

UD02. INSTALACIÓN Y CONFIGURACIÓN DE UN ERP

Sistemas de Gestión Empresarial

2º Curso // CFGS DAM // Informática y Comunicaciones

Alfredo Oltra

**Cicles
Formatius**

ÍNDIX

1 INTRODUCCIÓN	4
1.1 Contexto histórico de los ERP	4
1.2 Sistemas ERP localizados en la misma empresa	5
1.3 Sistemas ERP en la nube	5
1.4 ¿Qué elegir?	6
1.5 El software	6
2 TIPOS DE INSTALACIÓN DE UN ERP	7
3 HARDWARE Y SOFTWARE DE BASE	9
4 INSTALACIÓN DE ODOO 14 EN DESARROLLO	11
4.1 Desarrollo vs producción	11
4.2 Odoo 14 en Ubuntu Server	12
4.2.1 Instalación	12
4.2.2 Configuración para desarrollo	13
4.2.3 Arrancando Odoo	14
4.2.3.1 Arranque manual	14
4.2.3.2 Arranque automático	15
4.2.4 Errores típicos	15
4.2.4.1 OperationalError: FATAL: role Odoo does not exist	15
4.2.4.2 PostgreSQL no funciona	16
4.2.4.3 Perder la contraseña de la base de datos	16
4.3 Odoo 14 en Docker	16
4.3.1 Odoodock	17
4.3.1.1 Instalación	18
5 PUESTA EN MARCHA	20
5.1 Base datos	20
5.2 Modo <i>debug</i>	21
5.3 Empresa	21
5.4 Idiomas	21
6 AUTORES	22

Versión: 231017.2331

Licencia



Reconocimiento – NoComercial – CompartirIgual (by-nc-sa). No se permite un uso comercial de la obra original ni de las posibles obras derivadas, la distribución de las cuales se debe hacer con una licencia igual a la que regula la obra original.

Nomenclatura

A lo largo de este tema se utilizarán distintos símbolos para distinguir elementos importantes dentro del contenido. Estos símbolos son:

☑ Atención. Importante prestar atención a esta información.

📖 Interesante. Ofrece información sobre algún detalle a tener en cuenta.

1 INTRODUCCIÓN

En esta unidad vamos a estudiar qué factores generales hay que tener en cuenta para la instalación de un sistema ERP, centrándonos en las diferentes formas disponibles para realizar la instalación y configuración de *Odoo*, el ERP que utilizaremos durante este curso.

1.1 Contexto histórico de los ERP

Las posibilidades para la instalación de un sistema ERP se han multiplicado en los últimos años. Las tendencias, posibilidades técnicas y precios del hardware, software y la electricidad han ido modificando la manera en que las empresas usan los ordenadores personales y los servidores.

Haciendo un breve (e inexacto) repaso histórico podemos reflexionar sobre los motivos técnicos y económicos de esta evolución:

- En los primeros años de la informática empresarial el *Mainframe* era casi la única opción: un único ordenador muy grande y caro con varias terminales. Su instalación era interna a la sede de la empresa y el acceso a él a través de un red local (y sin salida al exterior). Además los costes asociados, tanto de compra e instalación como de mantenimiento era muy altos, por lo que sólo estaba al alcance de grandes empresas.
- La llegada de los computadores personales y miniordenadores hizo que muchas empresas más pequeñas utilizasen esta solución.
 - A veces, el servidor era un único ordenador pequeño gestionado por la empresa.
 - Mientras tanto, los *Mainframes* seguían ocupando el espacio de las grandes empresas.
 - En sus inicios, el PC y similares tenían carencias tanto a escala de potencia y como de sistemas operativos (orientados al mercado doméstico y no al mercado profesional). Aun así, su aterrizaje en la vida cotidiana influyó a la sociedad y la industria, haciendo que fabricasen servidores con la arquitectura de los PC y similares pero con una mayor fiabilidad requerida en el contexto industrial.
 - Esto es una gran mejora que permanece hasta hoy día. Los servidores con esta arquitectura mejoraron considerablemente el hardware con procesadores y memorias más estables y sistemas operativos libres y propietarios más robustos (Unix, Linux, Windows NT / Server ...).
 - La llegada de Internet supuso otra alternativa. Ahora las empresas podían utilizar una red común para interconectar sus ordenadores tanto con ordenadores centrales como con servidores remotos que podían tener un programa centralizado.

Al mismo tiempo, algunas empresas pudieron ofrecer servicios por Internet a otras empresas. Este fue el inicio de aquello que denominamos *servicios en la nube*.

1.2 Sistemas ERP localizados en la misma empresa

La opción de tener un servidor en la misma empresa tiene una serie de ventajas e inconvenientes (relacionados entre sí) viniendo dados los principales problemas tanto por los costes económicos como por la seguridad de los datos.

Tener un servidor a la misma empresa con un sistema ERP (o con cualquier otro tipo de servicio) supone algunos retos:

- Poner en marcha un servicio requiere una inversión inicial en hardware.
- El escalado de hardware para aumentar potencia/disminuir potencia es problemático:
 - Habitualmente se tiene hardware infrautilizado.
 - El aumento de potencia de un servicio requiere escalado vertical (aumento de prestaciones de servidor) o escalado horizontal (comprar más equipos).

