

MP0491

Sistemas de gestión empresarial
UF5 Desarrollo de componentes

**5.3. Entornos de desarrollo
y herramientas de desarrollo
en sistemas ERP y CRM**

Índice

☰	Objetivos	3
☰	Necesidad de modificar el ERP/CRM	4
☰	IDEs generalistas	9
☰	Características de Eclipse	12
☰	Instalar Eclipse	14
☰	Trabajar con Odoo en Eclipse	19
☰	Organizar el trabajo	24
☰	Resumen	29

Objetivos

Con esta lección perseguimos los siguientes objetivos:

- 1 Conocer las herramientas de que disponemos en los sistemas ERP y CRM.
 - 2 Saber cómo podemos trabajar con ellas.
 - 3 Conocer Eclipse y sus posibilidades.
 - 4 Comprender qué es el control de versiones.
-

¡Ánimo y adelante!

Necesidad de modificar el ERP/CRM

La primera pregunta que debemos hacernos es: ¿Para qué necesitamos un entorno de desarrollo?

Ya sabemos que instalar un ERP/CRM no va a ser la solución total. Aunque la decisión haya sido muy meditada, no vamos a conseguir nada que nos resuelva totalmente los problemas informáticos de la empresa.

Incluso si hubiéramos sido tan buenos, o hubiéramos tenido tanta suerte de haber encontrado la solución ideal, esa solución no será nunca definitiva. **Dentro de una semana, un mes o un año, necesitaremos modificar o ampliar nuestro ERP/CRM.**

La necesidad de modificación puede ocurrir en distintos niveles

- Puede que necesitemos añadir código, para realizar alguna acción que en principio no estaba prevista.
- Puede que necesitemos añadir alguna columna.
- Puede que necesitemos modificar alguna vista.

Para empezar, vamos a centrarnos en las soluciones disponibles para resolver el primer caso, es decir, cuando necesitamos **añadir código**.

Como siempre, primero tendremos que comprobar lo que nos ofrece el paquete. Luego, si la solución del paquete no es buena, nos podremos inclinar por las **soluciones estándar disponibles en el mundo de la programación, como las soluciones RAD o los IDES**.

Soluciones RAD

Habrá ocasiones en que la adaptación del código pasará por escribir nuevos componentes:

- Para hacerlos trabajar como funciones llamadas por nuevos formularios.
- Para hacerlos funcionar como nuevos módulos.
- O, simplemente, para añadirlos como *plugins* en nuestra instalación.

Todos estos casos que requieren programación y acostumbran a ser laboriosos, nos invitan a **buscar la herramienta de desarrollo adecuada**.

(i) Ten en cuenta que, aunque todos los lenguajes de programación nos permiten que escribamos nuestras fuentes utilizando un sencillo block de notas, en realidad esta es la peor forma de hacerlo.

Las razones son varias:

- Debemos acordarnos de todas las instrucciones, porque no se va a autocompletar nada.
- Si nos equivocamos, nadie nos va a avisar de ello.
- Nos va a costar repasar el programa escrito porque todo el texto aparece del mismo color, etc.

Para solucionar este problema, existen multitud de **herramientas RAD (Rapid Application Development)**. La idea de utilizar estas soluciones puede que nos venga un poco grande, pero si miramos lo que contienen, veremos que gran parte de lo que necesitamos nos lo proporciona el paquete y viene ya resuelto.

Veamos, a continuación, las **distintas fases en la aplicación de soluciones RAD**.

Modelado de gestión

El flujo de información entre las funciones de gestión se modela de forma que responda a las siguientes preguntas:

- ¿Qué información conduce el proceso de gestión?
- ¿Qué información se genera? ¿Quién la genera?
- ¿A dónde va la información?
- ¿Quién la procesó?

Modelado de datos

El flujo de información, definido como parte de la fase de modelado de gestión, se refina como un conjunto de objetos de datos necesarios para apoyar la empresa.

Se definen las características de cada uno de los objetos y las relaciones entre estos objetos.

Modelado de proceso

Los objetos de datos, definidos en la fase de modelado de datos, quedan transformados para lograr el flujo de información necesario para implementar una función de gestión.

Las descripciones del proceso se crean para añadir, modificar, suprimir o recuperar un objeto de datos.

Generación de aplicaciones

El RAD asume la utilización de técnicas de cuarta generación. En todos los casos se utilizan herramientas automáticas para facilitar la construcción del software.

Pruebas de entrega

Como el proceso RAD enfatiza la reutilización, ya se han comprobado muchos de los componentes de los programas. Esto reduce el tiempo de pruebas.

Sin embargo, se deben probar todos los componentes nuevos y se deben ejercitar todas las interfaces a fondo.

Los sistemas RAD proporcionan herramientas para la total o parcial mecanización de todas estas fases, simplificando su realización y controlando el avance. Normalmente, incorporan utilidades para facilitar el trabajo en grupo, permitiendo la compartición de todo el material y supervisando el uso que se hace del mismo.

En función de la modificación a la que nos tengamos que enfrentar, necesitaremos emplear más o menos fases. En estos entornos, nos centramos habitualmente en las dos últimas.

IDES

Para ayudarnos en el desarrollo y simplificarnos el trabajo de programar, tenemos otro acrónimo: IDE (*Integrated Development Environment*).

i Un IDE es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.

Normalmente, un IDE consta de un editor de código fuente, herramientas de construcción automáticas y un depurador. La mayoría de los IDE tienen función de autocompletado inteligente de código (*IntelliSense*). Y algunos contienen, además, un compilador, un intérprete o ambos, como, por ejemplo, NetBeans y Eclipse.

El límite entre un IDE y otras partes del entorno de desarrollo de software más amplio no está bien definido. Muchas veces, a efectos de simplificar la construcción de la interfaz gráfica de usuario (GUI, por sus siglas en inglés) se integra un sistema controlador de versión y varias herramientas.

Muchos IDE modernos también cuentan con un **navegador de clases**, un **buscador de objetos** y un **diagrama de jerarquía de clases**, para su uso con el desarrollo de software orientado a objetos.

Vamos a definir lo que debe hacer un IDE, es decir, sus características esenciales, para que pueda ser una ayuda en nuestro trabajo:

- Multiplataforma.
- Soporte para diversos lenguajes de programación.
- Integración con sistemas de control de versiones.
- Reconocimiento de sintaxis.
- Extensiones y componentes para el IDE.
- Integración con *frameworks* populares.
- Compilación automática.
- Depurador.
- Soporte FTP para desarrollo/actualización remota.
- Importar y exportar proyectos.
- Sistemas de generación de paquetes de producción.
- Múltiples idiomas.
- Soporte para la generación de Manual de Usuarios y Ayuda.

Pero nosotros estamos trabajando en un sistema más cerrado, con limitaciones y condiciones, por lo que **deberíamos añadir a esta lista**:

- Reconocimiento de nuestro paquete.
- Reconocimiento del lenguaje del paquete.
- Interacción con las APIs.
- Entorno de pruebas integrado.
- Liberación a producción controlada.

