ISO/IEC FDIS 20582~~:2024(en)~~

ISO/IEC JTC 1/SC 7~~/ N9694~~

~~Date: 2025-01-03~~

Secretariat: BIS

Date: 2025-02-18

Software and systems engineering — Capabilities of build and deployment tools

FDIS stage

ISO ~~Copyright Office~~copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: + 41 22 749 01 11
~~Email:~~ E-mail: copyright@iso.org
Website: www.iso.org

Published in Switzerland~~.~~

Contents

~~1~~

~~2~~

~~3~~

~~3.1~~

~~3.1.1~~

~~3.1.2~~

~~3.1.3~~

~~3.1.4~~

~~3.1.5~~

~~3.1.6~~

~~3.2~~

~~4~~

~~4.1~~

~~4.2~~

~~6.5.3~~ Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Introduction

Build and deployment tools and methods have long been central to software engineering, and the importance of their role continues to increase. Traditionally, build and deployment tools were simple mechanisms used to create executable code from the source code written in a programming language, typically comprising a compiler and a linker bundled with the operating system. Over time, these mechanisms have become much more complex, where application software is built using the source code and libraries distributed across multiple servers. Furthermore, the testing and operation of a system often requires deployment to a large number of servers. Increasing sophistication in software development environments is another factor in the increasing the importance of advanced build and deployment tools and methods.

Many modern build and deployment tools are now available on the market, independent of the vendor providing the operating system. These tools can be configured as independent automation on top of compilers and linkers bundled with individual operating systems, as well as repositories that manage assets. Therefore, there are multiple options available, and it is essential to choose the appropriate one for each different software engineering context. This ~~standard~~document provides guidance in the selection of commercially available tools from different vendors as well as tools provided as open source.

This document treats tasks and tools related to change management and configuration management as out of scope. Build and deployment processes are typically initiated by modifications to the source code, indicating they are closely related to change management and configuration management. However, while change management and configuration management primarily deal with source code, build and deployment involves the transformation of source code into sophisticated operational representations and structures. Therefore, change management and configuration management are excluded to enable focus on the single build and deployment concern.

From a user's perspective, it is important to choose the right tools from the variety of available build and deployment tools, especially in large organizations. It is imperative that tool evaluation and selection be done in accordance with fair and public standards. To this end, ISO/IEC 20741 was published in 2017 as a guideline to evaluate and select software engineering tools. However, ISO/IEC 20741 does not identify the standard capabilities specific to build and deployment tools since it is generalized without regard to a specific tool ~~categories~~category.

This document defines the requirements for capabilities of build and deployment tools and is intended to be used in conjunction with ISO/IEC 20741 to select the appropriate tool (see Annex A~~Annex A).~~). The document provides a list of capabilities in build and deployment tools. The capabilities described in this document are gathered from existing tools (see Annex B~~Annex B).~~).

Build, deploy, and CI/CD tools are separate and independent tools, which ~~could~~can logically be treated in separate standards. However, in most software engineering environments, they are applied in highly integrated, interdependent, and continuous workflows. Evaluation and selection are best conducted from this integrated viewpoint and this document functions as a single standard for this integrated collection tools.

Software and systems engineering — Capabilities of build and deployment tools

1 Scope

This document defines the requirements for capabilities of build and deployment tools to support and automate building, packaging, and deploying work.

This document is intended for use in evaluating and selecting build and deployment tools according to the procedures defined in ISO/IEC 20741.

This document is applicable to different development methodologies and approaches (e.g. Waterfall, Agile, or DevOps~~)~~).

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminology databases for use in standardization at the following addresses:

— ~~—~~ISO Online browsing platform: available at https://www.iso.org/obp

— ~~—~~IEC Electropedia: available at https://www.electropedia.org/

3.1 ~~3.1~~

build

process of generating an executable and testable system from source versions or baselines

[SOURCE: ISO/IEC/IEEE 32675:2022]

3.2 ~~3.2~~

continuous integration

CI

technique that continually merges artifacts, including source code updates from all developers on a team, into a shared mainline to build and test the developed system

[SOURCE: ISO/IEC/IEEE 32675:2022, modified — The abbreviated term "CI" has been added.]

3.3 ~~3.3~~

continuous delivery

CD

software engineering practices that allow for frequent releases of new systems (including software) to staging or various test environments through the use of automated tools

[SOURCE: ISO/IEC/IEEE 32675:2022, modified — The abbreviated term "CD" has been added.]