El hardware limita las posibilidades de cómputo, así que es imposible escalar para aumentar potencia en momentos puntuales.

- El mantenimiento del sistema informático, el suministro de energía y la seguridad de sistema son gastos y responsabilidades asociadas a la empresa.
 - Una ventaja de la gestión de datos interna es que los ordenadores no están fuera de la empresa, por lo que se evitan (o más bien se reducen) riesgos como por ejemplo el espionaje industrial.

1.3 Sistemas ERP en la nube

El gasto tanto energético como de potencia de computación en una sola empresa suele ser desigual durante el día, suponiendo esto una ineficiencia energética. Por este motivo, algunas empresas, especialmente aquellas que ofrecen servicios por Internet, se plantearon vender esta potencia cuando no la necesitaban o compartirla entre muchos clientes. Esto propició el nacimiento de que hoy denominamos "los servicios en la nube".

Con la llegada de los servicios en la nube (que realmente son servidores de otra empresa) las empresas pagan una cuota (fija, por tiempo de cómputo, por uso, etc.), pero a cambio:

- No realizan gastos relacionados con el hardware (instalación y escalado).
- No realizan gastos relacionados con el consumo eléctrico.
- Se reducen de forma drástica los gastos en mantenimiento y seguridad.
- Facilitan el acceso: para operar con estos sistemas las empresas solo necesitan un dispositivo (ordenador personal, smartphone, etc.) con conexión a Internet.
- Si se necesita más potencia, solo hay que contratarla (escalado vertical/horizontal).

Aun así, el uso de servicios en la nube poseen varios inconvenientes:

- En algunos contextos, puede resultar más caro que poner en marcha una infraestructura propia.
- Los datos están almacenados físicamente en un servidor de otra empresa, con posibles problemas relacionados (por ejemplo, espionaje industrial).
- El uso del sistema depende tanto de una buena conexión a Internet como del buen funcionamiento general de la empresa que presta servicios.

1.4 ¿Qué elegir?

No hay una respuesta definitiva a esta pregunta, puesto que depende del contexto y por lo tanto requiere de un estudio previo detallado.

Hoy en día en los entornos empresariales conviven las dos opciones mencionadas, e incluso opciones híbridas (servidor en la empresa, pero apoyo puntual o servicios en la nube).

Los principales factores en la hora de tomar esta decisión son:

- El cumplimiento de las leyes de protección de datos. Este punto influye sobre todo en la decisión de contratar o no un servicio en la nube, puesto que guardando determinadas datos en determinados servicios en la nube podríamos estar incumpliendo la ley.



En España la legislación vigente en materia de protección de datos viene definida por la *LOPDGDD*, mientras que en Europa se rige por el *RGPD*.

- El precio de la electricidad y el consumo del hardware.
 - Existen dispositivos hardware orientados a tener un bajo consumo.
- El precio del hardware.
 - Existen dispositivos orientados a tareas de servidor muy baratos.

1.5 El software

Durante la introducción hemos hablado de costes relacionados con el sistema que alojaría nuestro ERP. Pero ¿qué pasa con el software? Hay multitud de opciones software ERP, tanto gratuitos, de pago, libres, mixtos (partes libres, partes de pago), etc.

Tal y como vimos en la UD01 una de las opciones con mejor aceptación es *Odoo*, que se presenta en dos versiones: *Community Edition* (software libre y gratuito) y *Enterprise Edition* (de pago).

En este curso utilizaremos *Odoo Community Edition*. Por simplicidad, cuando nos referimos a *Odoo*, estaremos refiriéndonos a esta versión.

2 TIPOS DE INSTALACIÓN DE UN ERP

La variedad de sistemas y de necesidades hacen que existan diferentes maneras de realizar la instalación de un ERP. Vamos a analizar varias de ellas, empezando con las soluciones no recomendadas, para pasar aquellas simples pero correctas e ir evolucionando hacia las más sofisticadas:

- La carpeta compartida. Se trata de una práctica cada vez menos usada, pero que se mantiene en muchas pequeñas empresas.
 - Hace años, algunos programas de gestión estaban pensados para un solo usuario. La única opción que daban para poder ser accedidos por varios ordenadores en una red local era compartir la carpeta de la base de datos y configurar varias instalaciones para acceder al mismo archivo. Además, esta “base de datos” era un fichero de *Microsoft Access* o ficheros de texto plano.
 - Esta solución no es correcta ni recomendable ya que es una solución que propone muchos problemas, tanto con el acceso simultáneo, como problemas de seguridad y de integridad.
- Instalación *On-Premise*: aquí entra en juego un servidor instalado a la misma empresa.
 - Los ordenadores clientes pueden acceder a sistema ERP mediante un software cliente del mismo ERP o incluso mediante un navegador web.
 - Se tiene que configurar correctamente la red del sistema tanto para evitar conexiones externas no permitidas, como para proporcionar acceso seguro (si se requiere) desde fuera de la red.
 - Si se necesita acceder desde fuera de la red local, hay que configurar correctamente la red y la seguridad del acceso externo. Se suele pagar por el software entero o la instalación y el mantenimiento.

Si el servidor funciona de una forma segura y esta opción es viable económicamente para la empresa, es una solución correcta.

