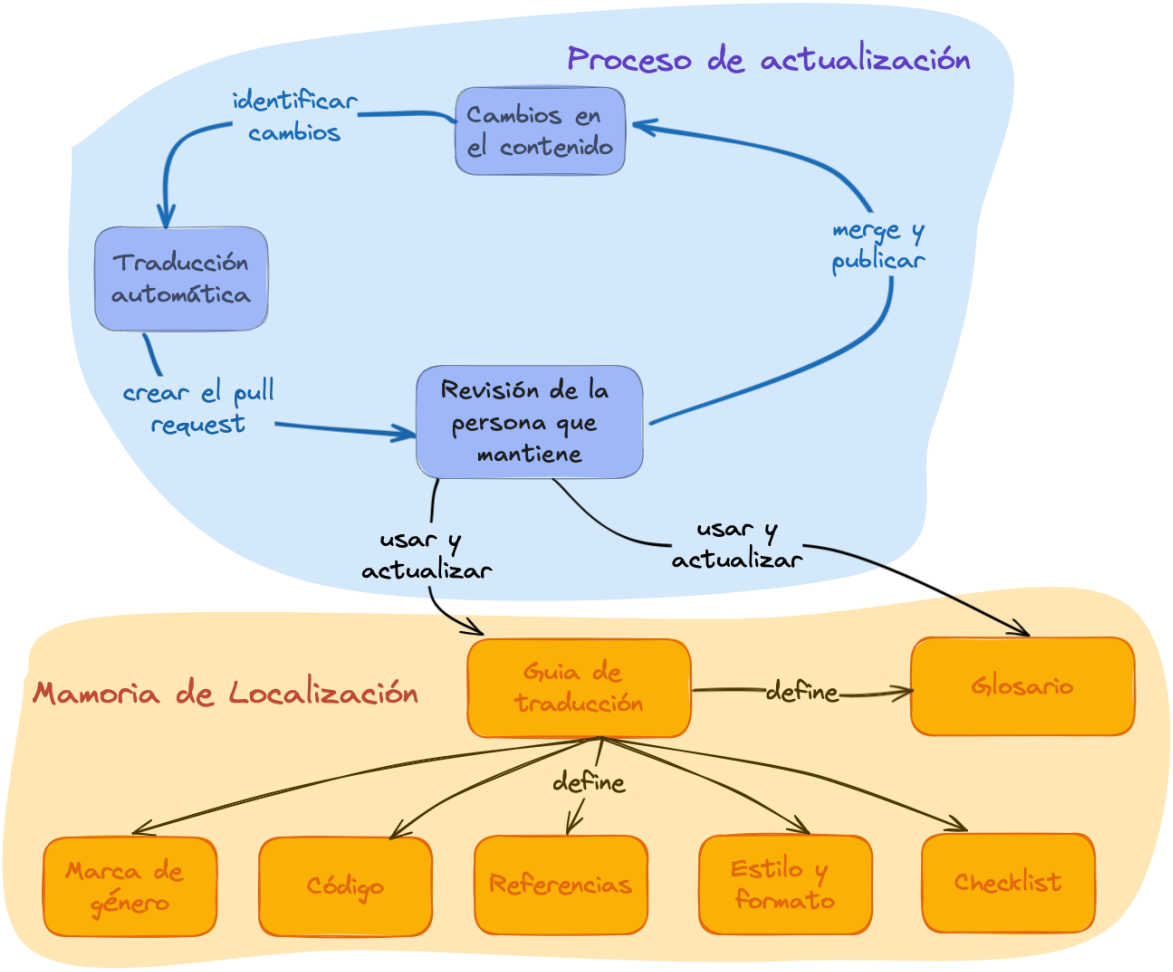


Figura 2.2: rOpenSci Update Localization Process

2.6 Referenciar y utilizar los materiales de localización

A continuación ofrecemos sugerencias para citar una traducción o hacer referencia a tu trabajo de traducción en un CV. Estos ejemplos se refieren específicamente a la traducción de “Paquetes

de rOpenSci: Desarrollo, mantenimiento y revisión por pares", pero puede aplicar estas recomendaciones a cualquier traducción.

2.6.1 Citar una traducción

El formato general es[5]:

> {Autores/as de la version en el idioma original}.{(año de publicación)}. {Título traducido}. {[Título original en el idioma original]} (Traducción a {idioma}: {Autoras/es de la traducción}). {DOI}. (Trabajo original publicado en {año de publicación})

El texto *Traducción a* y *Obra original publicada en* deben estar escritos en la lengua de las traducciones.

Ejemplo utilizando la traducción al español del libro "Paquetes de rOpenSci: Desarrollo, mantenimiento y revisión por pares":

> rOpenSci, Anderson, B., Chamberlain, S., DeCicco, L., Gustavsen, J., Krystalli, A., Lepore, M., Mullen, L., Ram, K., Ross, N., Salmon, M., Vidoni, M., Riederer, E., Sparks, A., & Hollister, J. (2021). Paquetes rOpenSci: Desarrollo, mantenimiento y revisión por pares [rOpenSci Packages: Development, Maintenance, and Peer Review] (Traducción al español: {nombre de las personas que tradujeron}) Zenodo. https://doi.org/10.5281/zenodo.6619350 (Trabajo original publicado en 2021)

2.6.2 Menciona tu contribución en tu CV

En general, puedes utilizar algo parecido a este formato:

> {Fecha de inicio} – {Fecha de finalización}. Localización colaborativa al {idioma} de "{nombre del material/libro}" (Edición general: {nombre completo}). Contribución como {role}. Detalles del proyecto: {Link al proyecto}.

Este es un ejemplo concreto de CV en inglés que hace referencia a un proyecto de traducción al español:

> Mayo 2020 – Julio 2022. Localización colaborativa al español del libro "rOpenSci Packages: Development, Maintenance, and Peer Review" (Lead editor Yanina Bellini Saibene). Contribución como revisor. Detalles del proyecto: https://github.com/ropensci/dev_guide.

[5]Basamos esta recomendación en el artículo de recomendaciones de la APA, Sección Libro, reeditado en traducción. También recomendamos incluir el título en el idioma original, siguiendo este ejemplo de Piaget (1950).

2.7 Referencias

3 Guía para traducir y revisar

3.1 Cómo colaborar

3.1.1 Contribuir a un proyecto activo

Si te interesa contribuir a un proyecto de traducción activo, antes de comenzar vas a tener que:

1) ponerte en contacto con las personas que mantienen el proyecto,
2) familiarizarte con el material que se va a traducir, y
3) leer esta guía, prestando especial atención a las directrices específicas del idioma.

