

# Proyecto Cursos: Es hora de aprender GNU/Linux!

Nicolás César  
Lucas Di Pentima  
Lunix

Copyright (C) 2002, Nicolás César y Lucas Di Pentima. Este artículo puede ser copiado y distribuido en las condiciones de la licencia GNU para documentación libre, FDL (<http://www.gnu.org/licenses/fdl.txt>), sin secciones invariantes.

Proyecto Cursos es una iniciativa que surgió en el año 2000 para la confección de material de estudio (para los alumnos) para el dictado de clases de GNU/Linux. Es parte del Proyecto TLDP-ES (<http://es.tldp.org>) y cuenta con voluntarios de todo el mundo. La reciente migración a un sistema modular de porciones de texto le da características únicas comparado con la confección tradicional de documentación.

## Tabla de contenidos

1. Historia	1
1.1. ¿Como empezó todo?	1
2. Motivación	2
2.1. ¿Que no es Proyecto Cursos?	3
2.2. ¿Que es Proyecto Cursos?	3
3. Arquitectura	4
3.1. Recompaginación manual y tediosa	4
3.2. Redundancia innecesaria de temas	4
3.3. Trabajo desperdiciado	4
4. Documentación por componentes	5
5. Problemas a resolver	5
6. Herramientas de trabajo	6
7. Roles dentro del proyecto	7

## 1. Historia

### 1.1. ¿Como empezó todo?

Comenzó en el mes de agosto del 2000 como la necesidad de tener una guía post-instalación para aquellas personas a las cuales les instalábamos GNU/Linux por primera vez. Posteriormente vino una oferta de nuestra universidad para dictar cursos de GNU/Linux en la facultad regional Santa Fe. En poco tiempo pasó de ser una guía resumida a un material de apoyo para el alumno de mayor extensión.

Para esa altura nosotros no teníamos mucha idea de los beneficios de crear documentación libre hasta que un discurso de Richard Stallman y unas palabras alentadoras de César Ballardini nos cambiaron la forma de pensar.

Para el mes de setiembre pusimos a disposición de la comunidad los 3 o 4 capítulos que teníamos ya escritos publicándolos en el sitio del proyecto LuCAS.

Fue enorme la sorpresa cuando en pocas semanas y después de haberse publicado un aviso del inicio del proyecto en Barrapunto el 12 de agosto, se acercaron a darnos una mano gente de todo el mundo. Ellos tenían las mismas necesidades que nosotros y a la vez todos colaborábamos para lograr los objetivos propuestos en grupo.

- 26 de setiembre: comienza curso con el temario finalizado en un 60%, aún no se habían recibido colaboraciones en la redacción pero sí en la revisión.
- Una vez establecida la base de colaboradores, el avance del proyecto tomó velocidad. Aunque los inscriptos a la lista eran cerca de 30 a fines del año pasado, el equipo de colaboraciones eran sólo unos pocos, pero se contaba con la ayuda de docentes, especialistas en el idioma español y una buena cantidad de voluntarios desinteresados dispuestos a darnos una mano.

## 2. Motivación

- El software libre debe tener docentes.
  - Se le debe prestar atención a la generación que viene. Los actuales usuarios de sistemas propietarios tienen mayor resistencia a la migración a GNU/Linux por diversas razones, pero los potenciales usuarios de la informática están más predispuestos al aprendizaje, es entonces éste el grupo de personas que debemos tener en cuenta para inculcar el uso del software libre y su filosofía en la sociedad. Es nuestra responsabilidad gestar catalizadores de este nuevo cambio.