- Instalación como servicio en la nube. En esta solución se prescinde del servidor en la empresa y se subcontrata la computación, lo que simplifica la instalación y el acceso externo. Además, se paga por lo que se necesita y es fácilmente escalable. Los servicios en la nube puede ser de muchos tipos, pero se suele distinguir entre:
 - *IaaS* (Infraestructura como Servicio): nos proporcionan acceso a servidores, máquinas virtuales o contenedores. Hay que instalar y proteger el sistema operativo, el ERP y todo el resto. Quizás la empresa de la nube ya nos lo ofrezca con un sistema operativo preinstalado, pero tendríamos que tener control total de este sistema operativo.

Ejemplo: los VPS entran dentro de este tipo de nube.

- *PaaS* (Plataforma como Servicio). En esta nube ya existe el sistema operativo y algunos programas configurados. Sobre este sistema se puede desplegar el sistema ERP.

Ejemplo: el que consideramos como hosting tradicional (PHP, MySQL, etc.) o sistemas más específicos como *Heorku*.

- SaaS (Software como Servicio). En este caso ya está el ERP instalado y nos dan acceso a determinadas características según contratamos (cantidad de usuarios, acceso simultáneo, espacio de almacenamiento, copia de seguridad, etc.). No hay que preocuparse de nada más, pero se está limitado al programa y características que el proveedor ofrece.

Ejemplo: soluciones tipos *Gmail* o *One Drive*, donde tenemos todo puesto en marcha y sólo consumimos el servicio.

- Aparte de estos tipos de servicios en la nube, podemos tener algunos servicios de forma individual en la nube, como por ejemplo servicios de bases de datos (como Firebase) o servicios de *APIs REST* o *GRAPHQL*, etc. En cualquier caso, tienen que garantizar una alta disponibilidad, seguridad y escalabilidad.

3 HARDWARE Y SOFTWARE DE BASE

Cada sistema ERP es diferente y posee necesidades de potencia diferentes. Estas necesidades suelen depender de factores como:

- Cantidad de datos que alberga el sistema.
- Número de usuarios simultáneos.
- Tecnología utilizada por el software ERP.
- Sistema operativo sobre el cual corre el software ERP.

Para poner en marcha un sistema ERP es recomendable utilizar hardware diseñado para servidores. La calidad de los componentes, la refrigeración o tecnologías como el sistema ECC en la memoria RAM, hacen más estables estos ordenadores ante ordenadores domésticos u otras alternativas, más exóticas como la *Raspberry Pi*, *Arduino*, etc.



Los usuarios que interactúan con el sistema ERP, sí que pueden usar PC, tablets, smartphones u otros sistemas domésticos como cliente del servicio.

Otro aspecto a tener en cuenta es la seguridad, teniendo en cuenta tanto medidas de seguridad pasiva como activa. Se tienen que tener en cuenta aspectos como:

- Protección contra personas (ataques físicos y remotos).
- Protección contra los elementos: calor, humedad, sobretensiones o incendios.
- Disponibilidad de sistema y copia de seguridad.
- Cumplimiento de la legislación vigente.

No es tarea de este módulo explorar todas las protecciones disponibles, pero se tienen que tener en cuenta y exigir las en una instalación real. Algunas de las medidas de protección básicas son:

- El servidor tiene que tener un *SAI* que lo proteja y lo mantenga encendido en caso de fallo de la red eléctrica.
- Es recomendable que el servidor en si disponga de elementos (procesador, RAM, discos, etc.) por encima de la potencia mínima necesaria porque funcione el sistema ERP.

Esa mejora no ha de ser muy grande, puesto que si lo hacemos incrementaremos gastos en hardware y de consumo eléctrico sin obtener un gran beneficio. Normalmente estos programas tienen una documentación en la cual describen los requerimientos mínimos.

- Respecto a la seguridad de los datos, se recomienda redundancia en los discos, ya sea con RAID o con sistemas de archivos redundantes como *ZFS* o *Btrfs*.



Esta redundancia no excluye la necesidad de establecer una política de copias de seguridad externas al sistema.

Una vez puesto en marcha el servidor físico, hay que elegir el sistema operativo base y un posible sistema de virtualización. En el caso del sistema *ERP Odoo*, que tratamos en estos apuntes, para una puesta en marcha en un sistema real se recomienda *Ubuntu Server*.

Este sistema operativo puede ser el sistema instalado en la máquina o estar virtualizado. La virtualización podemos realizarla con:

- Máquinas virtuales con hipervisor (tipo *Virtualbox*). Estas máquinas realizan una simulación completa del hardware, influyendo esto en una disminución de rendimiento.
- Una alternativa a la virtualización con hipervisor, pero con mejor rendimiento (próximo al rendimiento nativo) son los contenedores, sea completos como *LXD* o contenedores de aplicaciones como *Docker*.
- Además existen sistemas operativos base como *Proxmox* que simplifican la gestión tanto de máquinas virtuales como de contenedores, copias de seguridad de los mismos y almacenamiento en red.

En caso de optar por la nube, si contratamos un *IaaS* también tenemos que tener en cuenta la potencia contratada. De hecho, en este caso es más importante afinar correctamente que en una instalación *On-Premise*, puesto que podemos incrementar los costes sin tener un beneficio en cuanto a rendimiento.