Puede ponerse en contacto con un responsable del proyecto para expresar su interés a través del correo electrónico, Slack o GitHub. En la lista de proyectos activos, encontrarás información sobre cada proyecto, incluido las personas responsables de su mantenimiento. Cada proyecto de traducción tiene un canal en el Slack de rOpenSci donde se discuten y toman las decisiones de traducción, que se reflejan en las directrices específicas para cada idioma de esta guía y en un glosario. Si te unes a un proyecto de traducción, nos aseguraremos de que recibas una invitación al canal de Slack.

Es una buena idea preguntar cómo se están dividiendo las tareas de revisión para que todo el mundo sepa de qué es responsable.

No dudes en preguntar cualquier duda que tengas, como preguntas generales como «¿Cuál es el compromiso de tiempo esperado?», o preguntas técnicas como «¿Qué hago si me encuentro con problemas en GitHub?».

3.1.2 Proponer un nuevo proyecto de traducción

Empezar un nuevo proyecto de traducción requiere trabajo y compromiso con el proceso, pero puede ser una experiencia realmente gratificante.

Dado que se necesitan varias personas que revisen las traducciones y que debe haber al menos una persona activa manteniendoel proyecto , recomendamos que las personas interesadas en iniciar un nuevo proyecto de traducción inviten a su comunidad local a participar y formen un equipo inicial antes de comenzar el proceso.

Si tu idioma no está entre los idiomas con proyectos activos, ponte en contacto con el equipo de rOpenSci para proponerlo. Puedes enviar un mail a info@ropensci.org o abrir un *issue* en este repositorio.

Ten en cuenta que iniciar un nuevo proyecto requiere el compromiso para gestionar el repositorio donde se llevará a cabo toda la revisión. Pedimos a quienes propongan un nuevo proyecto de traducción que se comprometan a mantenerlo durante al menos 2 años, o a encontrar un quien lo mantenga si deben abandonarlo antes de los 2 años. Este es el mismo compromiso que pedimos a quienes mantienen nuestros paquetes.

3.2 Aspectos técnicos de la revisión

3.2.1 Directrices generales

En cada paso de este proceso, usamos las siguientes pautas:

1. Utiliza una voz conversacional en lugar de una voz formal o académica.
2. Si procede, especifica el dialecto o la variante lingüística regional utilizada. Por ejemplo, la traducción al español utiliza las convenciones latinoamericanas.
3. Intenta ser neutral en cuanto al género. Si la lengua en la que estás trabajando tiene un género gramatical fuerte, la traducción ajusta la redacción para evitar asignar un género. Cuando no se pueda evitar la marca de género, sigue las pautas especificas de cada lenguaje con respecto del uso de lenguaje inclusivo o no sexista.
4. Intenta ser idiomático. No te preocupes por las traducciones literales. Acerca el mensaje a su audiencia[1] eligiendo el texto y la forma en la lengua a traducir que mejor expresen el *significado* del fragmento del original.
5. Toma distancia para revisar. Cuando trabajamos mucho rato en un texto cuesta identificar errores de tipeo. Como sugerencia, una vez que termines la revisón del capítulo o sección deja pasar un tiempo (algunas horas o un día) antes de hacer la última lectura y enviarla para la siguiente revisión. Eso hace más fácil que salten a la vista este tipo de detalles y permite que quienes hagan la revisión se concentren en la calidad de la traducción más que en correcciones ortotipográficas. También es útil revisar el texto en una versión con formato (compilada como pdf o html), ya que seguramente algunos errores serán más visibles que en la versión cruda. Si tienes la posibilidad, una impresión en papel, tanto de la versión cruda como de la compilada, también ayuda a ver más claro los errores.

[1] Yanina Bellini Saibene y Natalia Soledad Morandeira. Ciencia de Datos Multilingüe: Diez Consejos para Traducir Contenidos Científicos y Técnicos. Chapter at Our Environment.A collection of work by data designers, artists, and scientists. ISBN:979-8-218-20191-3.

3.2.2 Organization

En esta sección usaremos como ejemplo la guia "Paquetes de rOpenSci: Desarrollo, mantenimiento y revisión por pares", pero todos los aspectos técnicos para hacer una revisión también aplican a otros materiales de la organización.

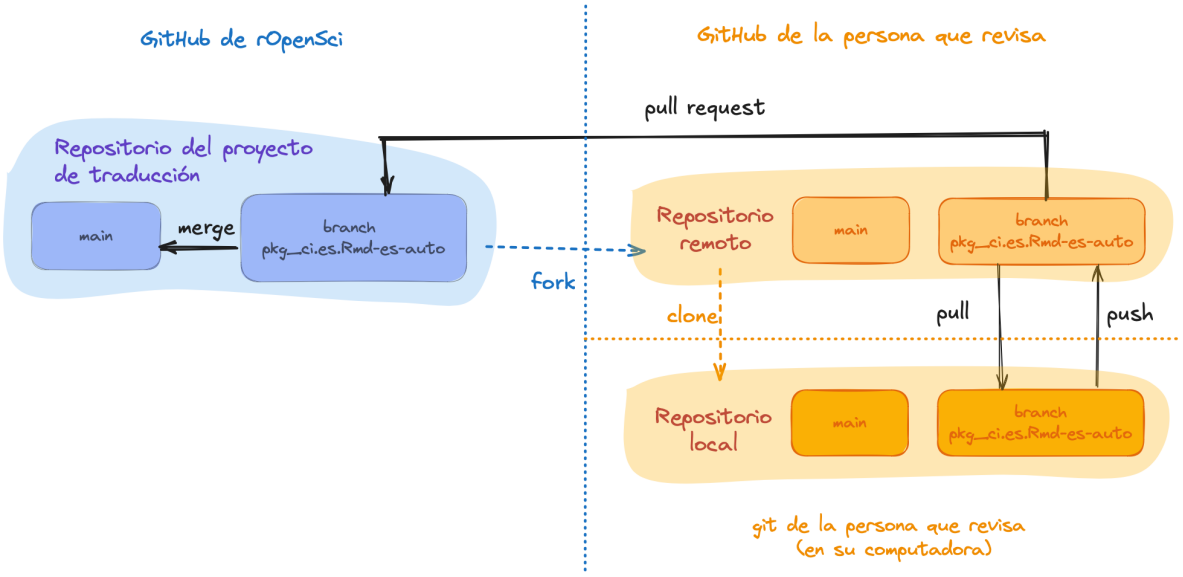
El código fuente que genera la guía vive en github.com/ropensci/dev_guide, y está organizada en archivos con formato .Rmd (RMarkdown) que contienen texto y a veces también código de R. En este caso también hay algunos archivos de configuración que requieren ser traducidos para que la guía en el nuevo lenguaje esté completamente traducida.