3.4 ~~3.4~~
continuous deployment
CD
automated process of deploying changes to production by verifying intended features and validations to reduce risk

[SOURCE: ISO/IEC/IEEE 32675:2022, modified — The abbreviated term "CD" has been added.]

3.5 ~~3.5~~
deployment
stages of a life cycle in which a system is put into operation and transition issues are resolved

[SOURCE: ISO/IEC/IEEE 32675:2022]

3.6 ~~3.6~~
package
combine related components into a single, deployable item

Note 1 to entry: A package is used to group elements and provides a namespace for the grouped elements. See ISO/IEC 19505-2.

[SOURCE: ISO/IEC/IEEE 32675:2022, modified — Note 1 to entry has been added.]

4 Object model for build and deployment tools

4.1 Overview

Generally, tools reduce manual effort by automatically generating entities to be created during the software development process.

The tools, the entities, and the process are expressed as three views:

— process view;

— entity view;

— tool view.

The model of build and deployment can be expressed as a package that represents three views as shown in Figure 1~~Figure 1.~~ Here, the use case model corresponds to the process view.

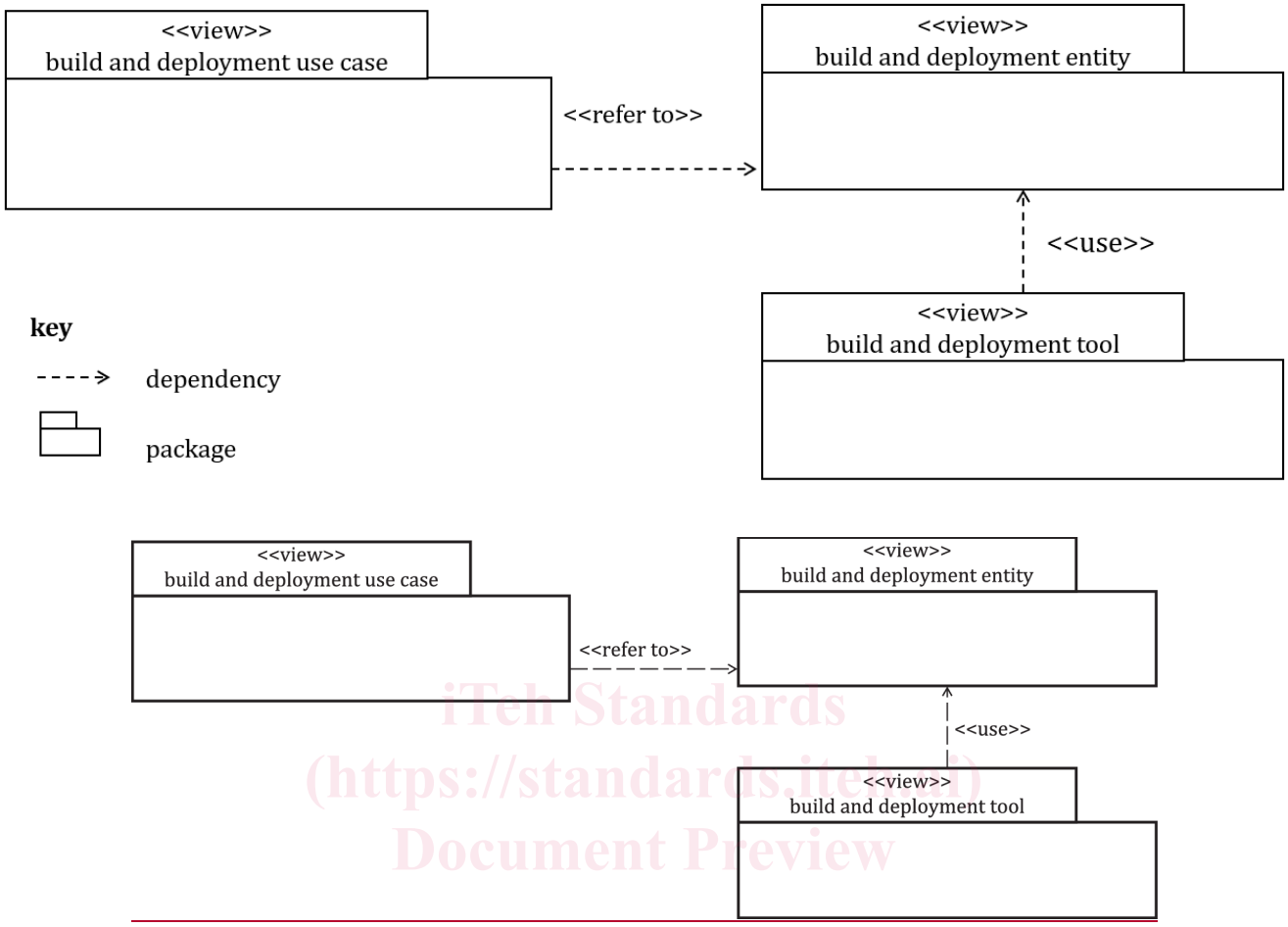


Figure 1 — Object model for build and deployment

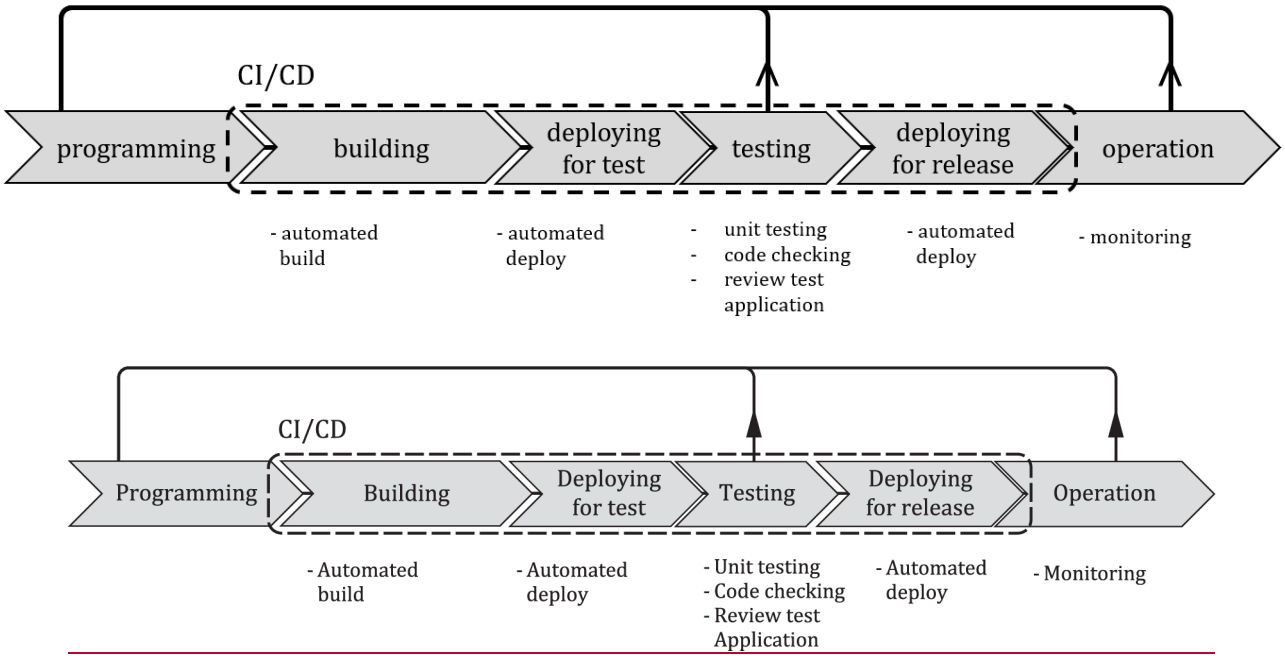