4 INSTALACIÓN DE ODOO 14 EN DESARROLLO



En el momento de escribir esta documentación, la última versión disponible de *Odoo* es la 16. Se ha decidido seguir con la 14 por razones de estabilidad.

De manera ideal, *Odoo 14* no necesita mucha potencia para funcionar. De hecho debería funcionar sin problemas en cualquier ordenador con varios núcleos y al menos 512MB de RAM. Esta configuración, aunque puede valer como punto de partida pierde validez en cuanto crece el número de accesos simultáneos.

Cómo en todas las aplicaciones que consultan bases de datos, el acceso al disco puede suponer un cuello de botella. Por eso es recomendable utilizar unidades SSD, RAIDs o sistemas de archivos como ZFS o Btrfs con varios discos. Y por supuesto cuanto más RAM tengan mejor.

Odoo 14 funciona perfectamente en máquinas virtuales y contenedores. Algunas opciones de configuración pueden ser:

1. Sistema operativo *Ubuntu Server* e instalación directa de *Odoo*.
2. Sistema operativo *Ubuntu Server* y virtualización con KVM o similar. Las máquinas virtuales tendrán instalado *Ubuntu Server*.
3. Sistema operativo *Ubuntu Server* con contenedores con LXD de *Ubuntu*.
4. Sistema operativo *Ubuntu Server* con contenedores *Docker*.
5. Sistema operativo *Proxmox* con máquinas virtuales o contenedores LXC gestionados por *Proxmox*.



Como se ve, en todas las ocasiones se eligen máquinas reales y virtuales *Ubuntu*. Esto es porque *Odoo* está desarrollado en este sistema y esto nos ayuda a garantizar que la implantación de sistema funcione correctamente. Eso no quita que exista una versión instalable nativa para *Windows*, pero no es la opción recomendable.

Evidentemente, resulta inabarcable tratar cada uno de estas opciones en la documentación, por lo que únicamente nos vamos a centrar en las dos más usadas, especialmente cuando hablamos de entornos de desarrollo: instalación directa o uso de contenedores *Docker*.

4.1 Desarrollo vs producción

El objetivo de este módulo no es tan solo desplegar un sistema ERP, sino aprender a desarrollar para estos sistemas. Por eso, una vez instalado *Odoo*, vamos a configurarlo para poder desarrollar con él.

En *Odoo* el entorno necesario para producción es diferente al entorno por desarrollo. De hecho es altamente recomendable que los dos sean sistemas separados.

El desarrollo de módulos implica probar cosas que pueden corromper datos, evitar que el servidor haya de pararse y arrancarse, etc.

¿Por qué separar producción y desarrollo?

1. El desarrollo de módulos implica probar cosas que pueden corromper datos, evitar que el servidor haya de pararse y arrancarse, etc.
2. Algunas configuraciones utilizadas en entornos de desarrollo, no son adecuadas para sistemas de producción por motivos de rendimiento y/o seguridad.
3. Algunos pasos del proceso natural de desarrollo pueden dejar residuos en la base de datos que pueden ser problemáticos. A veces podemos causar daños a sistema ERP de forma que sea más sencillo reinstalar el sistema de cero de intentar reparar el daño.

Algunos aspectos a tener en cuenta según el tipo de servidor:

- Servidor en producción. Para un servidor en producción tendremos que configurar correctamente el servicio en `/etc/odoo/odoo.conf`, utilizando una buena contraseña maestra y la ruta de los módulos adicionales en caso de haberla.

Además, se tiene que configurar correctamente las reglas de los cortafuegos y proporcionar acceso por HTTPS mediante un servidor que actúe como servidor intermediario (por ejemplo *Nginx*).

- Servidor de desarrollo. No hay que configurar una gran seguridad, pero es interesante tener un directorio fácil de encontrar para los módulos nuevos y una configuración específica para cuando arrancamos el servicio manualmente al actualizar un módulo.

4.2 Odoo 14 en Ubuntu Server

4.2.1 Instalación

Aunque existen varias opciones para realizar este tipo de instalación, vamos a centrarnos en explicar la que se lleva a cabo mediante un paquete *.deb*.

El primer paso es la comprobación de los idiomas instalados. En caso de querer realizar la instalación en un idioma diferente al inglés, es necesario hacer algunas comprobaciones previas. De hecho, algunos sistemas o VPS tienen configurado únicamente el idioma inglés. Eso podría provocar que una instalación de la base de datos *PostgreSQL* se configurará automáticamente en ASCII extendido y no aceptará caracteres específicos de ciertas lenguas. Por eso es recomendable configurar el idioma del sistema base:

```
$ dpkg-reconfigure locales
```

Ubuntu tiene en sus repositorios oficiales *Odoo*, pero suele ser una versión antigua. Es por ellos que hay que indicarle al gestor de paquetes como obtener la versión que nos interesa. Para ello hay que ejecutar como *root* o con un usuario *sudoer*:

```
$ apt-get update
$ apt-get install ca-certificates wget gnupg
$ wget -O - https://nightly.odoo.com/odoo.key | apt-key add - echo
"deb http://nightly.odoo.com/14.0/nightly/deb/ ./" >
/etc/apt/sources.list.d/odoo14.list
$ apt-get update && apt-get install odoo
```

Mediante las órdenes anteriores lo que hemos hecho es:

1. Actualizar las aplicaciones y recursos que son necesario para el proceso de instalación (generalmente ya están instalados en el sistema).
2. Instalar el certificado de *Odoo* para que nuestro sistema confíe en los repositorios que vamos a incluir.
3. Crear el fichero (*/etc/apt/sources.list.d/odoo14.list*) con la información de los repositorios de *Odoo*.
4. Finalmente instalar *Odoo 14*.