- Empresas de software propietario están «regalando» sus productos a las universidades para que éstas «se nivelen tecnológicamente con el mundo». Esto sucede aparentemente sin la conciencia de parte de las instituciones educativas, que lo que se crea es una dependencia tecnológica con dichas empresas. Viendolo desde un punto de vista económico, una cantidad significativa de inversiones son llevadas a cabo por dichas instituciones solamente para financiarles las campañas publicitarias a estas empresas, mientras que si esa misma cantidad de recursos se invirtiera en software y documentación libres, es la misma comunidad la que se ve beneficiada y así todos ganan.
- El software libre debe tener documentación libre. El único esquema posible de documentación para docentes, es que la documentación sea libre.
  - Nuestra experiencia nos llevó a evaluar inevitablemente la relación costo/beneficio entre las horas que toma el desarrollo de material para el dictado de cursos, y el tiempo total del curso dictado con este material. Nos dimos cuenta que el tiempo que invertíamos en generar la documentación era demasiado. La única forma que vimos de reducir este tiempo es la colaboración entre docentes para gestar el material, dividiendo así esfuerzos entre todos y beneficiándonos mutuamente.
  - Los docentes en general no son informáticos, y los informáticos en general no son docentes. Proyecto Cursos pretende facilitar a docentes la generación de material para sus cursos y proveer de herramientas pedagógicas a los informáticos/docentes que las necesiten. La documentación libre agiliza el flujo de colaboración entre informáticos y docentes.
  - Se necesita llegar a un ritmo de desarrollo de documentación para enseñanza equivalente al que tiene el software libre hoy en día, para llevar el espíritu de cooperación de los desarrolladores a los usuarios. Si el software es libre pero la capacitación depende de herramientas propietarias, establecemos dependencias con las mismas características que la dependencia del software propietario. Cuando nos referimos a herramientas propietarias, hablamos de libros con derechos de autor que restringen la copia y/o modificación. Si el docente depende de estas herramientas para la enseñanza, sus alumnos se verán limitados en la obtención del material de estudio por razones económicas o morales. Si apuntamos a la difusión de GNU/Linux, debemos eliminar esta limitación.
  - Sólo la documentación libre hará al software totalmente libre.

## 2.1. ¿Que no es Proyecto Cursos?

- No es un HOWTO.
- No es una guía.
- No es un texto de referencia.

## 2.2. ¿Que es Proyecto Cursos?

Es la creación de recursos lingüísticos para la generación de material bibliográfico para el dictado de cursos.

Las características especiales que requiere el dictado de un curso se ven reflejadas en el material de nuestro proyecto, ejemplos de esto serían:

- Ejercitación práctica
- Exámenes propuestos
- Sugerencias para dinamizar el dictado de ciertos temas
- Recursos pedagógicos: ejemplos, paralelismos, etc.

## 3. Arquitectura

El estilo «clásico» de los manuales no es el adecuado para el dictado de un curso, donde el docente quiere decidir el temario y proponer un rumbo en su dictado. Esta clase de documentación no provee la flexibilidad necesaria, porque aunque se dispone en muchos casos de su «código fuente», el trabajo de compaginación es bastante y poco reutilizable.

Ejemplo: como docente, yo poseo un manual de sendmail el cual en su primer capítulo habla de reglas m4. Ese capítulo es excelente si quiero dictar una clase de sendmail exclusivamente pero muchas veces sendmail es un tema mas del curso. De no querer incluir este capítulo deberé recompagnar la documentación existente, malgastando el tiempo en una tarea tediosa y no rentable.

Proyecto Cursos padeció este problema habiendo desarrollado dos cuadernillos (básico e intermedio), surgió un curso que requería capítulos de ambos; esto nos hizo dar cuenta de lo siguiente:

### 3.1. Recompaginación manual y tediosa

La recompaginación no era parte del proyecto, sino que era una actividad paralela de los docentes que la necesitaban. La generación de infinta cantidad de cuadernillos con las distintas combinaciones de temas para abarcar todos los posibles cursos, es algo obviamente imposible de hacer.

