WIKIPEDIA

Software deployment

Software deployment is all of the activities that make a software system available for use.[1]

The general deployment process consists of several interrelated activities with possible transitions between them. These activities can occur at the producer side or at the consumer side or both. Because every software system is unique, the precise processes or procedures within each activity can hardly be defined. Therefore, "deployment" should be interpreted as a *general process* that has to be customized according to specific requirements or characteristics.

Contents

History

When computers were extremely large, expensive and bulky (mainframes and minicomputers), software was often bundled together with the hardware by manufacturers. If business software needed to be installed on an existing computer, this might require an expensive, time-consuming visit by a systems architect or a consultant. For complex, on-premises installation of enterprise software today, this can still sometimes be the case.

However, with the development of mass market software for the new age of microcomputers in the 1980s came new forms of software distribution – first cartridges, then cassette tapes, then floppy disks, then (in the 1990s and later) optical media, the internet and flash drives. This meant that software deployment could be left to the customer. However, it was also increasingly recognised over time that configurability of the software by the customer was important, and that this should ideally have a user-friendly interface (rather than, for example, requiring the customer to edit registry entries on Windows).

In pre-internet software deployments, deployments (and their closely related cousin, new software releases) were of necessity expensive, infrequent, bulky affairs. It is arguable therefore that the spread of the internet made end-to-end agile software development possible. Indeed, the advent of cloud computing and software as a service meant that software could be deployed to a large number of customers in minutes, over the internet. This also meant that typically, deployment schedules were now determined by the software supplier, not by the customers. Such flexibility led to the rise of continuous delivery as a viable option, especially for less risky web applications.

Deployment activities

Release

The release activity follows from the completed development process, and is sometimes classified as part of the development process rather than deployment process. It includes all the operations to prepare a system for assembly and transfer to the computer system(s) on which it will be run in production. Therefore, it sometimes involves determining the resources required for the system to operate with tolerable performance and planning and/or documenting subsequent activities of the deployment process.

Installation and activation

For simple systems, installation involves establishing some form of command, shortcut, script or service for executing the software (manually or automatically). For complex systems it may involve configuration of the system – possibly by asking the end-user questions about its intended use, or directly asking them how they would like it to be configured – and/or making all the required subsystems ready to use. Activation is the activity of starting up the executable component of software for the first time (not to be confused with the common use of the term *activation* concerning a software license, which is a function of Digital Rights Management systems.)

In larger software deployments on servers, the main copy of the software to be used by users - "production" - might be installed on a production server in a production environment. Other versions of the deployed software may be installed in a test environment, development environment and disaster recovery environment.

In complex continuous delivery environments and/or software as a service systems, differently-configured versions of the system might even exist simultaneously in the production environment for different internal or external customers (this is known as a *multi-tenant architecture*), or even be gradually rolled out in parallel to different groups of customers, with the possibility of cancelling one or more of the parallel deployments. For example, Twitter is known to use the latter approach for A/B testing of new features and user interface changes. A "hidden live" group can also be created within a production environment, consisting of servers that are not yet connected to the production load balancer, for the purposes of blue-green deployment.

Deactivation

Deactivation is the inverse of activation, and refers to shutting down any already-executing components of a system. Deactivation is often required to perform other deployment activities, e.g., a software system may need to be deactivated before an update can be performed. The practice of removing infrequently used or obsolete systems from service is often referred to as application retirement or application decommissioning.

Uninstallation

Uninstallation is the inverse of installation. It is the removal of a system that is no longer required. It may also involve some reconfiguration of other software systems in order to remove the uninstalled system’s dependencies.

Update

The update process replaces an earlier version of all or part of a software system with a newer release. It commonly consists of deactivation followed by installation. On some systems, such as on Linux when using the system's package manager, the old version of a software application is typically also uninstalled as an automatic part of the process. (This is because Linux package managers do not typically support installing multiple versions of a software application at the same time, unless the software package has been specifically designed to work around this limitation.)