La versión instalada se trata de la versión nightly. Es decir, instalarás una versión que se actualiza cada noche con los cambios de *Odoo*.

4.2.2 Configuración para desarrollo

Cuando se instala *Odoo* mediante un repositorio (como hemos hecho en este documento), se crea automáticamente un usuario de sistema llamado *odoo*. El servicio arranca con este usuario por motivos de seguridad y aislamiento. El usuario *odoo* tiene asignados los justo y suficientes derechos para que todo el sistema funcione sin problemas y de esta manera evitar usar *root*.

De hecho, en un servidor de desarrollo, es una buena idea que el usuario con el cual se trabaje en el servidor de *Odoo* para el desarrollo sea el usuario *odoo*. Para eso podemos darle una contraseña y hacer que su shell sea */bin/bash*:

```
$ sudo passwd odoo
$ sudo usermod -s /bin/bash odoo
```



Es muy importante tener en cuenta que esta última parte no se aplica a un servidor de producción, donde no es buena idea que los usuarios que controlan servicios tengan acceso al intérprete de órdenes..

Una vez el usuario tiene shell y accedemos a él, es muy probable que su directorio personal sea */var/lib/odoo*. No hay que cambiar esto y aquí podemos crear nuestros módulos personales.

Por último, *Odoo* necesita *PostgreSQL* para funcionar. Antes que nada, tenemos que asegurarnos que está funcionando. Podemos hacerlo con:

```
$ service postgresql start
```

En caso de que el servicio no se encuentre instalado es posible hacerlo mediante:

```
$ sudo apt install postgresql postgresql-contrib
```

Una vez tenemos el servicio arrancado, nos pasamos al usuario *postgres* con:

```
$ sudo -u postgres psql
```

Y ejecutamos (suponiendo que el usuario que lanza el servicio es *odoo*):

```
$ createuser odoo
```

Si fuera otro usuario (*root*, *www*, *etc.*) cambiaríamos *odoo* por el usuario que se encargue de poner el servicio. Esta acción va a incluir al usuario *odoo* como usuario de *PostgreSQL* y, por lo tanto, le permitirá realizar operaciones sobre la base de datos al usuario. Después de esto ya podemos lanzar el servicio *Odoo* sin problemas.

4.2.3 Arrancando Odoo

Odoo puede arrancar de dos formas:

- Manualmente, invocando el mando *odoo*
- Automáticamente, como servicio de sistema.

4.2.3.1 Arranque manual

Para arrancar *Odoo* de manera manual, simplemente lo lanzamos con el comando:

```
$ odoo
```

Este comando arrancará el sistema siguiendo alguna configuración específica (fichero *.odoorc* ubicado dentro del *home* del usuario que lo ha lanzado) o si no hay, utilizando la configuración base de *Odoo* presente en el fichero */etc/odoo/odoo.conf*.

Para preparar *Odoo* para un sistema de desarrollo, tenemos que indicarle que los módulos que utilizará estarán tanto en el directorio oficial como en nuestro *home*. Podemos hacerlo con una orden parecida a:

```
$ odoo --addons-path="/var/lib/odoo/modules,/usr/lib/python3/dist-packages/odoo/addons" --save
```

Como se puede ver, el comando *odoo* permite utilizar la opción *--addons-path* para definir nuevas rutas en las que haya módulos, de manera que en esa ejecución puedan ser utilizados. Para evitar la definición de ese parámetro cada vez que ejecutamos *odoo*, utilizamos la opción *--save*, que guarda esas rutas, junto a otras configuraciones, en el fichero *.odoorc* del directorio personal del usuario *odoo* o del usuario que ejecutó el comando.



Este método puede ser interesante para entornos de desarrollo, pero no tanto en producción ya que es el usuario el responsable de levantar el sistema en caso de que este caiga por cualquier motivo.

4.2.3.2 Arranque automático

Esta opción permite que *Odoo* se ejecute de manera automática siempre que se reinicie el sistema. En este caso *Odoo* funciona como un demonio y guarda su historial (log) en `/var/log/odoo`.

Al contrario que la manual, esta configuración es útil para servicios en producción ya que es más cómodo para depurar. De hecho, en caso de querer depurar, en ocasiones se recomienda parar el servicio y posteriormente arrancar con el comando *odoo*.

Un ejemplo de parada de servicio y posterior arranque manual:

```
$ sudo service odoo stop
$ odoo
```

De esta manera podremos observar el historial (log) en tiempo real por la terminal y será más sencillo detectar algunos problemas. Además, esta forma de arrancar permite cosas como actualizar un módulo en una empresa al reiniciar:

```
$ odoo -u modulo -d empresa
```

Si además queremos lanzarlo como si estuviéramos en un entorno de consola *Python*, podemos hacer:

```
$ odoo shell -u modulo -d empresa
```



Como es de suponer, el comando *odoo* dispone de muchas más opciones que puede ser interesantes conocer (algunas de ellas las iremos viendo a lo largo del curso). Es posible consultarlas en línea desde [la página web de Odoo](#).

