Lifecycles of SUSE Linux Components

WHAT?

This article outlines how every SUSE Linux 16 software package is assigned one of three specific update policies: stable, balanced or agile.

WHY?

To understand how package updates may affect system stability and compatibility.

EFFORT

No configuration is required; only awareness and planning are needed.

GOAL

You will have an overview of how SUSE categorizes packages.

Publication Date: 07 Aug 2026

Contents

1 Types of package lifecycles

On SUSE Linux 16, component packages are sorted into different lifecycle categories. This section describes the criteria for such a sorting.

When categorizing a package, the impact of its changes on the system is considered. To estimate this impact, the interfaces of the component must first be identified. For example, in the case of a shared library, changes to its API or ABI may disrupt the system. For a compiler or interpreter, disruptive changes may involve supported languages, command-line options, or the performance of the compiled code. By contrast, a minor backward compatible change may have little to no impact.

In general, a package falls exactly into one of the categories: **stable**, **balanced**, or **agile**. However, some technologies may have packages sorted into several categories, for example, Python. The following sections describe the package categories in detail.

1.1 Stable

Packages that are marked as stable (also called conservative) are those that do not deliver a disruptive change while a customer is on any of the 16 minor versions. During the upgrade to another minor version, the package version may change but the newer version does **not** introduce incompatible behavior. Customers expect to have LTS on these packages.

The packages belonging to this category can change, but the following criteria apply:

- changes in functionality are backward compatible: functionality can be added, but not removed
- changes to interfaces are backward compatible
- the default behavior of applications does not change unexpectedly

Under exceptional circumstances like serious security issues, the package can be updated even at the cost of bringing disruptive changes. Alternatively, if a new version of a package contains disruptive changes, this version can be delivered as an alternative to the previous one.

A typical example of a stable package is `util-linux`. Another is `glibc`, which remains backward compatible except for symbols deprecated upstream.

1.2 Balanced

Packages categorized as balanced are changing (driven by upstream evolution and customer demands), but should not cause disruptive changes within one minor version. A few incompatible changes are possible, but always documented in the release notes for that particular minor release.

Customers expect a moderate number of changes during the upgrade from a minor release. However, the transition must be smooth, either by getting back to the original behavior or by providing the older version in parallel with the new one.

When change is being introduced, it can be in one of the following ways:

- A single version is provided with the minor release. The new version replaces the previous one while allowing for smooth transitions between the versions. New versions are released only with a new minor release.
- Two versions are provided simultaneously. The new version is introduced in addition to the existing version, which then becomes obsolete with the next minor release.

Such versions are supported at least until the end of the LTS of the minor release that introduced them.

To help with incorporating changes in a conservative environment, the tick-tock model can be used. For example, we could mark even-numbered minor releases as *tick* releases and odd-numbered minor releases as *tock*. These *tock* releases could still introduce version updates in packages with a balanced lifecycle, but these updates remain fully backward compatible in all relevant aspects.

For ISVs, a stable runtime environment is critical. Therefore, not to break third-party applications, in the case of shared libraries where SUSE provides the corresponding `-devel` package, the older `.so` version is not deprecated immediately. For example, for a package called `foo`, there are packages: `libfoo-0_1`, `foo-devel` and `foo-utils`. If the package is updated and the shared library version changes to `libfoo-0_2`, `libfoo-0_1` is not removed.

Typical examples of balanced packages include the following:

- `systemd`: changes should be backward compatible; incompatible changes are documented in the release notes
- kernel: the kernel is updated with each minor release
- virtualization components: they are updated with each minor release
- MariaDB: a new minor version with each minor release

- PostgreSQL: upstream versions are released on a roughly annual basis, so the new version is introduced either with a minor release or as part of a maintenance update for an older minor release
- Python: a version supported for a longer period of time with a set of modules on top

1.3 Agile

Packages categorized as agile are up to date with upstream even though they may bring incompatible changes to the system.

Updates to these packages are done in two possible ways:

- With the release of a new package version, one or more older versions are supported for customers that cannot switch easily to the new version. There is a sliding window in which different versions are supported concurrently (for example, for version N and version N - 1). These concurrent versions are supported for a certain period of time that can differ from the lifetime of the minor release.
- The package is just updated without support for the older version.

All new package versions are released simultaneously to all minor releases under general support and generally also in LTS.