Cada capítulo tiene un archivo .Rmd por idioma, con un sufijo que identifica el idioma usando su código de dos letras (según ISO 639-1). Por ejemplo, el capítulo "Políticas de la Revisión por Pares de Software" tendrá su versión original en inglés en `softwarereview_policies.Rmd` y su versión traducida al español en `softwarereview_policies.es.Rmd`.

Si el proyecto de traducción está en marcha, encontrarás una serie de *pull request*, **uno por cada archivo a traducir**, generados con el paquete babeldown que contienen el texto traducido automáticamente. **Cada uno de estos *pull requests* tienen una *branch* asociada**. Por ejemplo, las traducciones del archivo `pkg_ci.Rmd` al español se encontrarían en el PR que fusiona la rama `pkg_ci.es.Rmd-es-auto`, y se llamaría "Add Spanish automatic translation pkg_ci.es.Rmd".

De hecho si hay muchos proyectos de traducción en marcha, es posible que te encuentres con muchos *pull requests* de diferentes archivos y diferentes lenguajes. Te recomendamos filtrar los *pull requests* usando la etiqueta asociada a tu lenguaje, por ejemplo la etiqueta para la traducción al español es "traducción ". También podes ver el estado de cada *pull requests* en el proyecto de GitHub correspondiente a tu lenguaje.

Por seguridad y organización, quienes realicen la revisión no van a tener premiso de escritura en el repositorio principal. De modo que quien está a cargo de mantener el proyecto deberá hacer un *fork* (que llamaremos **repositorio de trabajo**) y darle permiso de escritura a todas las personas que revisen o contribuyan. Este *fork* tendrá las mismas *branchs* que el repositorio principal; esto es clave ya la **revisión de cada archivo se realizará sobre la *branch* asociada a su traducción.** Esto significa que una revisión de la traducción al portugués ocurriría en la rama `pkg_ci.pt.Rmd-pt-auto`, mientras que la revisión de la traducción al español tendría lugar en la rama `pkg_ci.es.Rmd-es-auto`.



Una vez que hayas contactado con la persona responsable del proyecto y esté todo listo para que empieces a contribuir, lo primero que debes hacer es clonar el repositorio de trabajo. Una vez clonado, puedes empezar a trabajar con el repositorio localmente en tu ordenador.

Si aún no has tenido una discusión sobre lo que vas a revisar, puedes preguntar en el canal de Slack o comentar un *pull request* para preguntar si puedes encargarte de la revisión de ese capítulo. Te recomendamos encarecidamente que preguntes **antes** de empezar a trabajar en tu revisión.

Ahora si, ¡empecemos a revisar!

3.2.3 Revisión 1

El objetivo de esta primera revisión es hacer una lectura detallada de la traducción automática para:

- Asegurarse de que se mantiene el sentido de la frase o párrafo.
- Comprobar que los términos del glosario acordados por la comunidad se han utilizado de acuerdo con el contexto.
- Aplicar decisiones estilísticas siempre que sea posible. Por ejemplo, en español decidimos evitar el marcado de género de sustantivos y adjetivos parafraseando las frases o utilizando «las/los» si no es posible parafrasear.
- Comprobar que los bloques de código o el marcado Markdown no se hayan visto afectados.

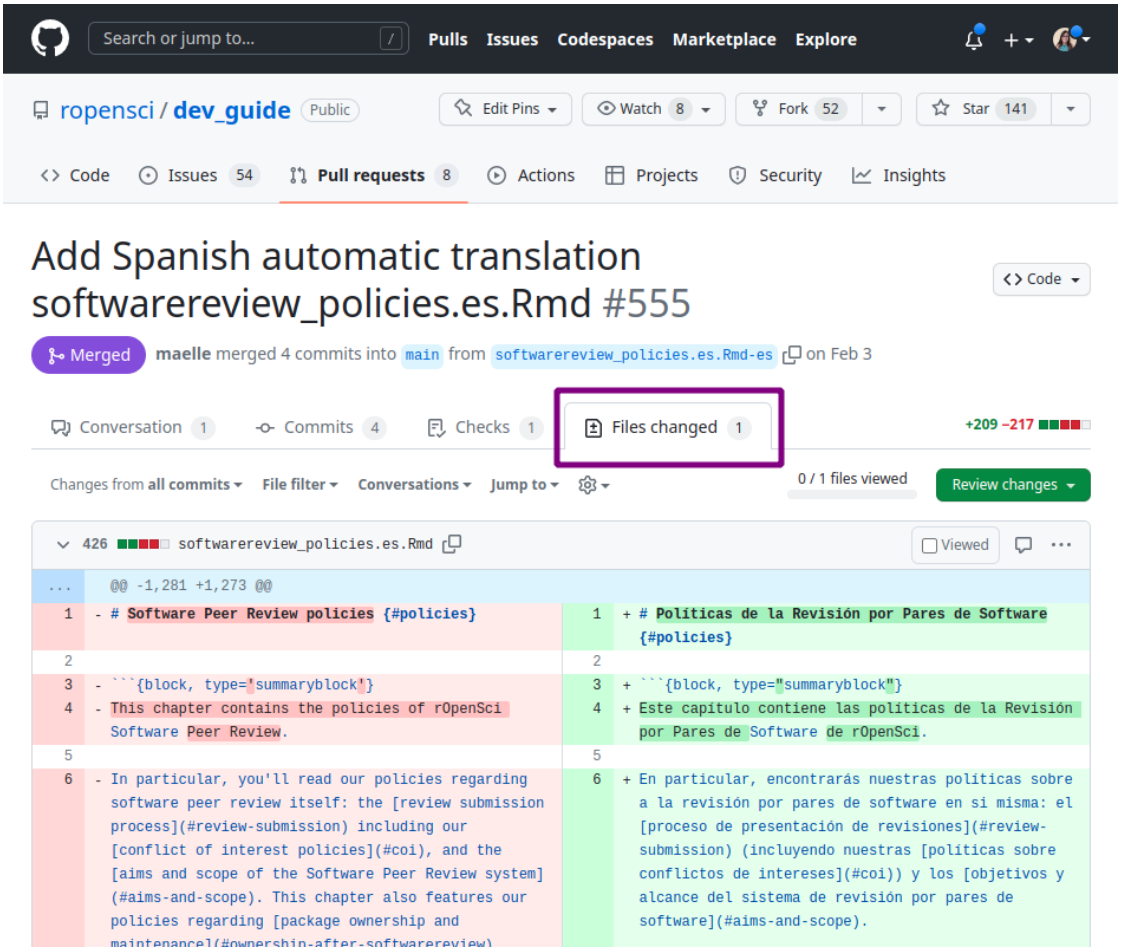
- Traducir los nombres de las variables y los comentarios visibles en los bloques de código cuando proceda.