Esta segunda parte solo la conseguiremos con aquellas herramientas desarrolladas para interactuar con el paquete, y a veces se quedarán cortas en requerimientos generales.

Si la herramienta está muy orientada al paquete, **nos ofrecerá soluciones parciales y solo nos permitirá resolver los problemas más frecuentes**, aunque eso lo hará con mucha facilidad.

IDEs generalistas

Recuerda que nuestro objetivo a conseguir es **simplificar la programación**.

Para ello, si tenemos pensado realizar modificaciones después de instalar un ERP/CRM, podríamos incorporar alguna herramienta de las que existen en el mercado.

Algunos de estos entornos pensados para programación convencional son:

- **Eclipse:** software libre. Es uno de los entornos más utilizados a nivel profesional. El paquete básico de Eclipse se puede expandir mediante la instalación de *plugins* para añadir funcionalidades a medida que se vayan necesitando.
Lenguajes: Java, ANSI C, C++, JSP, PHP, Javascript...

- **NetBeans:** software libre. Otro entorno muy utilizado, también expandible mediante *plugins*. Facilita bastante el diseño gráfico asociado a aplicaciones Java.
Lenguajes: HASKELL, C++, YACC, SH, LEX, PERL, ANSIC...

- **BlueJ:** software libre. Es un entorno de desarrollo dirigido al aprendizaje de Java (entorno académico) y sin uso a nivel profesional. Destaca por su sencillez e incluye algunas funcionalidades dirigidas a que las personas que estén aprendiendo tengan mayor facilidad para comprender aspectos clave de la programación orientada a objetos.
- **JBuilder:** software comercial. Se pueden obtener versiones de prueba o versiones simplificadas gratuitas en la web, buscando en la sección de *productos y desarrollo de aplicaciones*. Permite desarrollos gráficos.
- **JCreator:** software comercial. Se pueden obtener versiones de prueba o versiones simplificadas gratuitas en la web. Este IDE está escrito en C++ y omite herramientas para desarrollos gráficos, lo que lo hace más rápido y eficiente que otros IDEs.
Lenguaje: Java.
- **MsVisual Studio:** es un entorno de desarrollo integrado para sistemas operativos Windows y Mac, que permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web, en cualquier entorno que soporte la plataforma .NET
Lenguajes: C++, C#, Visual Basic .NET, F#, Java, Python, Ruby y PHP.
- **Visual C++:** también conocido como MSVC++, Microsoft Visual C++, es un entorno de desarrollo integrado para lenguajes de programación en el entorno Windows.
Lenguajes: C, C++ y C++/CLI.
- **JDeveloper:** Oracle JDeveloper es un entorno de desarrollo integrado y gratuito que simplifica el desarrollo de aplicaciones basadas en Java.
Lenguajes: Java, HTML, XML, SQL, PL/SQL, Javascript, PHP, Oracle ADF, UML y otros.
- **PHPSTORM :** IDE comercial para PHP. Es compatible con PHP 5.3 / 5.4 / 5.5 / 5.6 / 7.0 / 7.1 / 7.2, proporciona prevención de errores sobre la marcha, mejor autocompletado y refactorización de código, editor extendido de HTML, CSS y JavaScript.

La decisión de utilizar uno u otro IDE dependerá del ERP/CRM que contrates y del lenguaje sobre el que vayas a trabajar.

Una vez hecha esa elección, puedes escoger uno de los IDEs con los que tengas mas confianza, aunque siempre es preferible elegir uno altamente extendido.

La elección siempre va a estar marcada por el mercado. Piensa que, a medida que necesites interactuar con el ERP/CRM, puede que necesites a otra gente. Podrías ir conectando distintos IDEs sobre el mismo programa, pero eso no suele terminar bien ya que cada IDE acostumbra a dejar ficheros de control que él reconocerá, pero no los demás IDEs. Por eso, **no es conveniente dejar dos IDEs apuntando al mismo**.

Si más de un desarrollador va a trabajar con el IDE, deberíais activar el **control de versiones** que la mayoría lleva incorporado, para que cada programador disponga de su propia versión y podáis reaccionar ante cualquier incidencia.



En el siguiente apartado te mostraremos cómo instalar un IDE. En este caso, lo haremos con Eclipse.

Características de Eclipse

Eclipse es un IDE muy flexible y por esta razón hemos elegido trabajar con él.

Pero, además de su flexibilidad, Eclipse posee **características** que hacen que sea uno de los IDEs con más usuarios. Veamos cuáles son:

- **Perspectivas, editores y vistas:** en Eclipse el concepto de trabajo está basado en las perspectivas, es decir, una preconfiguración de ventanas y editores relacionados entre sí, que nos permiten trabajar en un determinado entorno de trabajo de forma óptima.
- **Gestión de proyectos:** el desarrollo sobre Eclipse se basa en los proyectos, que son el conjunto de recursos relacionados entre sí, como pueden ser código fuente, documentación, ficheros de configuración, árbol de directorios, etc.
- **Depurador de código:** Eclipse incluye un potente depurador, de uso fácil e intuitivo, y que nos ayuda a mejorar visualmente nuestro código. Para ello, solo debemos ejecutar el programa en modo *depuración* (con un simple botón).
- **Extensa colección de plugins:** publicados por Eclipse y también por terceros. Al haber sido un estándar de facto durante tanto tiempo, la colección disponible es muy amplia. Los hay gratuitos, de pago, bajo distintas licencias... Para casi cualquier cosa que nos imaginemos, tenemos el *plugin* adecuado.
- **Sin instalación:** esta característica nos permite tener distintas instalaciones de Eclipse en la misma máquina, lo que es muy útil, por ejemplo, si tienes que programar en Java y en PHP.

Eclipse consiste en un **Core** que es capaz de recibir todo tipo de *plugin*. Realmente, la adaptación a los distintos lenguajes se hace añadiendo o quitando *plugins*, pero no siempre es fácil encontrar el conjunto de *plugins* adecuado, y que sus versiones sean compatibles, además. Piensa que, como hemos dicho, el repositorio de *plugins* proviene de muchos orígenes, y algunos son públicos.

En la web de Eclipse tenemos descargas preparadas para cada trabajo y, dada la posibilidad de conservar en nuestra máquina múltiples configuraciones, ¿por qué no hacerlo?

 ¡Ojo! Recuerda que, para que Eclipse funcione, es necesario tener instalado Java.

Instalar Eclipse

Antes de empezar, recuerda que Eclipse nos puede servir si trabajamos con PHP, Java o Python. Pero si queremos utilizar los ERP/CRM de Microsoft, debemos instalar Visual Source Safe o similar.