The following packages are categorized as agile:

- Go and Rust: a new version is released roughly every 6 months with a sliding window of 2 months.
- GCC: a new version of the compiler is released with each minor release. Those introduced in odd minor versions are not used by default and have a shorter support period. See *Section 17.2, "Compiler for user space applications and libraries"* for details. However, the libraries `libgcc` and `libstdc++` are categorized as stable.
- CLI and SDK for Public Cloud: a new version every quarter, and the new version replaces the previous one.

- Python interpreter, library and pip: a new version is released annually.
- Data files for time zones: a new version is released when a new set of definitions becomes available.

2 Ansible update strategy

Ansible is an open source IT automation engine that simplifies the process of provisioning, configuration management, application deployment, and orchestration by using a human-readable, agentless language.

The Ansible stack follows the agile lifecycle category. Ansible packages are updated as described below:

TABLE 1: ANSIBLE COMPONENTS RELEASE FREQUENCY

Component	Packages	Release frequency
Ansible interpreter	`ansible`, `ansible-core`	Twice a yea (typically May and November)
Linux System Roles	`ansible-linux-system-roles`	Frequently
SAP roles & playbooks	`ansible-sap-install`, `ansible-sap-infrastructure`, `ansible-sap-operations`, `ansible-sap-playbooks`	Irregularly
Unsupported SAP roles	`ansible-sap-launchpad`	Irregularly

On SUSE Linux Enterprise Server, the Ansible interpreter and Linux system roles are supported. On SLES for SAP, also SAP roles & playbooks are supported. The support covers only the roles and playbooks that are delivered by SUSE.

3 Cockpit update strategy

Cockpit is a Web-based graphical interface that enables you to manage most administration tasks from one place. This section explains how often Cockpit is updated.

Cockpit is updated and supported according to the following rules:

- Each SUSE Linux minor release comes with the latest Cockpit version.
- Support for this version continues throughout the entire lifecycle of the minor release, with any necessary fixes provided through backports.
- To make use of upstream improvements, SUSE may upgrade to a newer version when reasonable (for example, due to bug fixes, security updates, or usability improvements), especially in LTS situations.

4 Update strategy of desktop components

In general, desktop components follow the balanced lifecycle of their packages.

The SLE 16 desktop environment provides the following components:

GNOME desktop

The minimum version is *48.0*. The exact version must be determined before the end of beta releases.

GStreamer, PipeWire and Flatpak

These components are updated to the latest stable branch version at the last beta release.

Firefox

The minimum version is *140.3*. The update strategy follows the ESR update lifecycle.

WebKit

The minimum version is *2.46*. WebKit has a periodic update cycle determined by critical CVEs. The component is updated but not backported.

BRLTTY

The accessibility tools are at the latest upstream stable versions in the 16.0. Only bug fixes will happen in this minor release.

QT

The Qt 6 is delivered. The initial version is Qt 6.9.

KDE

Not a standard SUSE Linux delivery; available only in PackageHub 16.

5 Go update strategy

Go is an open source, statically typed, compiled programming language designed for building reliable and efficient software. Go follows the agile lifecycle category.

5.1 Lifecycle and support policy

On SUSE Linux 16, Go and its components are updated and supported according to the following rules:

- **Release cycle:** Releases are kept in sync with upstream, which nominally occur every 6 months in February and August.
- **Version support:** SUSE maintains versions N and N-1, mirroring the upstream support policy.
- **Minor releases:** Go minor releases are continuously made available shortly after the upstream release.
- **Parallel installation:** Different versions are parallel installable and undergo automated testing. The `go` metapackage always points to the latest `go1.x` package.
- **Support phases:** Go is supported across the General Support and LTSS phases for all relevant SUSE Linux minor releases.

5.2 Building Go applications

When building applications, developers have two options regarding versions:

- To build with the latest recommended version, Go applications may depend on `BuildRequires: golang(API) >= go1.23`.
- To control the timing of build changes, applications may pin to a specific `go1.x` package using `BuildRequires: golang(API) == go1.23`.

In both cases, you must use the major version of `go1.x` specified in the Go application's `go.mod` file, as the Go compiler now requires this as a minimum version.

5.3 FIPS compliance

Alongside standard `go1.xx` releases, SUSE provides a parallel `go1.xx-openssl` variant. This version is specifically patched to utilize OpenSSL for FIPS-compliant cryptographic operations.

