El mantenimiento de una aplicación es muy importante ya que pueden darse ciertas situaciones en las que se necesiten hacer actualizaciones. Algunos motivos para realizar actualizaciones son:

1. Adaptar el software a un nuevo sistema operativo / versión.
2. Cubrir nuevas necesidades.

Una aplicación está finalizada cuando se distribuye al cliente y se instala en su dispositivo. La versión de una aplicación se refiere a su distribución sobre un estado concreto. Hay dos variantes para evolucionar la aplicación:

1. Generar nuevos paquetes autoinstalables que hagan la instalación de toda la app.
2. Realizar ampliaciones / sustituciones de las bibliotecas / paquetes que se han modificado.

Las fases de un ciclo de vida de un software son:

1. Análisis: Se definen y escogen los requisitos que determinan qué va a realizar el sistema. (Sin detalles tecnológicos)
2. Diseño: Se elaboran y toman las decisiones adecuadas para realizar el proyecto. Se define la estructura global y especifica qué debe hacer cada parte.
3. Desarrollo: Se implementa el código dependiendo de todo lo anterior.
4. Integración: Se recogen los módulos que se han implementado y se juntan para hacer la aplicación completa.
5. Pruebas: Se valida y verifica que funcionan correctamente los módulos por separado e integrados. Se revisa toda la información generada.
6. Finalización y transferencia: Empaquetamiento e instalación de aplicaciones. Tiene en cuenta la entrega y forma de instalación al cliente.

En la última fase se generan los archivos de instalación, está muy ligada a las anteriores ya que: A través del análisis se sabrá como es el producto, a partir del diseño se detalla más el producto y con el desarrollo, integración y pruebas se crea el producto.

Existen varios modelos a seguir para establecer estas fases del ciclo de vida: Modelo iterativo incremental, en cascada, en cascada iterativo, evolutivo incremental, en V y en espiral.

Dependiendo del autor, el mantenimiento puede ser una fase más después de la finalización, pero hay otros que lo incluyen en la finalización y transferencia o lo repiten todo en otro proyecto.

.

Una aplicación es el conjunto de componentes software que son necesarios para que funcione. Los componentes software son recursos que se han desarrollado para una determinada tarea y puede resolver procedimientos de forma única y en conjuntos de componentes. Estos se diferencian a nivel de desarrollo, estructura, tecnología, etc…

Además de estar compuesta por componentes software, la aplicación cuenta con:

1. Librerías: Son un conjunto de archivos / programas que se usan para desarrollar aplicaciones. Se caracterizan por disponer de funcionalidades concretas ya implementadas que facilitan el trabajo. Estas se encuentran incrustadas en la aplicación.
2. Entorno de trabajo: Un framework ofrece una arquitectura con varios componentes software de la cual se pueden desarrollar otros proyectos software. Puede estar compuesto de librerías, paquetes, otros software y lenguajes de programación. Es una aplicación semicompleta y de control invertido. En un framework se llama al código desarrollado y las bibliotecas son código llamado en el código desarrollado. Incorporan apoyo a software, facilidades, bibliotecas, lenguaje interpretado…
3. Empaquetado: Un paquete de software es un conjunto de aplicaciones que se distribuyen en una única unidad. Estas aplicaciones se necesitan para que el software funcione. El empaquetado de aplicaciones hace referencia a la distribución de aplicaciones en paquetes que contienen toda la tecnología y bibliotecas junto a la aplicación. Un gestor de paquetes se encarga de las dependencias de los archivos de una aplicación. Mantiene las versiones y gestiona las descargas. Ventajas del empaquetado:
4. Evitan problemas de dependencias al instalar o desinstalar.
5. Mejoran y facilitan el proceso de instalación.
6. Permiten una sencilla vinculación de las partes del programa.
7. Ofrecen la posibilidad de trasladar aplicaciones entre máquinas.
8. Ensamblaje: Es uno o varios archivos que contienen el código y archivos necesarios para una aplicación informática. Este paquete será reutilizable, versionable y autodescriptivo de una aplicación de Common Language Runtime en tiempo de ejecución. El CLR es un componente de la máquina virtual de la plataforma .NET, este es capaz de diferenciar el tiempo de compilación y de ejecución.

.