A menudo quedarán algunas dudas y es posible que se pasen por alto algunas cosas. Las decisiones de estilo suelen ser las más difíciles de aplicar y a veces es necesario consultar con el resto del equipo o discutirlo más tarde con la persona que realizara la segunda revision.

Trabajando localmente en tu computadora, estos son los pasos a seguir para realizar una primera revisión de un archivo.

En tu repositorio local:

1. **Activá la *branch* asociada al archivo que vas a revisar**. En nuestro ejemplo, como estas revisando `softwarereview_policies.en.Rmd` la *branch* sería `softwarereview_policies.es.Rmd-es`‘.
2. **Abrí el archivo**, por ejemplo `softwarereview_policies.es.Rmd`. Es posible que veas cosas para mejorar inmediatamente, ya que la traducción automática hace un buen trabajo pero no es perfecta.
3. **Comenzá a revisar el texto.** Para esto te sugerimos tener la versión en inglés al lado, es posible que necesites volver a la versión original para entender el contexto de alguna frase o palabra que el traductor no haya podido traducir correctamente. Para esto puede usar la versión web del material que estás traduciendo o la pestaña *“Files changed”* del PR en el repositorio principal.



4. - **Envia tus cambios haciendo *commit*.** Muchas veces el contenido de los archivos será extenso y necesitarás bastante tiempo para completar la revisión. En estos casos te sugerimos hacer *commits* al menos luego de cada sesión de trabajo indicando en el mensaje hasta que línea del archivo llegaste. De esta manera, si necesitaras delegar la revisión en otra persona, esa persona puede revisar la historia del archivo y retomar donde vos dejaste.

5. - **Envia tus cambios al repositorio remoto haciendo *push*** Hasta ahí la revisión solo vive en tu repositorio local, el paso siguiente es hacer un *push* al repositorio de trabajo. Es muy importante que el *push* sea a la *branch* asociada al archivo en que estas trabajando, de lo contrario no podremos conectar tu revisión con la traducción original.

6. **Notificar a las personas responsables del proyecto o la segunda revision** Una vez que completaste tu revisión, es importante que avises a quien mantiene el proyecto o a la persona asignada a la revisión 2, si la hay.

3.2.4 Revisión 2

Te tocó la revisión 2, buenísimo.

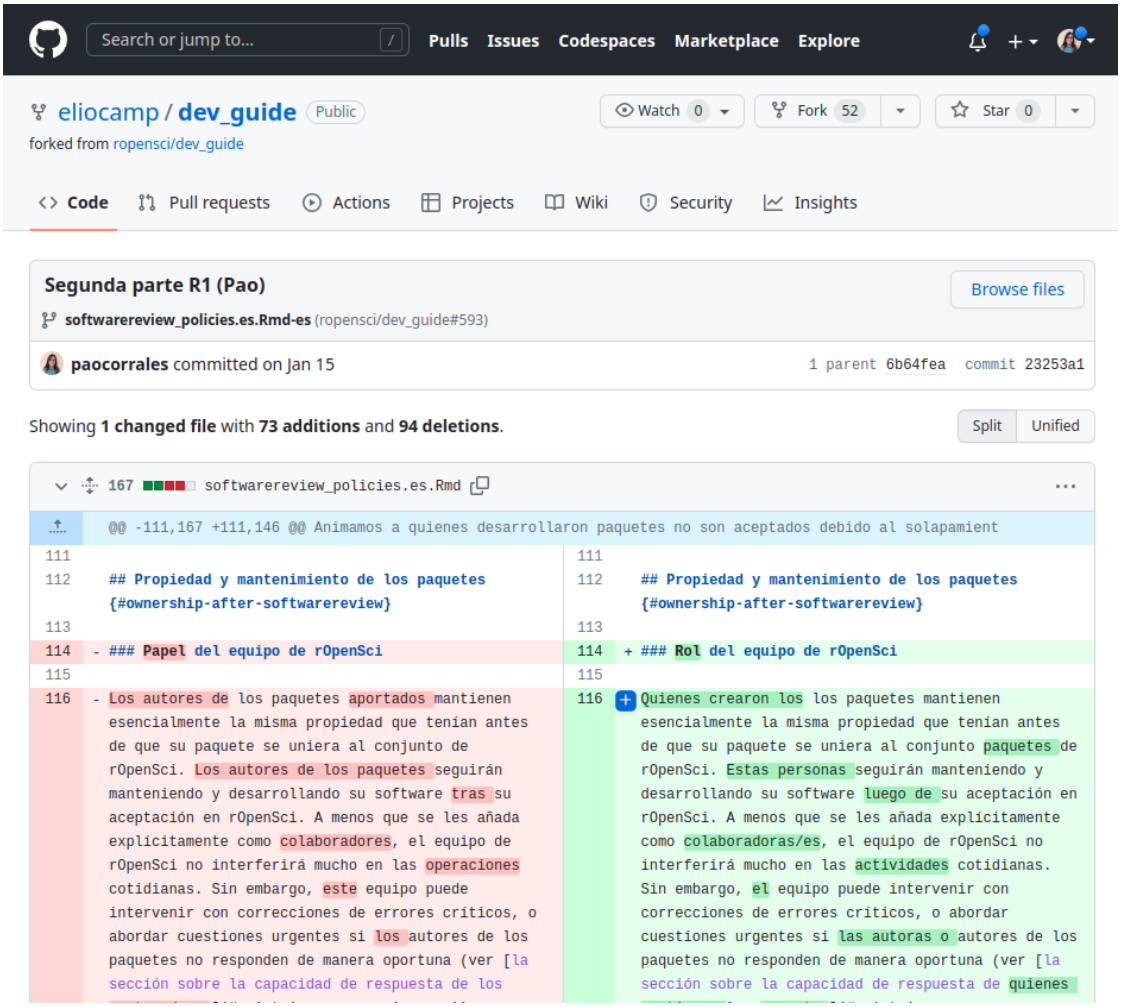
Quien hizo la revisión 1, a quien llamaremos **R1**, de este capítulo seguramente generó uno o mas *commits* con cambios sobre la traducción original. Esto significa que tu trabajo no arranca en cero. Esta segunda revisión busca construir sobre la primera para generar un texto que mantenga el contenido original pero que además suene natural en el nuevo lenguaje. Para esto hay que tener en cuenta:

- Que se respeten el uso de términos según el glosario y el contexto específico.
- Que se use el formato definido por la comunidad, por ejemplo, para las traducciones al español, usar formato *italizado* para palabras que se mantienen en inglés.
- Aplicar las decisiones de estilo que R1 haya dejado pasar. Por ejemplo, en español intentar reemplazar cualquier "las/los" por una frase sin marca de género.
- Chequear que las frases se entiendan y suenen bien evitando las traducciones literales.