Starting with `go1.24`, Go upstream introduced an integrated FIPS 140-3 cryptographic implementation. As this implementation achieves certification, the parallel `go1.xx-openssl` variant will be deprecated in favor of the upstream-supported FIPS mechanism. SUSE packaging will follow the upstream deprecation schedule and support lifecycle.

6 Java update strategy

On SUSE Linux 16, Java is delivered and supported in the following versions:

- *Java 17*: supported until December 2027
- *Java 21*: supported until October 2031
- *Java 25*: support date to be confirmed after this version is released, expected support date is October 2033

For the available versions, the following criteria apply:

- Concurrent installation of different versions is supported
- The Java versions are maintained independently of the SUSE Linux minor version release cycle
- Each version is supported throughout the general support and LTS phase
- Java packages can be updated in an agile manner

7 MariaDB update strategy

MariaDB is a high-performance, open source relational database management system (RDBMS) that was created as a community-developed fork of MySQL to ensure the software remains free and accessible under the GPL license.

MariaDB follows the balanced lifecycle—only LTS MariaDB releases are supported.

Each SUSE Linux minor version gets a new version of MariaDB. This version is supported for the General Support or LTS period. The starting version of MariaDB is 11.4.5.

Minor version updates, aligned with upstream releases, are provided exclusively for the `mariadb` and `mariadb-connector-c` packages.

8 `node.js` update strategy

`Node.js` is an open source, cross-platform JavaScript runtime environment that allows developers to build and run server-side applications outside of a web browser. It is built on Google Chrome's V8 engine and uses an event-driven, non-blocking I/O model, making it highly efficient for creating scalable network applications.

On SUSE Linux 16, `node.js` is updated and supported according to the following rules:

- The `node.js` version is kept in sync with the upstream LTS version.
- Each SUSE Linux 16 minor release has the latest Active LTS version (https://nodejs.org/en/about/previous-releases), starting with `nodejs 22`, and comes with a default version of `node.js`.
- All `node.js` minor releases are available generally after security fixes or on request.
- Different `node.js` versions can be installed in parallel.
- Versions in a specific SUSE Linux minor release are supported until the end of LTS.
- To always use the most recent recommended version available in the repository, applications should use `BuildRequires: nodejs >= 22`.

9 Perl update strategy

Perl is a high-level, general-purpose programming language originally developed for text manipulation that has since evolved into a versatile tool used for system administration, Web development, and network programming.

The Perl update strategy is considered stable and no new version is expected to be released.

10 PHP update strategy

PHP is an open source server-side scripting language designed specifically for Web development. It is used to create dynamic, interactive content that interacts with databases.

PHP packages follow the balanced lifecycle. The starting version is 8.4, and a new PHP version is shipped with each minor release of SUSE Linux 16. Patch-level version updates are delivered throughout the support period of each SUSE Linux minor release.

11 PostgreSQL

PostgreSQL is an open-source object-relational database management system (ORDBMS) known for its reliability and extensive support for advanced data types and complex queries.

On SUSE Linux 16, all PostgreSQL versions maintained by the upstream community are supported. When a new version is released by upstream, it is added to each minor SUSE Linux version under general or LTS support.

Once a PostgreSQL version reaches its end of life, it remains accessible within the repositories but will no longer receive security patches or bug fixes.

12 Python update strategy

For SUSE Linux Enterprise 16.0, the System Python is Python 3.13.

12.1 Python interpreter lifecycles

SUSE Linux Enterprise 16.0 includes several Python interpreter versions with different support lifecycles:

System Python interpreter

System Python is the Python version for which SUSE provides a large set of Python modules needed for SUSE Linux Enterprise 16.0 applications and solutions.

The System Python version and its modules remain stable and are supported throughout the SUSE Linux Enterprise 16.0 lifecycle.

Support covers both the General Support and Long Term Support (LTS) periods of SLES 16.0 and applies for each future SUSE Linux Enterprise 16.x minor release.

Developer Python interpreter

Developer Python provides a more recent Python version. Unlike System Python, it is intended to be used together with Python wheels provided by PyPI, which are installed and maintained via `pip` and `pipx`.

It is supported during the General Support period of SLES 16.0 but excludes LTS.

Legacy Python interpreter

In future SUSE Linux Enterprise 16.x minor releases, System Python will be updated to a newer version. At that time, the previous System Python version and all its modules will be provided as legacy Python.

12.2 Python release cycle

SUSE Linux Enterprise 16.0 follows a defined release and support schedule for Python interpreter versions.