- ❶ Además de Eclipse, debemos tener Git, un control de versiones que permite retroceder un módulo cuando nos equivocamos, o trabajar en una variante del módulo dejando el actual sin tocar, utilizar una modificación o desecharla, controlar versiones, liberar por versión... Todo lo que un programador puede necesitar.

Vamos a ver los **pasos para descargar e instalar Eclipse**

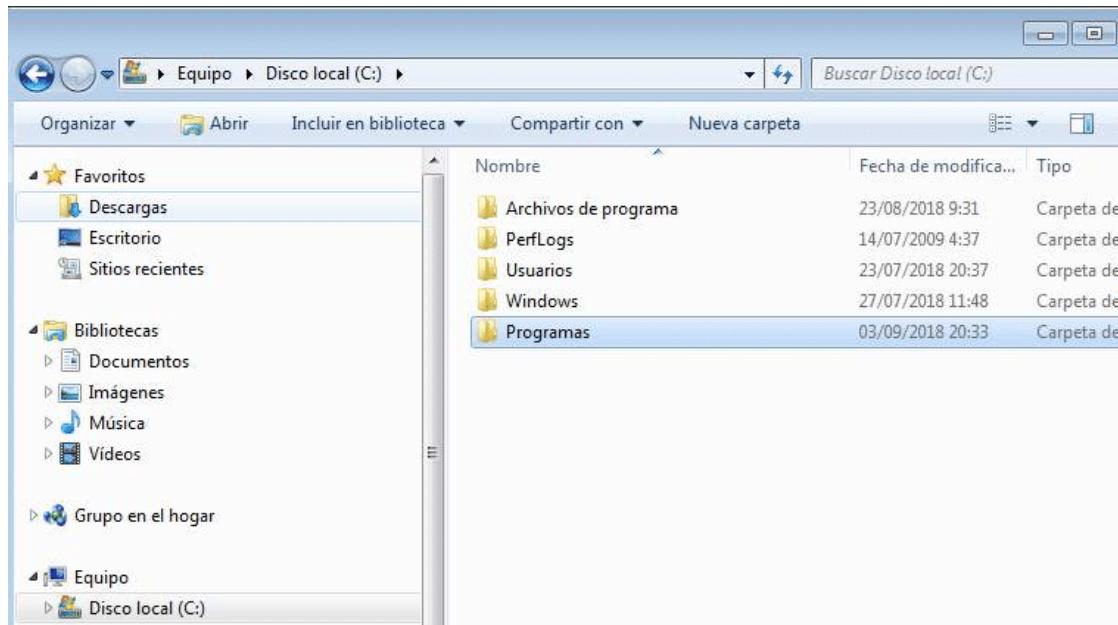
Nos vamos a <https://www.eclipse.org/downloads/packages/> y descargamos el fichero que nos interesa.

The screenshot shows the Eclipse Packages page on the Eclipse Foundation website. It lists several editions of the Eclipse IDE:

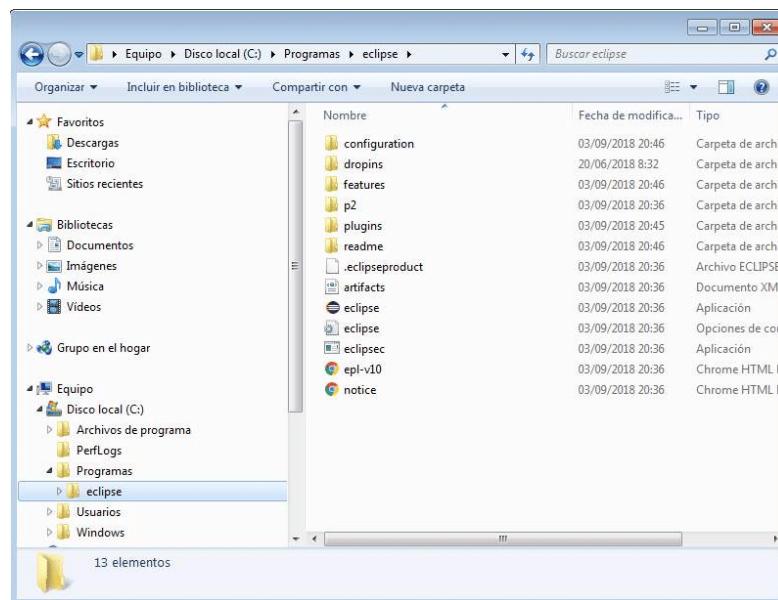
- Eclipse IDE for Eclipse Committee**: Includes Yatta Launcher for Eclipse.
- Eclipse IDE for C/C++ Developers**
- Eclipse IDE for Java and GDI Developers**
- Eclipse IDE for Java Developers**
- Eclipse IDE for JavaScript and Web Developers**
- Eclipse IDE for Java EE Developers**
- Eclipse IDE for Scientific Computing**
- Eclipse IDE for PHP Developers**
- Eclipse IDE for RCP and RAP Developers**

Each entry provides a download link for Windows, Mac OS X, and Linux. A sidebar on the right includes links for related Eclipse products like Photon and related links such as Eclipse 4, Eclipse 3.8, and Eclipse 3.7.

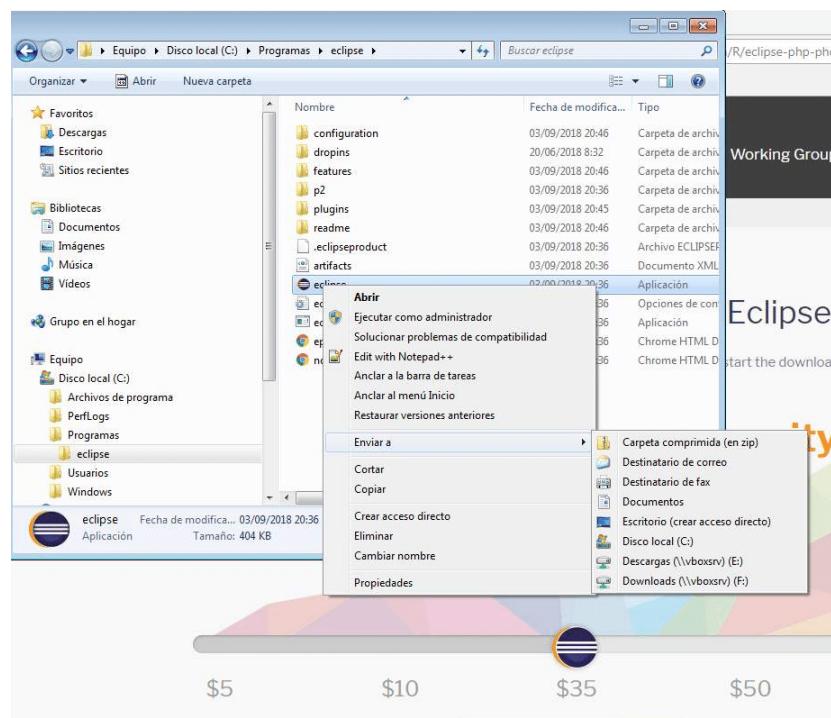
Descargamos el fichero elegido. Cuando finalice la descarga, podremos descomprimirlo . Una buena costumbre es tener una carpeta en el disco C, que se llame *Programas*, para todos estos complementos que no requieren instalación.