The rest of this clause is based on Figure 1~~Figure 1,~~ with 4.2~~4.2~~ defining use cases and 4.3~~4.3~~ defining the structure of entities related to build and deployment. Clause 5~~Clause 5~~ defines entities in more detail, based on which Clause 6~~Clause 6~~ specifies tool capabilities.

The object model diagrams, Figure 1~~Figure 1, Figure 3~~, Figure 3 to Figure 8~~Figure 8,~~ and Figure 10~~Figure 10~~ to Figure 12~~Figure 12~~ are constructed using Unified Modelling Language (UML) 2 (ISO/IEC 19505-2).

4.2 Use case

4.2.1 Build, deployment and CI/CD in system lifecycle

Figure 2~~Figure 2~~ illustrates the scope of build and deployment in the latter part of the system development process. Building involves transforming the source code created during coding into an executable package. Deployment involves setting the package, created during the build process, to operate in either the test environment or the production environment.



Key

Figure 2 — Build, deployment and CI/CD in the system lifecycle

The details of each task described in Figure 2~~Figure 2~~ are shown below. In CI/CD, tasks b), c), d), and e) are continuously and repeatedly performed by automation.

a) ~~a)~~ Programming

Developers create or modify source code or libraries according to application specifications or change orders. To achieve this, both change management tools and configuration management tools are typically used. Developers obtain the source code as a personal version from the configuration management tools and apply changes to the original version.