4.2.4 Errores típicos

Al realizar la instalación manual en *Ubuntu Server*, hay algunos errores que se suelen repetir si nos saltamos algún paso. Estas son las soluciones a los errores más frecuentes:

4.2.4.1 OperationalError: FATAL: role Odoo does not exist

Este error es debido a una configuración incorrecta de *PostgreSQL*. Suele ocurrir cuando ya estaba instalado el SGBD con configuraciones no compatibles con el instalador de *Odoo*.

Para solucionarlo, simplemente hay que crear el usuario de manera manual:

```
$ su - postgres -c "createuser -s odoo"
```

4.2.4.2 PostgreSQL no funciona

En ocasiones, cuando instalamos una base de datos en un idioma no inglés, el servidor no funciona. Generalmente suele ser un problema relacionado con la configuración de la UTF-8 en la base de datos y, por lo tanto, la solución consiste en cambiar dicha configuración. Para ello desde el usuario *postgres*, arrancamos el comando *psql*:

```
$ su postgres
```

```
$ psql
```

y ejecutamos el siguiente código:

```
postgres=# update pg_database set datallowconn = TRUE where datname =
'template0';
postgres=# \c template0
template0=# update pg_database set datistemplate = FALSE where
datname = 'template1';
template0=# drop database template1;
template0=# create database template1 with template = template0
encoding = 'UTF8';
template0=# update pg_database set datistemplate = TRUE where datname
= 'template1';
template0=# \c template1
template1=# update pg_database set datallowconn = FALSE where datname
= 'template0';
```

4.2.4.3 Perder la contraseña de la base de datos

Un error (demasiado) habitual, es olvidar la contraseña del usuario de una base de datos. Es posible solucionarlo accediendo con el usuario *postgres* y ejecutando con *psql* el siguiente código:

```
update res_users set password='test' where login='admin';
```

4.3 Odoo 14 en Docker

A la hora de arrancar *Odoo* vía *Docker* lo mínimo necesario es la creación de dos contenedores, el servidor de *Odoo* y el servidor de la base de datos *PostgreSQL*.

Una primera solución pasa por la creación de los contenedores de manera separada, pero teniendo en cuenta la necesidad del arranque de los dos servicios parece más recomendable el uso de *Docker Compose*.



Docker Compose es una herramienta que nos facilita el despliegue de varios contenedores utilizando una configuración definida previamente en un fichero denominado por defecto *docker-compose.yml*.

4.3.1 Odoodock

Una opción que permita ejecutar un docker de *Odoo* en modo desarrollo pasa por la creación de un fichero *ymf* básico como el que está disponible en el aula virtual. En él, de manera muy sencilla únicamente se indican las dos imágenes, las variables de entorno que permiten configurarlas (básicamente usuarios y contraseñas), los puertos expuestos y aquellos volúmenes en los que vamos a almacenar la información que se quiere conservar, especialmente la carpeta en la que van a estar ubicados los módulos de desarrollo.

Sin embargo, esta opción resulta demasiado sencilla en cuanto queremos profundizar o desarrollar algo más complejo. Es por ello que una solución más adecuada puede ser *odoodock*.

Odoodock es un entorno de desarrollo para *Odoo* basado *Docker* y pensado con fines educativos, que sigue la idea propuesta por *laradock*.



Laradock es un entorno de desarrollo pensado para *Laravel*, un framework de PHP enfocado al desarrollo de servidores.

Entre sus características podemos encontrar:

- Creación de imagen personalizada con la instalación de varias herramientas de apoyo.
- Soporte para *odoo*, *odoo shell* y *scaffolding* (veremos que es cada cosa en temas posteriores)
- Soporte de varios servicios de apoyo: *pgAdmin*, *Jekyll*, *Moodle*...
- Cada servicio corre en un propio contenedor.
- Basado en imágenes de confianza.
- Scripts para la generación automática de módulos por *scaffolding*, clonado de repositorios *git*, descompresión de paquetes.
- Permite el desarrollo desde dentro del contenedor de *Odoo*.
- Configuración de serie para realizar depuración en *Visual Studio Code*.
- Fácilmente configurable a través de variables de entorno.
- Pensado con fines académicos: código muy comentado.
- Soporte para *Odoo* versión 14 (funcional en *Odoo* 15, pero no testado)



De todas ellas cabe destacar la posibilidad de poder depurar el código. Aunque parece algo trivial, cuando trabajamos con contenedores no lo es. En ese caso el código de *Odoo* está dentro del contenedor y se ejecuta desde él, mientras que el código de los módulos (sobre el que trabajamos) se ejecuta sobre el código de *Odoo* y se

encuentra en un volumen externo. Esta arquitectura complica la depuración, siendo necesario el desarrollo de los módulos desde dentro del contenedor.

4.3.1.1 Instalación

1. Es conveniente crear en local una carpeta, por ejemplo *odoo_dev* (en el resto del documento haremos referencia a ella como *odoo_dev*), en la que ir dejando todo lo necesario para el proyecto: documentación, código, configuraciones, datos...