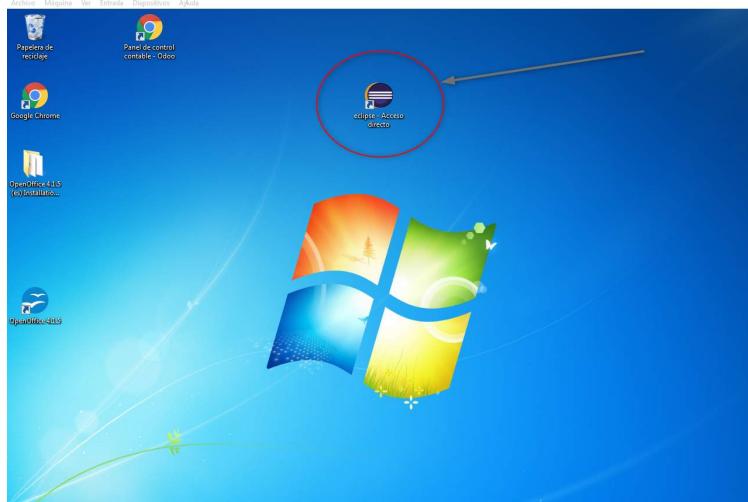
De esta forma, y una vez finalizada la descarga, descomprimimos sobre esa carpeta, para que quede así:



Ahora, para dejar Eclipse accesible desde el escritorio, colocamos el cursor en la aplicación Eclipse, botón derecho de ratón, Escritorio, y clic izquierdo.

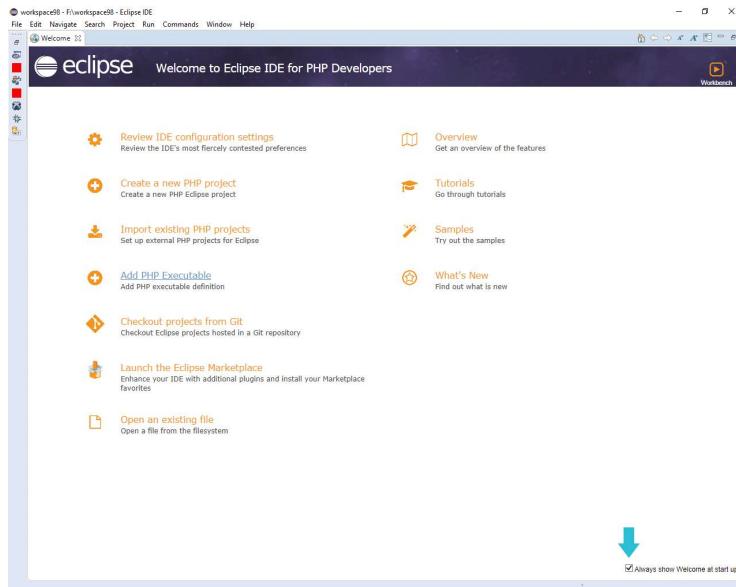
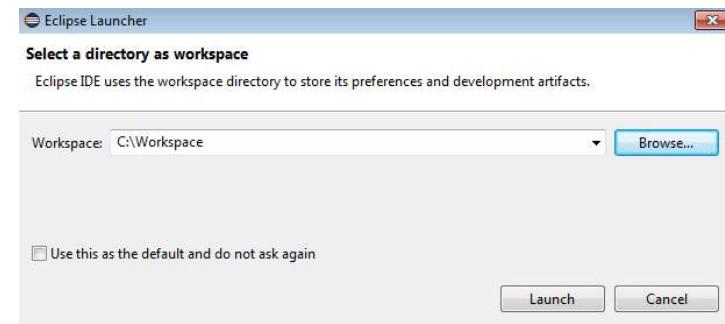


Al momento, nos aparece nuestro enlace en el escritorio.



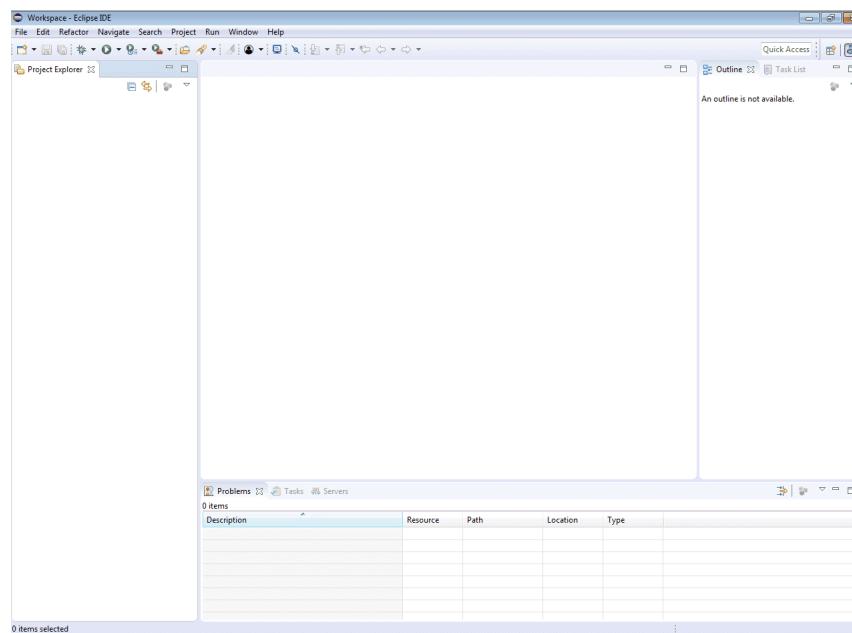
Ahora, podemos **abrir** este enlace con un doble clic.

Lo primero que nos pide es el **área de trabajo**. Se trata de la zona del disco que utilizará para guardar los ficheros sobre los que trabajará el programador. Le asignamos una zona limpia y Eclipse arranca con su pantalla inicial.



En esta pantalla, desmarcaremos el *checkbox*, para evitar que nos la muestre cada vez; de cualquier forma, siempre tendremos disponible esta pantalla en el menú *Help->Welcome*. Ten en cuenta que con Eclipse podemos obtener tutoriales y programas ejemplo que nos pueden ayudar mucho en nuestra formación.

Podemos cerrar esta pantalla con la X que aparece al lado de la palabra *Welcome*, justo encima de Eclipse. Ahora, veremos el área de trabajo que nos ofrece este IDE.