Es posible que en el proceso de la revisión 2 necesites consultar con R1 sobre las decisiones que tomó o con el resto de la comunidad si hay alguna frase o término que te genera dudas. Es posible que luego de esto sugieras incorporar nuevos términos al glosario o actualizar esta guía de traducción.

Estos son los pasos a seguir para realizar la segunda revision, trabajando en tu repositorio local:

1. **Activa la *branch* asociada al archivo que vas a revisar.** Por ejemplo, si estas ayudando a traducir `softwarereview_policies.en.Rmd`, tenes que buscar por la rama `softwarereview_policies.es.Rmd-es`.
2. **Descarga todos los cambios.** Haz un *pull* para descargar todos los cambios que hizo R1 y tener el archivo actualizado en el repositorio local.
3. **Comenzá a revisar el texto.** Para esto te sugerimos abrir la historia del archivo para visualizar los cambios que realizó R1. Esto te permitirá saber si una frase o palabra proviene de la traducción automática o de otro ser humano.



4. **Actualiza tus cambios haciendo *commit*** De manera periódica, o cuando completes la revisión, tendrás que hacer un *commit* con los cambios que generaste. Muchas veces el contenido de los archivos será extenso y necesitarás bastante tiempo para completar la revisión. En estos casos te sugerimos hacer *commits* al menos luego de cada sesión de trabajo indicando en el mensaje hasta que línea del archivo llegaste. De esta manera, si necesitaras delegar la revisión en otra persona, ella podrá revisar la historia del archivo y retomar donde vos dejaste.

5. **Envia tus cambios al repositorio de trabajo haciendo *push*** Hasta ahí la revisión solo vive en tu repositorio local, el paso siguiente es hacer un *push* para mover tu trabajo al repositorio de trabajo online. Es muy importante que el *push* sea a la *branch* asociada al archivo en que estas trabajando, de lo contrario no podremos contectar tu revisión con la traducción original.

6. **Abre un *Pull Request*** Una vez que completaste tu revisión, ¡el capítulo está listo! Sin

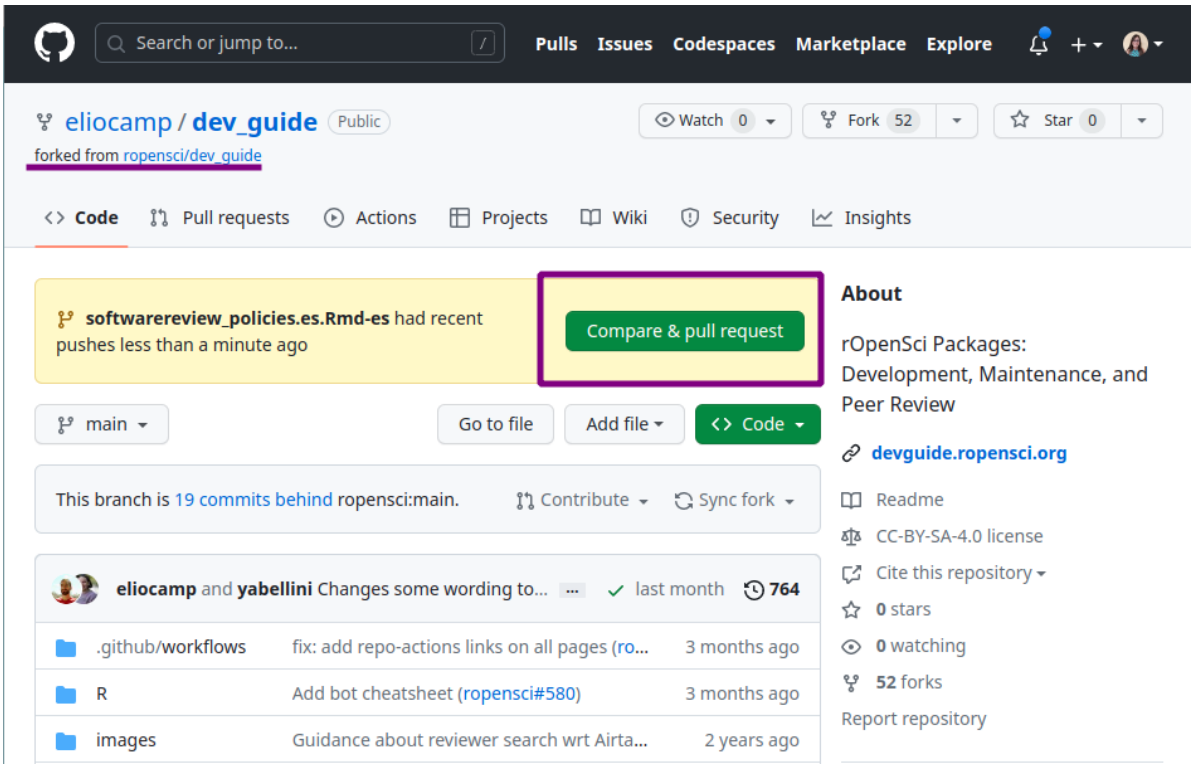
embargo, esta nueva versión solo vive en el repositorio de trabajo. Tendrás que abrir un *pull request* en el repositorio principal para que lo revisé alguien con rol de editor y sea aprobado. En la siguiente sección veremos como abrir el *pull request*.

3.2.5 Pull Request y edición de un capítulo

Una vez que el proceso de revión de un capítulo está completo, es hora de transferir esa nueva versión del repositorio de trabajo al repositorio principal de rOpenSci. Para esto **R2** tendrá que abrir un *pull request* que comparé la versión revisada con la versión que solo tiene traducción automática.