Built-in update

Mechanisms for installing updates are built into some software systems (or, in the case of some operating systems such as Linux, Android and iOS, into the operating system itself). Automation of these update processes ranges from fully automatic to user initiated and controlled. Norton Internet Security is an example of a system with a semi-automatic method for retrieving and installing updates to both the antivirus definitions and other components of the system. Other software products provide query mechanisms for determining when updates are available.

Version tracking

Version tracking systems help the user find and install updates to software systems. For example: Software Catalog stores version and other information for each software package installed on a local system. One click of a button launches a browser window to the upgrade web page for the application, including auto-filling of the user name and password for sites that require a login. On Linux, Android and iOS this process is even easier because a standardised process for version tracking (for software packages installed in the officially supported way) is built into the operating system, so no separate login, download and execute steps are required – so the process can be configured to be fully automated. Some third-party software also supports automated version tracking and upgrading for certain Windows software packages.

Adaptation

The adaptation activity is also a process to modify a software system that has been previously installed. It differs from updating in that adaptations are initiated by local events such as changing the environment of customer site, while updating is a consequence of a new release being made available. Adaptation may require specialist technical skills such as computer programming, in certain complex cases.

Deployment roles

The complexity and variability of software products has fostered the emergence of specialized roles for coordinating and engineering the deployment process. For desktop systems, end-users frequently also become the "software deployers" when they install a software package on their machine. The deployment of enterprise software involves many more roles, and those roles typically change as the application progresses from test (pre-production) to production environments. Typical roles involved in software deployments for enterprise applications may include:

- in pre-production environments:
  - application developers: see Software development process
  - build-and-release engineers: see Release engineering
  - release managers: see Release management
  - deployment coordinators: see DevOps
- in production environments:
  - system administrator
  - database administrator
  - release coordinators: see DevOps
  - operations project managers: see Information Technology Infrastructure Library

See also

- Application lifecycle management
- Product lifecycle management
- Systems management
- System deployment
- Software release
- Definitive Media Library
- Readme
- Release management
- 1E

Deployment tools

- Ansible
- SaltStack
- OSGi
- JNLP
- RPM
- Apt
- Capistrano
- Lansweeper
- XebiaLabs
- ICEFLO
- Electric Cloud
- BuildMaster
- Serena Deployment Automation
- Chef

References

1. Roger S. Pressman Software engineering: a practitioner's approach (eighth edition)

External links

- Standardization efforts
  - Solution Installation Schema Submission request to W3C (http://www.w3.org/Submission/2004/04/)
  - OASIS Solution Deployment Descriptor TC (http://www.oasis-open.org/committees/sdd/charter.php)

- OMG Specification for Deployment and Configuration of Component-based Distributed Applications (http://www.info.fundp.ac.be/~ven/CIS/OMG/Deployment%20and%20Configuration%20of%20Component-based%20Distributed%20Applications.pdf) (OMG D&C)
- JSR 88: Java EE Application Deployment (http://jcp.org/en/jsr/detail?id=88)

- Articles
  - The Future of Software Delivery (https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-tfosd&S_TACT=105AGY59&S_CMP=WIKIWP&ca=dtl-2108wp5) - free developerWorks whitepaper
  - Carzaniga, Antonio; Fuggetta, Alfonso; Hall, Richard S.; Van Der Hoek, André; Heimbigner, Dennis; Wolf, Alexander L. (April 1998). "A Characterization Framework for Software Deployment Technologies – Technical Report CU-CS-857-98" (http://www.cs.colorado.edu/department/publications/reports/docs/CU-CS-857-98.pdf) (PDF). Boulder, CO: Department of Computer Science, University of Colorado Boulder.
- Resources
  - Visual Studio Release Management (https://www.visualstudio.com/en-us/features/release-management-vs.aspx/)

Retrieved from "https://en.wikipedia.org/w/index.php?title=Software_deployment&oldid=876396933"