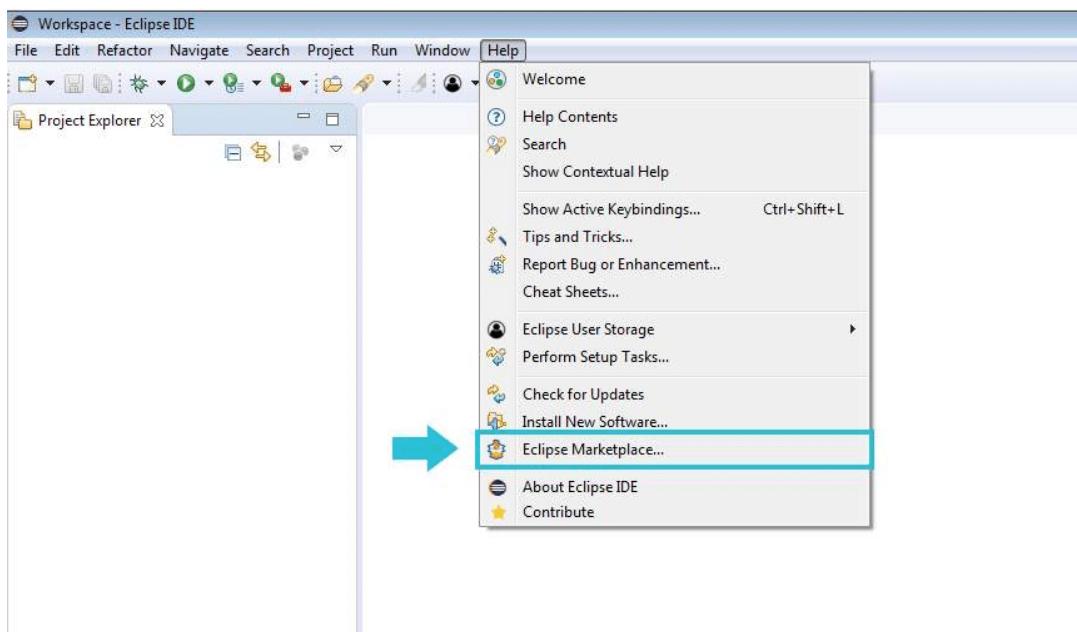


Trabajar con Odoo en Eclipse

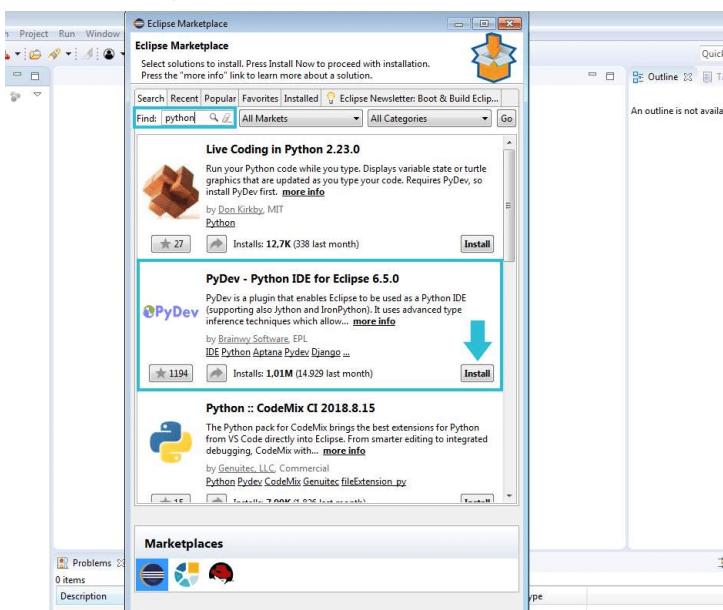
Ahora vamos a ver cómo trabajar con Odoo desde Eclipse

Hemos descargado la versión para PHP (Eclipse + PDT) por comodidad, ya que instala Git y alguna otra cosa. Pero si, por ejemplo, vamos a **trabajar con Odoo**, necesitaremos programar en Python y deberíamos instalar herramientas para que nos ayuden en la codificación.

Para eso, nos vamos al menú *Help* y seleccionamos *Eclipse Marketplace*.

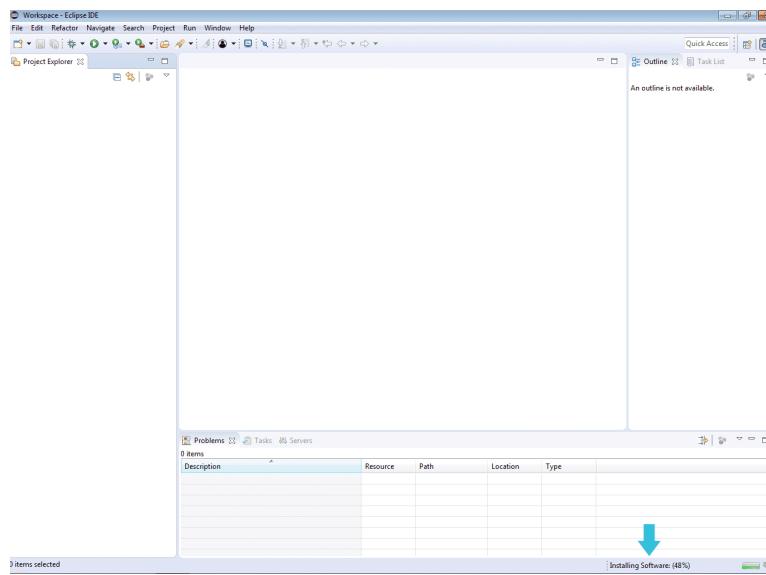


En la ventana que nos aparece, podemos escribir en la casilla de búsqueda "python" y nos aparecerán todos los *plugins* que hay para ese lenguaje.



Elegimos uno y pulsamos "Install"; luego, deberás confirmar que lo quieras instalar y aceptar la licencia.

Con eso, se iniciará la descarga. Fíjate en la parte inferior derecha de la pantalla:

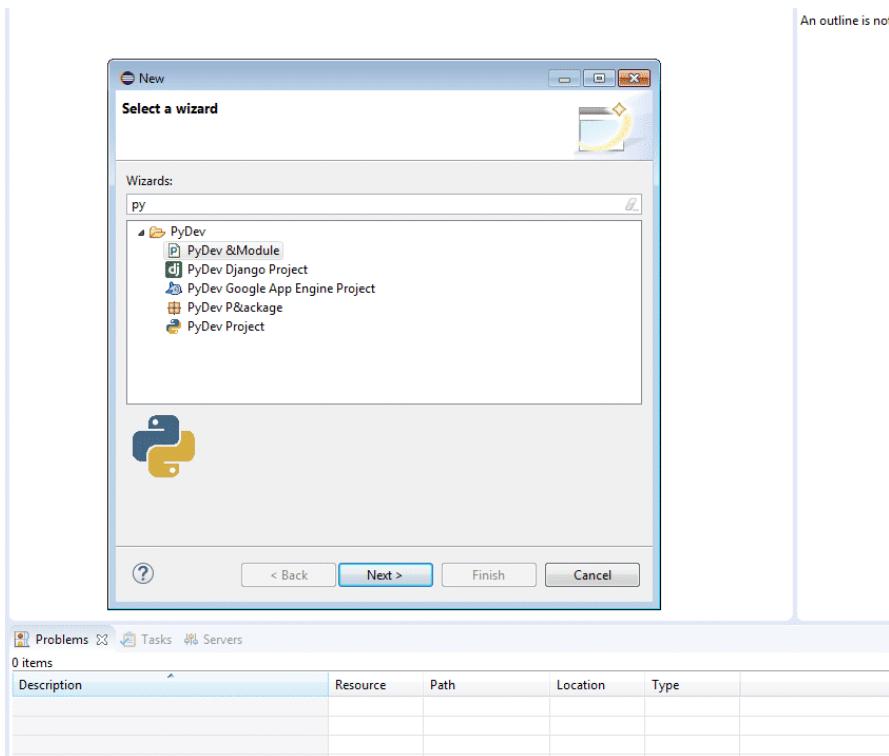


Aunque hayamos vuelto a la ventana de trabajo, Eclipse nos va informando de que está descargando e instalando el *plugin* solicitado. En este caso, cuando termine nos solicitará que reiniciemos Eclipse, y así lo hacemos. Eclipse se cierra y vuelve a arrancar, con el *plugin* instalado.

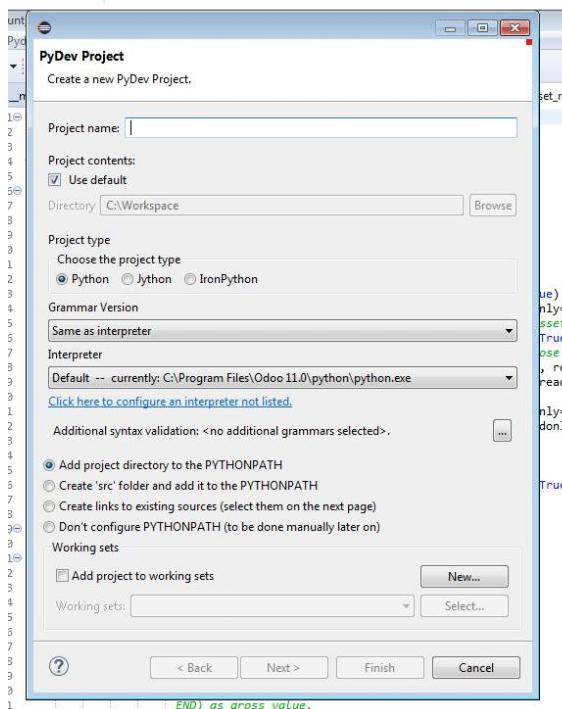
Mientras tanto, podríamos ir creando el proyecto. Como sabes, **Eclipse trabaja por proyectos** y, para nosotros, lo más sencillo es crear ese proyecto, de tal forma que contemplemos todos los programas que podamos necesitar.

Si trabajamos con *Control de versiones*, se hace un poco mas complicado, por lo que vamos a explicar cómo podríamos trabajar en un ordenador que tenga instalada una versión de la plataforma sobre la que podamos hacer pruebas. La idea es sencilla: creamos el proyecto y dejamos una carpeta apuntando a donde están las fuentes; eso nos permitirá ir modificando y visualizando los resultados.

Para crear el proyecto, seleccionamos el menú "**New**" y "**Other**", ya que, como todavía no hemos creado ningún proyecto, no tenemos proyectos Python.

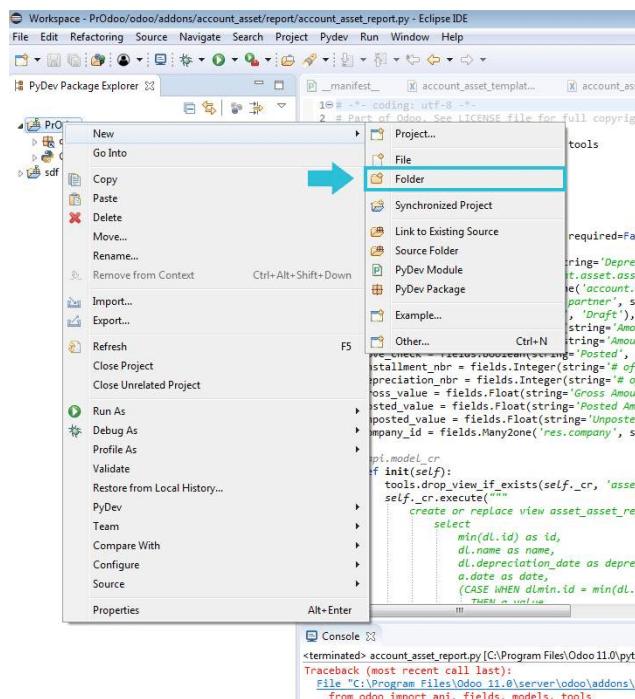


Si, en la parte superior de la pantalla que aparece, indicamos "py", vemos todo lo que tiene de python y comprobamos que el último registro es un proyecto. Lo seleccionamos y pulsamos en "**Next**".



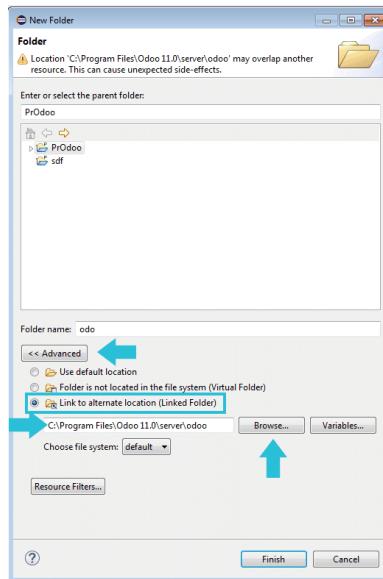
En la nueva pantalla, podemos **indicar el nombre del proyecto** y dejar el resto igual. Cuando indiquemos el nombre, se activará la opción "**Finish**", que podremos pulsar, dejando el proyecto creado.

El siguiente paso es **crear una carpeta** que enlace con el área de Odoo. Eso lo hacemos colocándonos sobre el proyecto y, con el botón derecho del ratón, seleccionamos "**Folder**".

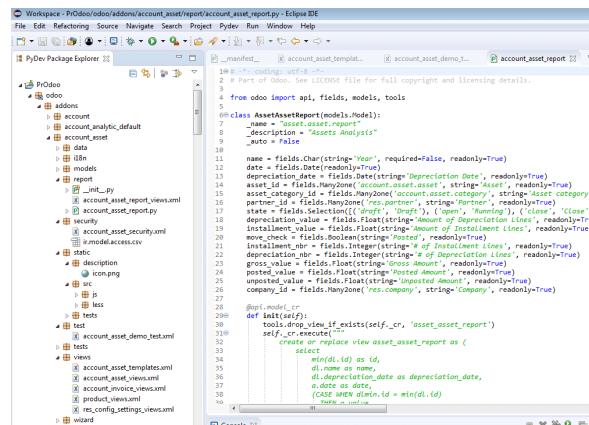


Nos aparece la pantalla para definir la carpeta, en la que indicamos el nombre.