1. Asegurate que todos los cambios están en el repositorio de trabajo y que estás trabajando sobre la *branch* asociada al archivo en revisión, en nuestro ejemplo será `softwarereview_policies.es.Rmd-es`.

2. Si estás en la *branch* correcta, en la interfaz web de GitHub observarás lo siguientes:

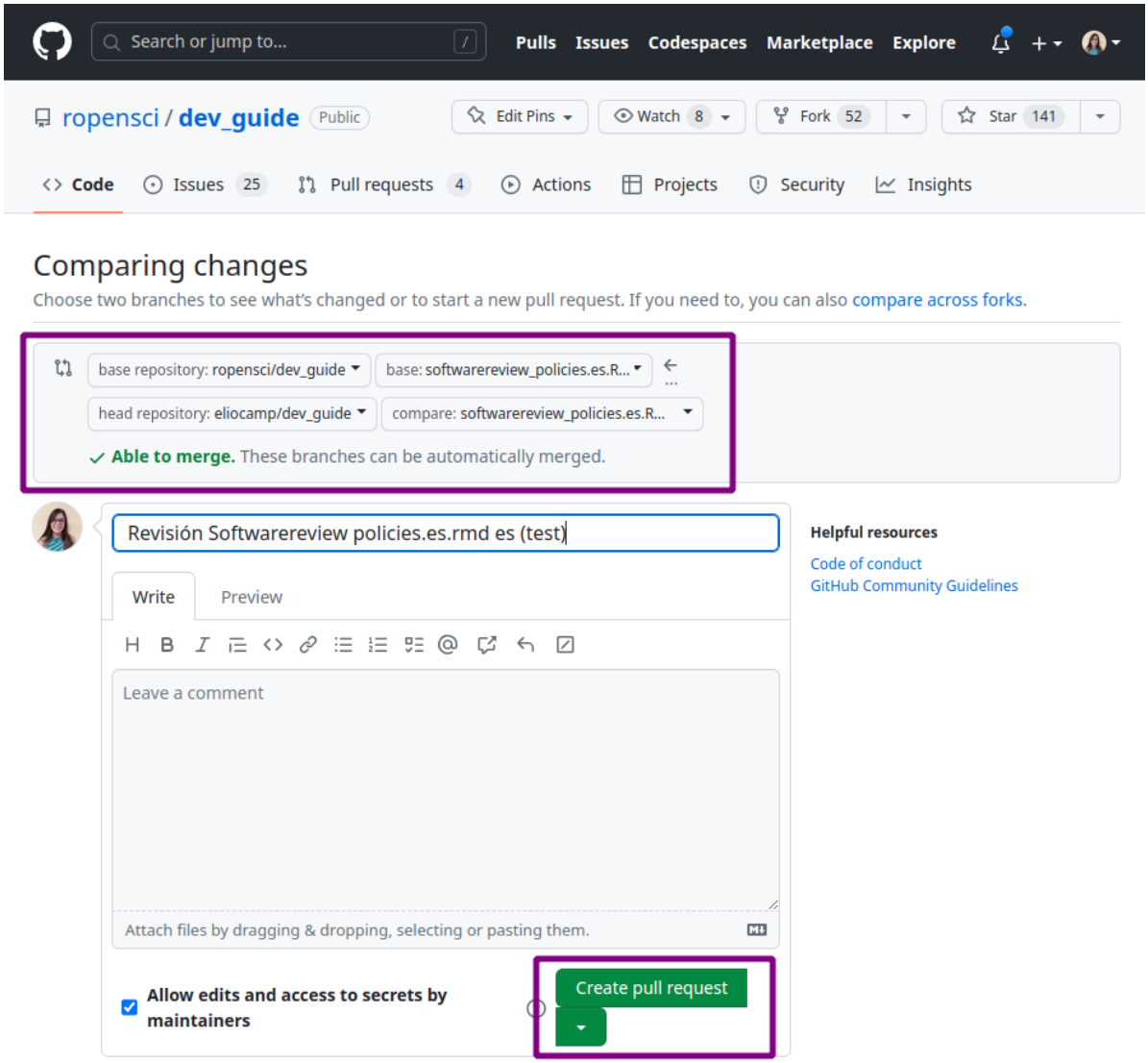


GitHub identifica que hay cambios nuevos (*pushes*) en la *branch* y sugiere compararlos y hacer un *pull request* al repositorio principal.

3. **Haz click en el botón *"Compare & pull request"*.**

4. **Edita el *pull request*** A continuación tendrás la opción de modificar el título del *pull request* y dejar un mensaje. Te sugerimos 2 cosas:

    1. Cambia el título para identificar que se trata de una "Revisión". Es importante para separarlo del *pull request* de la traducción automática.
    2. En el cuerpo del mensaje menciona (usa `@nombre-de-github`) a quien mantiene el proyecto de traducción para que esté al tanto del avance y pueda luego continuar con el proceso.



5. **Chequea nuevamente la *branch*** Antes de crear el *pull request* es **muy importante** que revises que se haga desde la *branch* correcta. Esta es la única manera de conectar

el trabajo de revisión con la traducción automática y poder unificar las versiones correctamente.

6. **Crea el *pull request*** Y listo! Haz click en "*Create pull request".*

A partir de este momento, la revisión queda en manos de **quien mantiene** el proyecto de traducción. Esta persona deberá revisar el *pull request* y hacer consultas a **R1** y **R2** antes de incorporar los *pull requests.*

3.3 Revisión global

El proceso de revisión es un proceso colaborativo en el que, esperamos, participen muchas personas. Esto tiene la gran ventaja de incorporar distintas miradas sobre el uso del lenguaje y generar una traducción que represente a la mayor cantidad de personas. Sin embargo, con toda esta diversidad de miradas y a pesar de la guía específica de cada lenguaje, es posible que las revisiones sean heterogéneas, que a veces se uses criterios distintos o simplemente haya errores de tipeo.

Por todo eso es importante que el proyecto de traducción se complete con una revisión global de todo el material. Sugerimos fuertemente que esta revisión general la realicé una sola persona para mantener un criterio único a lo largo de todo el texto.

Ahora si, **la traducción está lista** y a partir de este punto el equipo de rOpenSci se ocupa para asegurarse que tu trabajo se incluya junto con el material original.

¡Gracias y felicitaciones!