### 3.2. Redundancia innecesaria de temas

Cuando un mismo tema era incluido en varios cursos, pasaban a haber dos copias (o mas) de un mismo texto, haciendo el mantenimiento de éstos imposible, y quedando estos nuevos cuadernillos obsoletos para posteriores cursos por el

simple hecho que no eran actualizados. Esto provocaba mas trabajo extra aún, ya que debíamos realizar el mismo trabajo de recompaginación a la hora de generar el material para una nueva versión.

### **3.3. Trabajo desperdiciado**

El duro trabajo de refinamiento que se realiza a los cursos derivados se perdía. La acción de actualizar y corregir el material en estos cuadernillos provocaba un desperdicio de esfuerzo ya que no se veían reflejados los cambios en las versiones principales en el proyecto, dicho de otra manera: el trabajo de redacción que no se hacía en repositorio el proyecto, no se podía reutilizar.

Ejemplo: Debemos dar cuatro cursos, uno destinado a programadores, otro a administradores de sistemas, otro a usuarios de escritorio y otro oficinistas. Los programadores y los administradores tienen varios temas en común como ser los comandos básicos y la programación en shell. Al mismo tiempo los usuarios de escritorio y los oficinistas necesitan aprender el uso de un paquete ofimático (a diferentes niveles). Por otro lado, un programador necesita conocer el funcionamiento de un entorno de escritorio al igual que un oficinista. Preparamos los cuatro cuadernillos duplicando los contenidos en común y damos los cursos, finalmente ponemos en línea el material generado y en ese momento finaliza el «ciclo de vida» de dicho material. A la hora de dar un curso similar, deberemos recompagnar el material, malgastando tiempo y esfuerzo ya que el cuadernillo anterior está desactualizado y por ejemplo el KDE versión 1.x ya fue reemplazado por la versión 2.x o 3.x. Por otro lado, al dictar el curso nos damos cuenta de un error en la documentación, para corregirlo tenemos dos opciones: corregimos nuestra copia del cuadernillo o todas las copias, presentándose en ambos casos obvios problemas.

## **4. Documentación por componentes**

Los laboratorios de investigación y desarrollo del Proyecto Cursos elaboraron una solución a estos problemas: las porciones. Las porciones consisten en la división de todos los temas en fracciones de texto fácilmente reubicables, permitiendo así la composición de manera rápida de nuevos cuadernillos.

Además se logró la separación del contenido de la estructuración propia de cada curso. El contenido (porciones) es entonces combinado en «plantillas» que definen el orden y nivel de importancia de cada porción, además de establecer el estilo del documento, márgenes, etc.

Ejemplo: Un temario define las porciones que se utilizarán en ese curso, esto pasa a ser una plantilla. Si cualquiera de las porciones es actualizada habrá que recompilar la plantilla y un nuevo documento actualizado se obtiene en minutos. Además, si en nuestro curso un tema específico como por ejemplo Apache toma relevancia, fácilmente se puede ascender a la categoría de capítulo si anteriormente estaba como una simple sección.

Otra ventaja de este modo de trabajo es que el intercambio y expansión de temarios entre docentes se realiza de manera simple permitiendo la personalización de cada curso.

Ejemplo: en la forma original de trabajo, sin entrar en la recompaginación del temario, se debían incluir todos los temas de los manuales, pudiendo no ajustarse a las necesidades del grupo al cual se le dictaba la clase. Hoy con las porciones en poco tiempo se puede generar un temario personalizado.

## 5. Problemas a resolver

El sistema de documentación por componentes acarrea sus propios problemas que debemos enfrentar y solucionar:

1. Referencias externas a una porción: Cuando dentro de una porción referenciamos un tema que pertenece a otras porciones, no sabemos si dichas porciones referenciadas serán parte del cuadernillo resultante.
2. No se puede suponer la existencia anterior o posterior de una porción, teniendo en lo posible que generar porciones auto-contenidas.
3. No se debería hacer referencia a la estructura del documento, ya que una porción no posee estructura definida. Por ejemplo, mencionar la palabra "capítulo".
4. No hay una correlación directa entre un cuadernillo y las porciones que contiene. Esto dificulta la corrección de la documentación impresa.
5. Al existir una gran cantidad de porciones, se hace difícil la ubicación de un determinado contenido en el repositorio de porciones. En la actualidad existen mas de 200 porciones (y no se ha terminado la migración de todos el material aún).