Pulsamos "Advanced" para poder indicar que es un link, y con el botón de "Browse" le indicamos la carpeta:



Cuando pulsamos "Finish" tenemos la carpeta creada, y los módulos que componen nuestro ERP/CRM aparecen en su sitio.



Ya podemos abrir el módulo que nos interese y modificarlo.

Para probarlo, solo tendremos que arrancar la aplicación y ver cómo se comporta el módulo recién modificado.

Organizar el trabajo

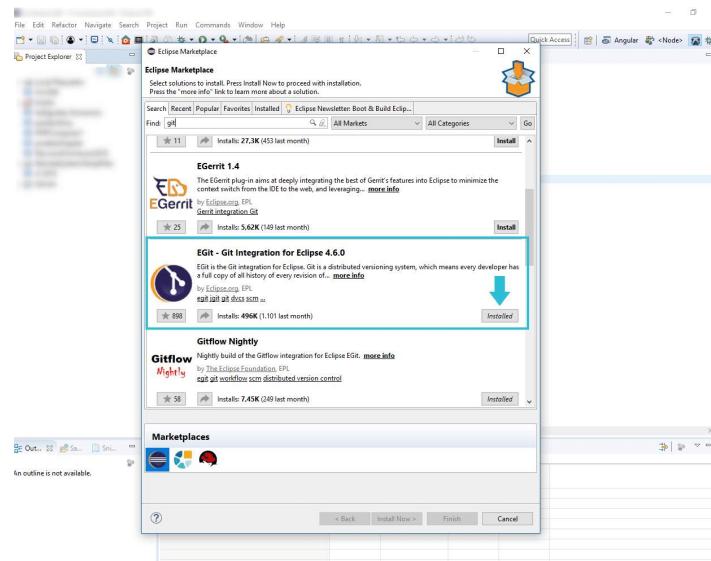
Organizar el trabajo significa que estamos guardando cada cosa en su sitio correspondiente.

Nuestro siguiente paso es ver cómo podemos organizarnos para conseguir que todo funcione.

La organización del trabajo

Eclipse nos facilita la interacción con los programas gracias a potentes editores, pero necesitamos cubrirnos ante la eventualidad de que, por accidente, borremos algún modulo o, incluso, alguna instrucción.

Este trabajo nos lo va a solucionar un **gestor de versiones** que hemos instalado durante la instalación de Eclipse. Pero, por si la versión que tenemos no lo ha hecho, podemos asegurarnos volviendo a Help->MarketPlace, y buscando Git.



Observa que el *plugin egit*, que era el que queríamos, ya está instalado, por lo que podemos cerrar esa pantalla sin hacer nada mas (*Cancel*).

Organización de los directorios

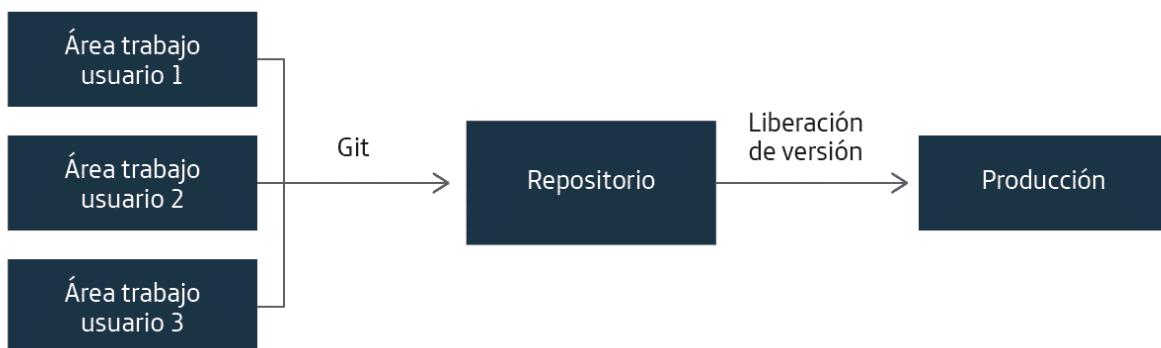
El siguiente paso es ver **cómo organizamos los directorios**, ya que puede haber cualquier número de programadores. Incluso aunque haya un solo programador, una estructura como la que vamos a comentar siempre te será útil.

Ya hemos hablado del área de trabajo que Eclipse necesita para cada usuario; supongamos que lo declaramos **área git**, luego veremos cómo. En ese área de trabajo, el usuario podrá ir haciendo las modificaciones que considere oportunas, y las podrá ir marcando como finalizadas cuando crea conveniente.

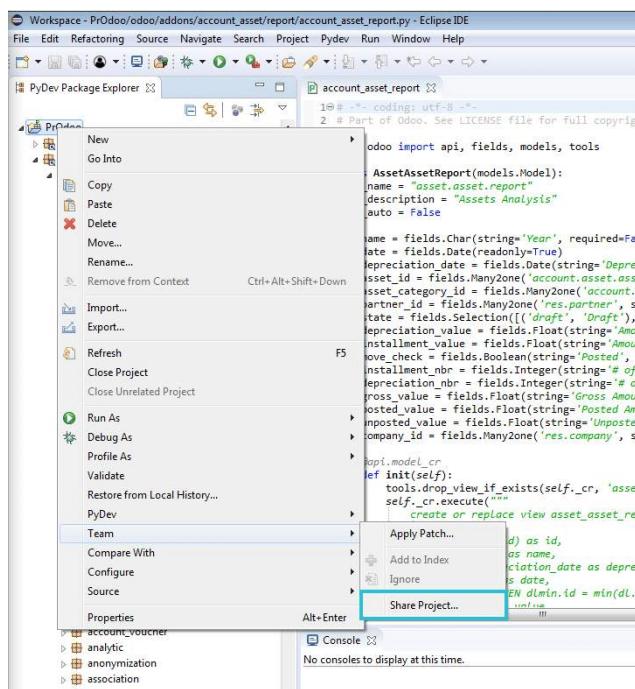
Si creamos un **repositorio centralizado**, mantendremos allí las versiones en curso cuando haya varios programadores y, a la vez, permitiremos que los programadores vayan sincronizando sus copias para poder hacer pruebas. Ese repositorio puede ser una nueva zona Git, o bien estar situado en una máquina local o en una remota, por ejemplo.

Cuando toda la versión esté lista y haya pasado por las correspondientes pruebas, podremos liberarla desde el repositorio y mandarla a producción.