4 Pautas especificas del lenguaje

> **Nota**
>
> Este capítulo es diferente para cada idioma. Contiene las directrices específicas para traducir el contenido bajo los acuerdos de la comunidad de cada lenguaje.
> Para leer las instrucciones de un idioma específico, cambie la versión de la Guía de Traducción a ese idioma utilizando la lista de idiomas a la izquierda (bajo el simbolo).
> Estas orientaciones se aplican cuando se traduce contenido al español.

4.1 Dialecto

La variedad dialectal del español que usaremos en la traducción es la de Latinoamérica. La decisión es consistente con otras traducciones realizadas en la comunidad R de habla hispana. Además, el posible público destinatario que la habla es más amplio. Intentaremos generar una versión lo más neutra posible, por lo que:

- Evitaremos expresiones o usos locales/regionales, es decir, que no están extendidos en toda Latinoamérica.
- No utilizaremos el voseo (*vos/vosotros*). La guia *rOpenSci Packages: Development, Maintenance, and Peer Review* está dirigido a una segunda persona, así que para mantener lo mas posible la neutralidad la traduciremos como *tú* > Ejemplo: … *We hope that you'll find the guide* > … *esperamos que encuentres esta guía*; *You can view updates* > … *Puedes ver actualizaciones*).

4.2 Género gramatical.

El español tiene género gramatical (masculino, femenino y muy pocos neutros). Estas son las convenciones para manejar el género gramatical cuando aparezcan en el texto. Las situaciones que no estén contempladas en la guía las discutimos en comunidad (en los *issues* o el canal de Slack correspondiente) para tomar una decisión y las agregamos a esta sección.

- En principio, intentaremos ajustar la redacción para evitar tener que asignar un género: por ejemplo, en esta frase: "*... We are thankful for all authors, reviewers and guest editors*". se podría traducir como: *Agradecemos a todas las personas que han sido autoras, revisoras y editoras invitadas.*
- Si no podemos evitar usar marca de género, lo más aceptado por el momento es el desdoblamiento, femenino-masculino o masculino-femenino, que puede ser de dos maneras, por ejemplo: *las/los autoras/es, los/las revisores/as*; o bien *los y las autores y autoras*, etc.
  En esta traducción, al desdoblar:

  a. Vamos a utilizar *los/las* ó *las/los* privilegiando la agilidad y fluidez del texto, que el mismo se entienda y que sea claro el mensaje

  b. Para que haya coherencia a lo largo del texto y mostrar que no hay una determinada jerarquía alternaremos el uso del femenino masculino primero, entre capítulos y el uso será consistente durante todo el capítulo.

  c. El uso de las barras puede entorpecer la lectura, aquí algunas opciones a tener en cuenta que pueden darle más fluidez al texto y respetar el lenguaje no sexista:

Opción	Alternativa
del/de la revisora	del equipo revisor
al/a la editora	al conjunto de editores y editoras, al equipo editorial

  d. La lectura del lenguaje no sexista con las conjunciones "al" y "del" es complicada.

- Caso de capítulo donde se debe usar *los/las*: dejar el masculino en la conjunción, pero no en el sustantivo si este tiene marca de género: *"Al autor/a"*.
- Caso de capítulo donde se debe usar *las/los*: dejar el femenino en el artículo, pero no en el sustantivo si este tiene marca de género: *"A la desarrolladora/or"*.

4.3 Verbos

En español, como los verbos tienen marca de persona, género y número, tenemos la flexibilidad de poder omitir el sujeto, ya que por contexto se suele entender a qué nos estamos refiriendo. Esto nos permite evitar la repetición de palabras.

Los modos y tiempos verbales tambien pueden ser diferentes al del idioma original. Al traducir, por lo tanto, se debe priorizar la forma verbal que sea mejor para expresar el sentido del fragmento en español, no la que parezca ser literal del inglés.

En todos los casos, hay que usar la opción que suena más natural en español y que queda más claro para quien lee.

4.4 Regularidades

Hay regularidades que no siempre se cumplen. Por ejemplo, en inglés los adjetivos se anteponen a los sustantivos, ej: *simple model*, *correct answer*, etc., mientras que en español suele ser al revés: ponemos los adjetivos después del sustantivo: *modelo simple*, *respuesta correcta*, etc. Sin embargo, hay casos en que en español la forma *"no marcada"*, es decir, la que nos suena más natural, es con el adjetivo al principio:

Ejemplo: > …There's a better way… > …Hay una mejor manera…

En general, ante dudas de este tipo, pensar en qué es lo que suena más natural/normal en español. Siempre se puede pedir opinion en los *issues* o en el canal de Slack.

4.5 Expresiones idiomáticas

Las expresiones idiomáticas no son traducibles de manera literal. En caso de que las hubiere, hay que proponer una traducción que permita entender el sentido de expresión original. A veces existen expresiones de similar significado.

Ejemplo: > …Birds of a feather, drawn together… > …Ellos se crian y el viento los amontona…

4.6 Traducción (o no) de términos técnicos

Hay términos técnicos que será necesario traducir y otros que no. Para decidir tomemos en cuenta si existe una versión en español extendida o si se suele utilizar la versión original en inglés.

Como fuentes de consulta se pueden utilizar Wikipedia, Diferencias de vocabulario estándar entre países hispanohablantes, GLosario y el diccionario bilingüe de Enseñar Tecnología en Comunidad

Los términos técnicos que se mantienen deben ir en un formato especial, en nuestro caso cursiva. De ser pertinente, se debe ofrecer una posible traducción al español, ya que en algunos casos permite entender mejor el concepto que está detrás y determinar qué género gramatical asignarle.

La primera aparición en el texto de los términos técnicos que se traducen puede acompañarse con la palabra en el idioma original, por ejemplo inglés, entre paréntesis para ayudar a entender el contexto y el significado. Por ejemplo:

… el conjunto de datos (*dataset*) …

En ls [sección @glosario-es] se cuenta con el listado actualizado tanto de términos técnicos que se mantienen como los que se traducen y como se deben traducir segun el contexto.

Además, tengamos en cuenta como escribir estos términos del Latín:

Latin	Español
i.e. (id est)	"es decir"
e.g. (exempli gratia)	por ejemplo, abreviatura "p. ej."

4.7 Material referenciado en la guía

Cuando se encuentre una referencia a la denominación de material mencionado en el libro, como títulos de libros, nombres de instituciones, sitios web, guías, etc. Se deja el título o nombre original en Inglés con la letra en cursiva y se aclara entre paréntesis una traducción del título o nombre en español. Si existe una version en español del material entonces se utilizará esa versión en la aclaración.