2. Clonar en su interior *odoodock*

```
$ cd odoo_dev  
$ git clone https://github.com/aoltra/odoodock.git
```

3. Entrar en la carpeta *odoodock*

```
$ cd odoodock
```

4. Copiar el fichero *.env-example* a *.env*

```
$ cp .env-example .env
```

5. Modificar el fichero *.env* para adaptarlo a nuestras necesidades (O).



En un principio la versión de ejemplo es suficiente para empezar a trabajar.

6. Copiar el fichero *.services-example* a *.services*

```
$ cp .services-example .services
```

7. Modificar el fichero *.services* para incluir los servicios que deseamos arrancar. Los servicios se separan por espacios y van entrecomillados.



En un principio únicamente es suficiente arrancar el servicio *web* para empezar a trabajar.



El servicio *web* es obligatorio para arrancar *odoo*. De manera automática se arranca también el servicio *db*.

8. Asignar permisos de ejecución para el usuario al fichero *up.sh* y *create-module.sh*.

```
$ chmod u+x ./up.sh  
$ chmod u+x ./create-module.sh
```

9. Arrancar los servicios

```
$ ./up.sh
```

10. Comprobar que los contenedores están en ejecución. Conviene comprobar que el estado de todos los contenedores es *UP*, al ejecutar

```
$ docker compose ps    0    $ docker ps
```

11. Para comprobar que todo ha ido correctamente, acceder desde un navegador a *localhost:8069* (o la ip del servidor), donde debe aparecer la página del selector de la base de datos.

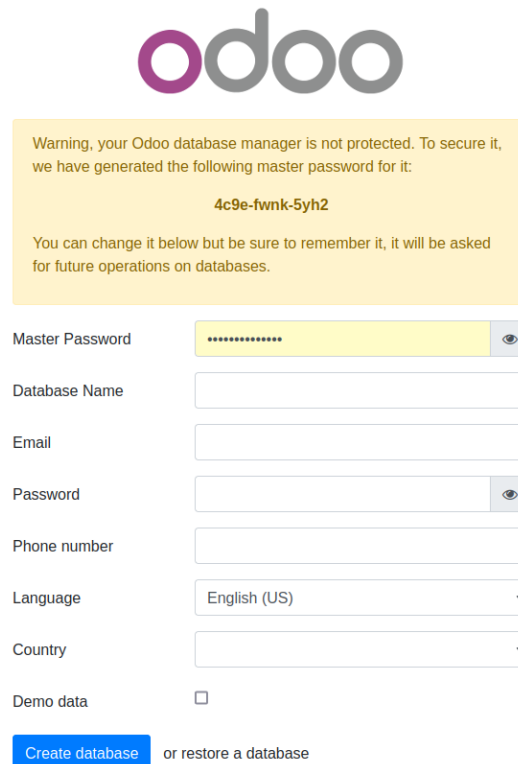


Figura 1. Configuración inicial base de datos



En caso de que esa pantalla no aparezca hay que dar por hecho que existe algún problema con los contenedores. Para obtener información de qué ha sucedido lo mejor apagarlos y reaniciar compose de manera manual sin la opción *-d* (*detach mode*), de tal manera que se muestre por consola toda la información del arranque.



Toda esta información (y más) la puedes ver detallada en la documentación online de [odoodock](https://odoo.com/help/10/es/odoodock.html).

5 PUESTA EN MARCHA

Una vez tenemos *Odoo* instalado (de cualquiera de las maneras) es necesario realizar unas configuraciones mínimas para su puesta en marcha.

5.1 Base datos

1. Configurar la base de datos.

Tal y como se ve en la Figura, los datos solicitados¹ son:

- **Master Password:** es la contraseña general del SGBD. Nos va a permitir, hacer copias de seguridad, restaurarlas, etc. *Odoo* nos propone una, pero podemos modificarla e incluir la nuestra



El *master password* es una contraseña muy importante por lo que conviene almacenarla en un lugar seguro.

- **Nombre de la base de datos:** indica el nombre de la base de datos a crear. Suele ser recomendable tener una política que los nombres permitan fácilmente identificar que contiene o cual es su objetivo.

Odoo siempre necesita, como mínimo, una base de datos para trabajar, aunque puede trabajar con varias de ellas. El acceso a una u otra se realiza desde la ventana de *login*.

- **Email:** email del usuario administrador de *Odoo*. Aunque se solicita un email, es posible introducir un login como *admin*, *root*, *administrador*, o algo similar.
- **Password:** contraseña del usuario administrador.
- **Language:** idioma que será instalado por defecto. Posteriormente pueden instalarse más e indicar cual será el idioma por defecto de cada usuario.
- **Country:** país base sobre el que trabaja *Odoo*, es importante para todo lo referido a la localización que no tienen que ver con el idioma: moneda, unidades de medida, etc, así como para tener en cuenta normativa relativa al país..

2. Una vez cumplimentados todos los campos, podemos darle a *Create Database*
3. Si todo ha ido correctamente, una vez finalizada la creación de la base de datos, deberá cargarse en el navegador la página *Aplicaciones*.