In SUSE Linux Enterprise 16.0, the System Python is Python 3.13. The current plan (subject to change) is to provide newer Python versions in future SUSE Linux Enterprise 16.x releases.

A Developer Python version will be released with each minor release and will be supported only for the general support period of that release (two years).

13 Ruby delivery strategy

The Ruby stack is not delivered as a standard component of SUSE Linux 16.

However, specific components that depend on Ruby include it as a supported dependency:

- **Hawk (High Availability Web Konsole):** As a component of SUSE Linux Enterprise High Availability built on the Ruby on Rails framework, Hawk inherently requires a Ruby runtime environment to function.
- **RMT (Repository Mirroring Tool):** RMT requires Ruby and is provided as a supported exception.

14 Rust update strategy

Rust is a multi-paradigm, general-purpose programming language designed for performance and safety, especially safe concurrency. Rust follows the agile lifecycle category.

On SUSE Linux 16, Rust and its components are updated and supported according to the following rules:

- **Release cycle:** Releases are kept in sync with upstream updates, which occur every six weeks.
- **Version support:** SUSE maintains versions N and N-1.
- **Upgrade window:** There is a 6-week window provided to perform upgrades and find any potential issues.
- **Parallel installation:** Parallel installable versions are available and undergo automated testing.
- **Support phases:** Rust is supported across the General Support and LTSS phases.

15 Time zone update strategy

Data files for time zones are categorized under the agile lifecycle.

Within the SUSE Linux 16 code stream, the `timezone` is updated and supported according to the following rules:

- **Release syncing:** When a new set of time zone definitions becomes available, the package needs to be updated across all supported SUSE Linux systems.
- **Support for older versions:** As is standard for certain agile packages, the package is simply updated to the newer version without providing ongoing support for the older version.

This update model maintains the exact same approach that is taken for SLE 15.

16 Update strategy of security components

This document defines the update policy and lifecycle strategy for core security components within the SUSE Linux 16 code stream.

16.1 Stable security components

The following security components follow the stable life cycle and remains the same within the same minor version. New versions are typically introduced only to the next minor release:

- OpenSSL: The version is fixed for the purpose of FIPS 140-3 certification within the minor release.
- GNUTLS, Nettle: The version is fixed for the FIPS 140-3 certification.
- `libgcrypt`: The version is fixed for the FIPS 140-3 certification.
- OpenSSH
- `liboqs`/`oqs-provider`
- GnuPG
- krb5
- smartcards and u2f tokens
- IMA tools
- `openCryptoki`
- rage
- rsign2
- PAM stack and polkit

16.2 Balanced security components

The following security components follow the balanced life cycle and can be updated with a maintenance update or get version update with releasing a new minor release:

- firewalld
- SELinux: Policies are updated with monthly snapshots.
- Network Security Services (`mozilla-nss`): The version is fixed for the FIPS 140-3. However, updates of the package matches the Firefox ESR updates
- strongSwan: new versions are typically introduced with new minor release.

- Stunnel
- TPM libraries and tools

16.3 Agile security components

The following security components follow the agile life cycle and are kept up-to-date with upstream:

- Cosign and Rekor
- ClamAV
- `scap-security-guide`
- `ca-certificates-mozilla`: Updated when new NSS certificates bundle is released.

17 Update strategy of toolchain components

The toolchain components include the following tools: the GNU C library, the GCC and G++ compilers, `binutils`, GDB and LLVM. Each of the component update strategies is described in corresponding sections.

17.1 GNU C library (`glibc`)

The initial `glibc` version is *2.40*. The package is updated with each minor SUSE Linux release if there are reasons for changes to the package (for example feature requests, performance tuning and so on).

Package updates provide backward compatibility for dynamic linking, allowing programs built on previous SUSE Linux 16 releases to run. On the contrary, symbols deprecated in the upstream `glibc` version will not be declared for the compiler and not available for link editing (static linking).

17.2 Compiler for user space applications and libraries

Developers of user-space applications can use the supported GNU Compiler Collection (GCC) C and C++ built-in compilers. Compilers for other languages, cross-compilers and accelerator offloading compilers are not available on SUSE Linux from standard repositories, but developers can install them from Package Hub with community support.