4.8 Código

No se traducen los nombres de paquetes, funciones y sus parámetros correspondientes al lenguaje R. Si se puede, se debe ofrecer una traducción de estos elementos la primera vez que aparecen. Por ejemplo,

> usaremos la función *group_by()*, que significa "agrupar por", para …

> Usa *message()* — del inglés *mensaje* — y *warning()* — del inglés *advertencia* — para comunicarte con quien use tus funciones.

Tampoco se traducen los caminos relativos a archivos.

Se debe traducir:

1. nombres de variables;
2. nombres de constantes;
3. comentarios;
4. funciones y sus parámetros, *solo* cuando son ejemplos de funciones generadas por la persona que desarrolla: si un fragmento de código crea una función como ejemplo, podemos poner su nombre y el de sus parámetros en español.

ejemplo:

```
# Código original:

   inside <- function(point, lower, higher) {
        if (point <= lower) {
            return false
        } else if (point >= higher) {
            return false
        } else {
            return true
        }
   }

# Traducción al Español:

  adentro <- function(punto, mas_bajo, mas_alto) {
        if (punto <= mas_bajo) {
            return false
        } else if (punto >= mas_alto) {
            return false
        } else {
            return true
        }
   }
```

4.9 Diagramas

1. Las figuras y los diagramas se traducen.
2. Las capturas de pantalla sólo se traducen si la interfaz de usuario está en castellano.

4.10 Datos

Si existe una versión traducida de los datos, por ejemplo con los nombres de las variables en castellano, se utilizan esos datos.

Si no la hay, se utilizan los datos originales, mencionando lo que significan los nombres de las variables.

Por último, se puede iniciar un proyecto para traducir los datos o seleccionar datos en castellano significativos para el público destinatario que puedan sustituir a los datos originales. Se puede ver el proyecto dados como un ejemplo de traduccion de datos del inglés al portugués.

4.11 Aspectos de ortografía y gramática

- Ni los demostrativos ni el adverbio "solo" se tildan.
- Días y meses se escriben con minúscula en español.
- Los títulos llevan mayúscula solo en la palabra inicial (salvo que incluyan un nombre propio).
- Para consultar la forma convencional de una abreviatura en español, revisar este listado de la real academia española.

Palabra	Abreviatura
por ejemplo	p. ej.

4.12 Aspectos de puntuación del español[1]

No usar *coma* antes del *y*, excepto que:

a) sea necesario para darle claridad al texto;

b) sea necesario luego de una enumeración anterior, según este ejemplo:

- "Preparé el examen y tomé un café":_ va sin coma
- "Preparé el examen y la clase de la semana siguiente, y tomé un café":_ va con coma para distinguir del "y" de la primera parte de la oración.

Punto y paréntesis:

- Si hay una oración completa dentro del paréntesis, el punto va adentro. (Por ejemplo si acá sigue un comentario extra sobre algo que decidió poner en una oración aparte, creo que lo hace con acotaciones.)
- Si el paréntesis es una aclaración dentro de la oración, el punto va afuera (este caso).

[1] Diccionario panhispánico de dudas (DPD) [en línea], https://www.rae.es/dpd/coma, 2.ª edición (versión provisional). [Consulta: 31/10/2024].Real Academia Española y Asociación de Academias de la Lengua Española.

4.13 Glosario

Este glosario se desarrolla en el marco de las traducciones del material de rOpenSci. Sirve de apoyo a la tarea de determinar qué términos técnicos se traducen y cuales no. También es la fuente que determina que palabras preferimos utilizar antes la posibilidad de contar con más de una opción.

Como fuentes de consulta para generar nuestro glosario y mantenerlo actualizado se pueden utilizar:

- Wikipedia,
- Diferencias de vocabulario estándar entre países hispanohablantes,
- Glosario,
- Diccionario bilingüe de Enseñar Tecnología en Comunidad

english	spanish	synonym	observation
aesthetics	estéticas	NA	NA
assignment operator	operador de asignación	NA	NA
atomic vectors	vectores atómicos	NA	NA
back slash	barra invertida	NA	NA
backtick	acento grave	NA	NA
base R	R base	NA	NA
by default	por defecto	NA	NA
click	hacer clic	NA	NA
console	consola	NA	NA
debugging	depurar	NA	NA
dataset	conjunto de datos	set de datos	NA
facet	faceta	NA	NA
input	input	NA	la palabra existe como anglicismo en español.No se traduce
intercept	ordenada al origen	NA	NA
key	clave	NA	NA
legend	leyenda	NA	NA
list-columns	columnas-lista	NA	NA
log-transformation	transformación logarítmica	NA	NA
mapping	mapear	NA	NA
match	coincidencia	NA	NA

english	spanish	synonym	observation
missing values	valores faltantes	NA	NA
modelling	modelado	NA	NA
named list	lista nombrada	NA	NA
partial matching	coincidencia parcial	NA	NA
parse	segmentar	analizar	depende del contexto
placeholder	marcador de posición	NA	NA
query	consulta	NA	NA
raw data	datos sin procesar	datos crudos	NA
regular expression	expresión regular	NA	NA
test set	datos de validación	set de validación	NA
training set	datos de entrenamiento	set de entrenamiento	NA
tidy data	datos ordenados	NA	NA
feedback	devolución	retroalimentación	NA
call stack	pila de llamada	NA	NA
chunking	particionar	fragmentar	NA
loop	bucle	NA	NA
path	ruta de acceso	NA	NA
stack	pila	NA	NA
stack frame	marco de pila	NA	NA
best practices	buenas prácticas	NA	NA
test	test	NA	en el contexto de test unitarios
issue	issue	NA	en el contexto de git
push	push	NA	en el contexto de git
unit tests	tests unitarios	NA	NA
badge	etiqueta	NA	NA
deprecated	obsoleto	NA	NA
defunct	caduco	NA	NA
branch	rama	NA	en el contexto de git
cheatsheet	guía rápida	NA	NA
cheat sheet	guía rápida	NA	NA
r universe	r universe	NA	NA
runiverse	runiverse	NA	NA
r-universe	r-universe	NA	NA

4.14 Fuentes

Estas directrices de traducción se basan en la experiencia y los documentos generados durante la traducción de Enseñar Tecnología en Comunidad. También consultamos y tuvimos en cuenta la guía para traductores de Programming Historian, el proyecto de traducción colaborativa de "R para Ciencia de Datos" y el Glosario y lineamientos para las traducciones al español de The Carpentries.

5 NEWS