¹ Sólo se indican los campos obligatorios

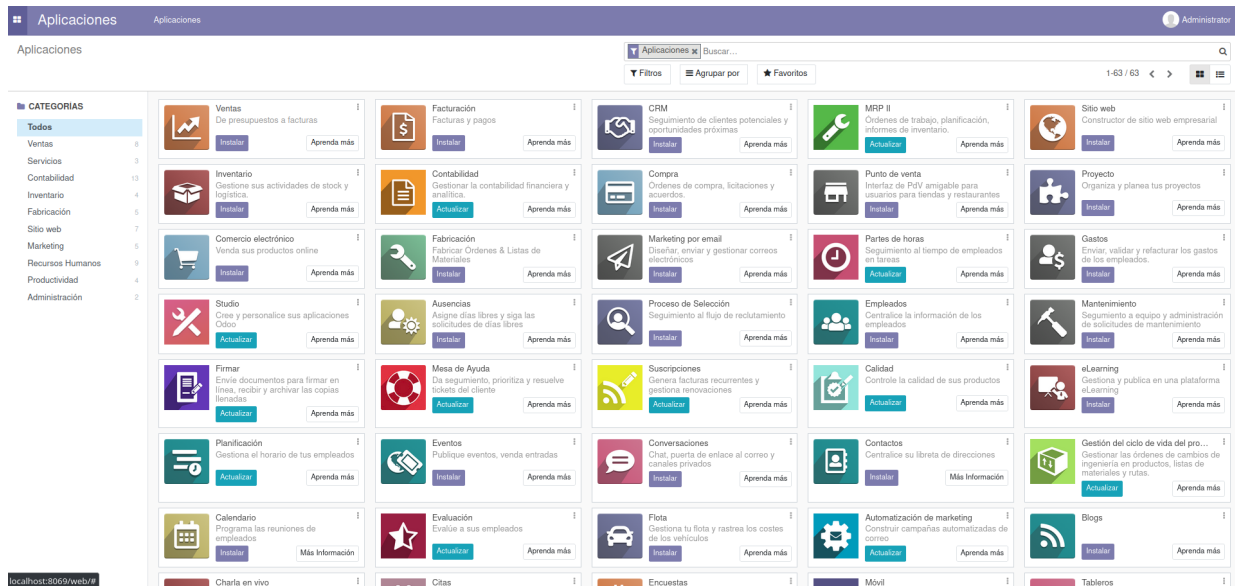


Figura 2. Página de aplicaciones

5.2 Modo debug

El modo *debug* es fundamental para el trabajo en desarrollo ya que permite la visualización de mucha información que de otra manera permanece oculta. Su activación puede realizarse, como *administrador*, desde:

 / Ajustes / Opciones generales / Herramientas de desarrollador




La opción con activos es una opción vitaminada del modo desarrollador, con más funcionalidades que la sin activos.

De todas maneras la opción más sencilla pasa por la instalación de una plugin (*Odoo Debug*) en el navegador que activa o desactiva el modo desarrollador de una manera muy sencilla.

Es posible encontrar este plugin en versiones para Firefox o Chrome.

5.3 Empresa

De manera similar a las bases de datos, Odoo requiere como mínimo una empresa para trabajar. Su configuración se realiza, como *administrador*, desde

 / Ajustes / Opciones generales / Compañía

5.4 Idiomas

Aunque en la página de configuración inicial se indica un idioma, es posible instalar todos aquellos que se consideren necesarios. Para ello hay que acceder, como *administrador*, a:

 / Ajustes / Opciones generales / Idiomas




En caso de querer administrar los idiomas (cambiar características, actualizarlo) es necesario acceder en modo desarrollador..

Servidor de correo saliente

Por último, es interesante configurar un servidor de correo que permita a Odoo mandar mensajes por correo electrónico, por ejemplo de bienvenida o de recuperación de contraseña.

Para ello hay que acceder a:

 / *Ajustes / Opciones generales / Conversaciones / Servidores de correo electrónico / Servidores de correo electrónico saliente*

Si no dispones de acceso a un servidor de correo, una opción gratuita (limitada en tamaño) interesante para poder hacer pruebas es utilizar a tu proveedor de correo como *gmail* o *outlook*.

La forma configurar la cuenta para que permita hacer las tareas de servidor de correo saliente varía mucho del proveedor e incluso de la versión de sus herramientas en el momento en el que se realiza, pero, por poner un ejemplo, podríamos decir que a grandes rasgos la configuración en *gmail* consiste en:

1. Ir a la configuración de la cuenta de *Google*
2. Pulsar en *Cuenta de Google*.
3. Posteriormente seleccionar *Inicio de Sesión en Google* y posteriormente selecciona *Verificación en dos pasos*. (la verificación en dos pasos es necesaria para poder utilizar *gmail* como servidor de correo saliente... además de que aporta más seguridad de acceso a tu cuenta)
4. Una vez confirmada la verificación en dos pasos, vuelve al paso 3.
5. Habrá aparecido una opción *Contraseña de aplicaciones*. Seleccionala.
6. En seleccionar aplicación elige otra y pon como nombre *Odoo-SGE*. El sistema te generará una clave.
7. Utiliza esa clave en la configuración de *Odoo*, junto con la dirección *smtp.gmail.com*, puerto 465 y seguridad *SSL/TLS*

6 AUTORES

A continuación ofrecemos en orden alfabético (por apellido) el listado de autores que han hecho aportaciones a este documento.

- Jose Castillo Aliaga
- Sergi García Barea
- Alfredo Oltra