The initial major version in SUSE Linux 16.0 and 16.1 is GCC 15. Later SUSE Linux releases will introduce the tick-tock model:

- Each even minor release of SUSE Linux (the tick release) introduces a new major version of GCC as the default compiler. This GCC version is supported during the LTS for the minor version that introduced it and also for the next SUSE Linux minor version. For example, if GCC 17 is introduced in SUSE Linux 16.2, it will be supported until the end of LTS for SUSE Linux 16.3.
- Each odd minor release of SUSE Linux (the tock release) comes with a new non-default major version of GCC. As this version is not the default one, you must explicitly invoke the binaries `gcc-x`, `g++-x` and `gfortran-x` to use it. These non-default versions are supported for 24 months.

17.2.1 Supported compiler flags

Any combination of the following compiler flags is supported:

- `-O0`, `-O2` and `-O3`
- `-ffast-math`
- `-flto`
- `-fpie` and `-fno-pie`
- `-fPIC`
- `-g`

The following options are also supported on AMD64/Intel 64:

- `-march=x86-64-v2` (the default one)
- `-march=x86-64-v3`
- `-march=x86-64-v4`

Other compiler flags are not supported by SUSE. However, we can assist in reporting issues to the upstream GCC project.

17.2.2 Supported language versions

For the C language, the most recent supported version is ISO/IEC 9899:2024 (known as C23) with GNU extensions.

For C++, the most recent supported version is ISO/IEC 14882:2017 (known as C++17) with GNU extensions.

For Fortran, we support versions up to Fortran 95 (ISO/IEC 1539:1997) with specific features of later standard versions as described in the manual.

SUSE also provides unsupported packages with compilers for Ada and Go (gcc-go).

17.3 The kernel module compiler

To build a kernel module, you need the same compiler version used to build the kernel. Therefore, SUSE provides the same GCC version—initially GCC 13.N.

The compiler for kernel modules is not intended for general use. The kernel module compiler may also be dropped in a future minor release of SUSE Linux.

17.4 The build compiler

There is a default compiler that is used internally to build SUSE Linux 16 packages in the version GCC 13. The build compiler is provided as an unsupported package in PackageHub.

Package maintainers can use the newer GCC available in the internal build service, but in this case, they must be aware of possible ABI issues (for example, avoid linking code written in different C++ standards).

17.5 GCC and C++ runtime libraries

GCC runtime libraries (`libgcc`, `libstdc++`) are updated to the versions of a new major version of GCC on a yearly basis in all SUSE Linux minor releases under LTS. The runtime libraries are updated during maintenance updates.

The runtime libraries are fully supported during the general support and LTS of each SUSE Linux minor version.

When security incidents may happen, these runtime libraries are the only part of the toolchain that can be used for that matter.

17.6 The GNU Binutils

The GNU Binutils are upgraded to the latest upstream version in all SUSE Linux 16 minor releases under general support of LTS.

17.7 The GNU project debugger

GDB is updated to the newest major version on all SUSE Linux 16 minor releases under general support or LTS. However, this means that certain functionality may be removed from the package.

17.8 LLVM

LLVM is available for use exclusively with MESA. Any other use of LLVM is **not** supported. Front-ends like Clang are not provided on SUSE Linux. You may get them only from a community-supported repository.

17.9 Compatibility and deprecation policy

SUSE maintains backward dynamic-linking compatibility for `glibc` and the C++ compiler runtime library. This means that a binary built on an earlier minor version of SUSE Linux 16 runs correctly on a later minor release.

Features deprecated by upstream are removed either from newer major versions of compilers or from all toolchain components in later SUSE Linux minor versions. In the case of `glibc`, the deprecated symbols are then removed from header files and are not available for link editing. Therefore, the code using these symbols will no longer compile or (statically) link.

17.10 Security considerations

Development tools like the compiler are not hardened to process untrusted input. In contrast, the GCC and C++ runtime libraries are the only parts of the toolchain that are hardened for this purpose.

18 Legal Notice

For SUSE trademarks, see https://www.suse.com/company/legal/. All other third-party trademarks are the property of their respective owners. Trademark symbols (®, ™ etc.) denote trademarks of SUSE and its affiliates. Asterisks (*) denote third-party trademarks.

All information found in this book has been compiled with utmost attention to detail. However, this does not guarantee complete accuracy. Neither SUSE LLC, its affiliates, the authors, nor the translators shall be held liable for possible errors or the consequences thereof.

A GNU Free Documentation License

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non-commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The

"Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or non-commercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

```
Copyright (c) YEAR YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.2
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled “GNU
Free Documentation License”.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.