Las soluciones que vemos viables:

- Solución a las referencias internas a un documento: se realiza en dos etapas, en la primera se asigna un identificador único a cada referencia en las porciones y se registran las porciones incluídas. En la segunda etapa con ese identificador se comprueba si la porción existe en el documento generado; en caso afirmativo se asigna el valor correspondiente (número de página, capítulo, sección, etc.) y en caso contrario el indentificador se debe reemplazar por una dirección electrónica que contenga la porción. Para esto necesita confeccionar una tabla de referencias como parte del proyecto.
- Correlación entre cuadernillo y sus porciones: mediante alguna opción poder compilar un cuadernillo de depuración el cual tenga referencias a las porciones que contiene y detalles como números de línea, etc.
- Para asegurarnos de la existencia de porciones se debería implementar un sistema de dependencias entre porciones.

## 6. Herramientas de trabajo

Luego de vagabundear con nuestro proyecto fuimos cordialmente invitados por los coordinadores de LuCAS, a ser parte de este gran equipo.

**CVS** Porque lubrica los mecanismos colaborativos de trabajo en grupo a través de Internet.

**Bugzilla** Para centralizar la asignación de tareas, y que no se pierdan entre los mensajes de la lista de correo.

**Lista de correo** El «centro de operaciones» del Proyecto Cursos que se utiliza para coordinar esfuerzos como en cualquier otro proyecto.

**Web (Sistema Curso-o-matic)** Es una herramienta para la creación automática de temarios, permitiendo elegir el orden de los temas y su nivel de importancia en el curso a dictar. Su dirección es:

<http://www.lunix.com.ar/cursos-temario.php>

**LaTeX/DocBook** Comenzamos a escribir el proyecto en LaTeX debido a que es una poderosa herramienta libre y lo teníamos a César Ballardini cerca. LaTeX tiene alta calidad en la presentación, no obstante es un lenguaje con poca estructuración comparado con DocBook. DocBook presenta ventajas múltiples: es específico para documentación técnica, es un derivado de XML que presenta un horizonte tentador en la cantidad de aplicaciones que se están creando para su manipulación. Es la herramienta oficial de TLDP-ES y será soportada por la imprenta-e.

## 7. Roles dentro del proyecto

Aunque existan distintos tipos de trabajo dentro del proyecto, no existe una jerarquía en todo el esquema. El rol de coordinador simplemente sirve para organizar un poco las tareas a realizarse, y canalizar esfuerzos. El contenido de los manuales se decide en conjunto y cualquiera dispuesto a ayudar tiene todo el derecho de opinar, tratando de llegar a un consenso entre todos.

Roles:

**Redactor** El encargado de generar los contenidos. Redactor puede ser de un capítulo, de una sección o de un pequeño párrafo dentro de un conjunto de puntos. Si alguien nota que falta algo importante en un texto ya redactado, previa discusión en la lista, podrá agregarlo tranquilamente. La tarea de

redacción también incluye la adaptación de textos ya escritos (con el correspondiente permiso) a las necesidades del proyecto.

**Revisor** Aunque parezca de menos importancia, tiene la misma importancia y responsabilidad que un redactor. Un redactor puede cometer errores graves sin darse cuenta y la tarea de revisar tanto la ortografía como la gramática y los contenidos no es para menospreciar.

**Migrador** Actualmente aún no hemos finalizado la migración a porciones, si bien hay más de 200 porciones suponemos muchas más cuando la totalidad del material sea procesada. Además está programada la migración de formato, de LaTeX a DocBook, por lo que necesitaremos gran cantidad de voluntarios para realizar esta tarea lo más rápido posible.