This task serves as a trigger for the build process but is outside the scope of this document.

b) ~~b)~~ Building (include packaging)

The source code is converted into an executable package.

c) ~~c)~~ Deploying for Test

The servers and the package are configured to run in the test environment.

d) ~~d)~~ Testing

Dynamic testing is performed on the package deployed in the test environment.

e) ~~e)~~ Deploying for release

The servers and the package are configured to run in the production environment.

f) ~~f)~~ Operation

The package is operated in the production environment.

4.2.2 Use cases of the build and deployment tool

The build and deployment tool is used in the following use cases (building, deployment, and CI/CD). The following actors appear in the use cases, as shown in Figure 3~~Figure 3~~.

a) ~~a)~~ Developer

Members of the project who write the source code, build the source code, and create the package, by using the build and deployment tool.

b) ~~b)~~ Deployer

Members of the project who deploy the package by using Build and deployment tool to each environment.

c) ~~c)~~ Repository

Storages for resources such as source code and manage versions of resources.

d) ~~d)~~ Environment

Servers providing the hardware and software necessary for executing the package.

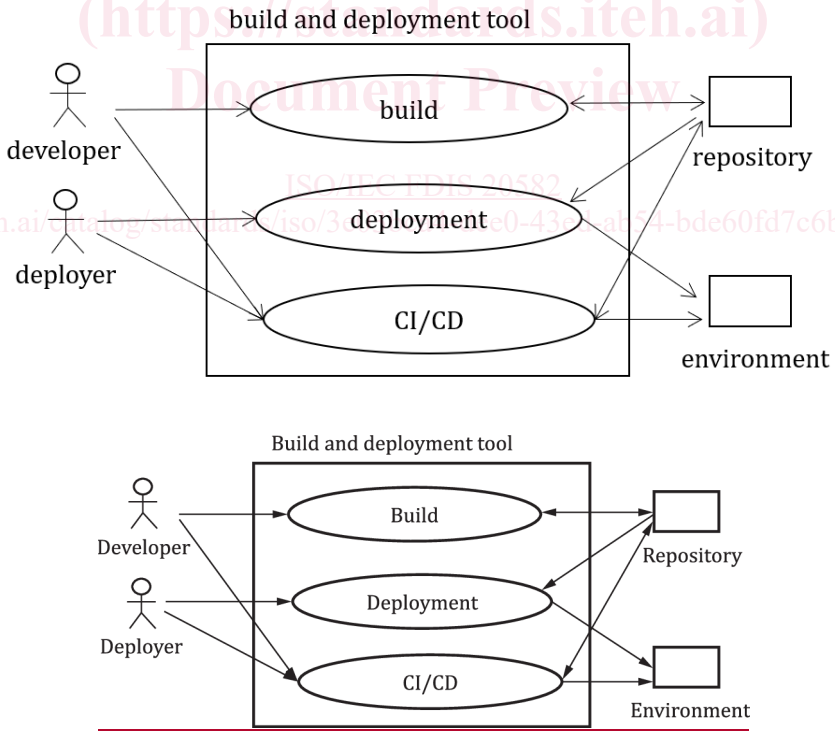


Figure 3 — Use case diagram of the build and deployment tool

4.2.3 Use case scenario

4.2.3.1 Use case scenario of building

The use case scenario of building is defined by using the activity diagram shown in Figure 4~~Figure 4~~.

The information items described in the swim lane "Repository" in Figure 4~~Figure 4~~ are the following three types:

— ~~—~~SourceRepository stores source code;

— ~~—~~ExternalLibraryRepository stores libraries created outside the project applying the build and deployment tools;

— ~~—~~PackageRepository stores libraries, executable packages, and data created in the project.

Building is executed in the following procedure according to Figure 4~~Figure 4.~~.

a) ~~a)~~ The developer requests the build and deployment tool to build.

b) ~~b)~~ The build and deployment tool gets the source code from SourceRepository. At the same time, ExternalLibrary having a calling relationship is copied from ExternalLibraryRepository, and InternalLibrary is copied from PackageRepository.

c) ~~c)~~ The build and deployment tool invokes the translator to generate the package from the source code. Then the build and deployment tool register the package in PackageRepository.

d) ~~d)~~ The build and deployment tool notifies the developer of the results of c).