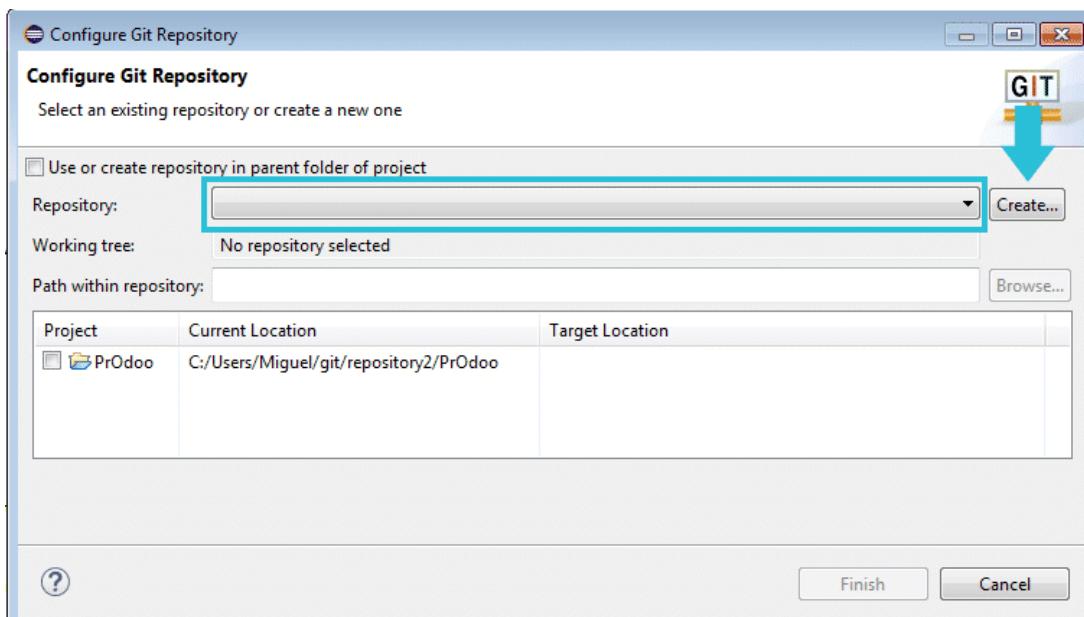
Podrás verlo más claramente en el siguiente diagrama:



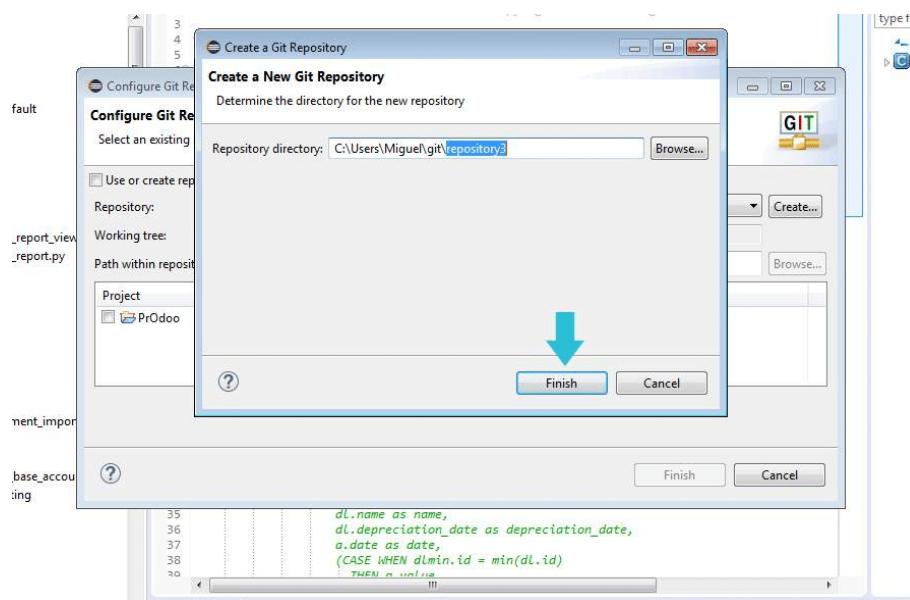
Para activar el control de versiones en el proyecto, basta con situarse en Eclipse encima del nombre de proyecto y con el botón derecho clic en >Team->Share Project.



Como no tenemos creado un **repositorio**, la primera vez nos pedirá que lo creemos.

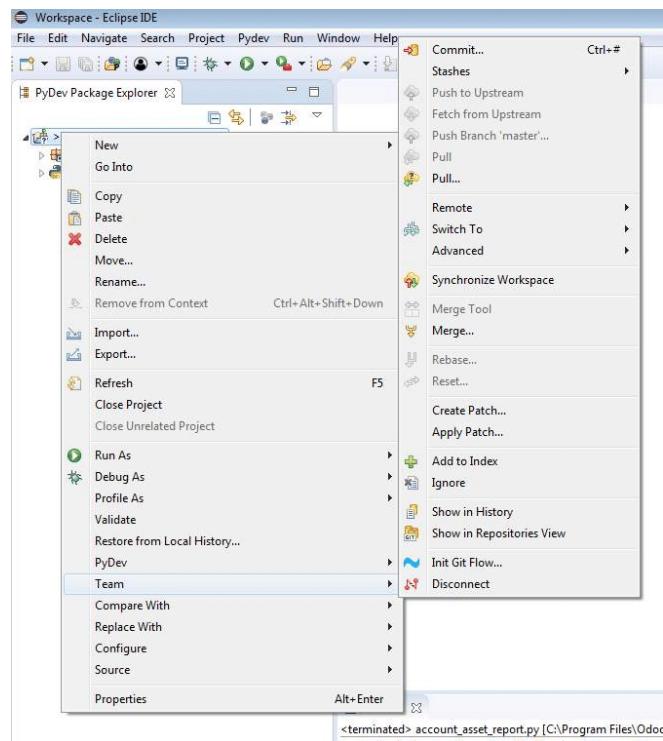


Podemos pulsar en "Create".



Veremos que nos ofrece crearlo dentro de nuestra área personal y, dado que es nuestro repositorio, lo aceptamos pulsando "Finish".

Ahora, nuestro proyecto se encuentra en área Git, aunque **todavía no está indexado**. Si pulsas con el botón derecho sobre el proyecto y eliges **Team**, verás que se nos han abierto un montón de opciones.



Con estas opciones, podremos **controlar nuestras actualizaciones**.

Comprobarás que, en cuanto modifiques algo, el elemento se marca como *modificado*, y deberás marcarlo para liberación cuando creas conveniente.

Resumen

Has terminado la lección, repasemos los puntos más importantes que hemos tratado.

- Hemos hablado de las distintas herramientas genéricas de que disponemos para **modificar nuestro ERP**.
- Como ejemplo, hemos instalado un **IDE, Eclipse**, y le hemos añadido un *plugin*.
- Por último, hemos presentado un **control de versiones** y comentado cómo podemos **organizar nuestro trabajo**.



PROEDUCA