Existen muchas formas para empaquetar aplicaciones y se pueden clasificar en función de los sistemas para los que se han desarrollado y sus características: MSI para Windows, DEB/RPM para Linux, PKG para MAC OS y SysV para Solaris.

El paquete será libre o privativo: Si un paquete se puede usar en varios sistemas operativos debe ser software libre. Si se tratan de paquetes privativos: HP-UX, Distributor, SysV y .NET

También se pueden clasificar en función de si son binarios o se generan desde un código híbrido: deb/dpkg para Debian, rpm para RedHat, tgz para Linux y fink para MAC Os. Si no es híbrido: FreeBSD, MacPorts, pkgsrc.

Cada sistema operativo tiene sus propios tipos de empaquetamiento:

DEB: Extensión de paquetes de Debian y sus derivadas (Ubuntu). Se trata de dos archivos tar y qzip; uno con información de control y otro con información de datos.

RPM: Formato de archivo usado en Linux. Package Manager es una herramienta de administración de paquetes usado por SuSE, Mandriva o Fedora. Es multiplataforma.

MSI: Para la creación de estos se usa; Microsoft Visual Studio, Microsoft Installer Development Tools, MSI Factory, MSI Package Builder y 7-zip.

EXE: Extensión de archivos ejecutables; a diferencia del anterior que abre el instalador de Windows. Los .msi contienen los datos y archivos referentes a una instalación mientras los .exe no llaman al instalador de Windows.

JAR: Es el que se corresponde con Java (Conjunto de clases relacionadas entre sí). Estos definen una jerarquía de directorios que permiten la agrupación en clases. Es multiplataforma.

PKG / IPS: Usado en Solaris y Sun Microsystem. Se apoya en paquetes SVR4 de Solaris.

.

Los paquetes instalables ofrecen la forma de instalar y/o desinstalar aplicaciones. Se pueden parametrizar con opciones. Uno de estos paquetes no ofrece muchas opciones, solo mostrará la barra de progreso y la posibilidad de seleccionar alguna opción. Se encarga de ubicar todos los archivos necesarios para la aplicación. Para una aplicación es mejor usar paquetes desinstaladores ya que ejerce la tarea de forma directa.

Un sistema de gestión de paquetes es un elemento muy importante. Estos aportan un conjunto de herramientas que automatizan el proceso de gestión de paquetes software del sistema. Los sistemas administradores de paquetes son más comunes en Linux.

Existen los lenguajes interpretados y compilados; Los primeros cuentan con su propio sistema de gestión de paquetes (PHP, PEAR). Perl cuenta con CPAN (red importante de documentación y software.

Características de un SGP: Se encarga de su instalación, cada uno tiene su instalador, solo trabajarán con una aplicación, etc…

Funcionalidades de un SGP: Organización de paquetes, mantener usabilidad, resolver problemas entre módulos, CRUD de paquetes, etc…

Inconvenientes de un SGP: Necesidad de administrador de dependencias externo, no permite gestionar varias versiones de componentes, muchos paquetes no se conocen hasta que se abren para su instalación, etc…

Un SGP amplía las funcionalidades de un instalador, además permite trabajar con muchos paquetes a la vez.

Cuando se ha desarrollado la aplicación se decide cómo se va a empaquetar o crear el instalador. La mayoría de entornos incorporan herramientas para facilitar crear paquetes instalables. Si el entorno no incorpora estas herramientas se necesita un software específico para crear los paquetes. Herramientas:

InstallBuilder: Herramienta privativa que crea instaladores válidos en diferentes plataformas y sistemas operativos. Permite crear paquetes RPM y DEB.

InstallAnywhere: Herramienta privativa que permite crear instalables en varios sistemas.

IzPack: Herramienta gratuita que permite crear instaladores muy sencillos en la distribución de software de Java. Un solo instalador sirve para los demás sistemas.

En Windows existen Inno Setup, WiX e InstallShield.

Autopackage es un SGP que crea paquetes DEB y RPM para su instalación en Linux.

.

Hay una serie de parámetros que permiten modificar los SGP en función de sus necesidades. Estas opciones se muestran en tiempo de ejecución del instalable y están pensadas para usarse; en cambio, hay otras que sí serán escogidas por el desarrollador: Icono del paquete, diálogo a establecer con el usuario, logos, barra de estado e imágenes.