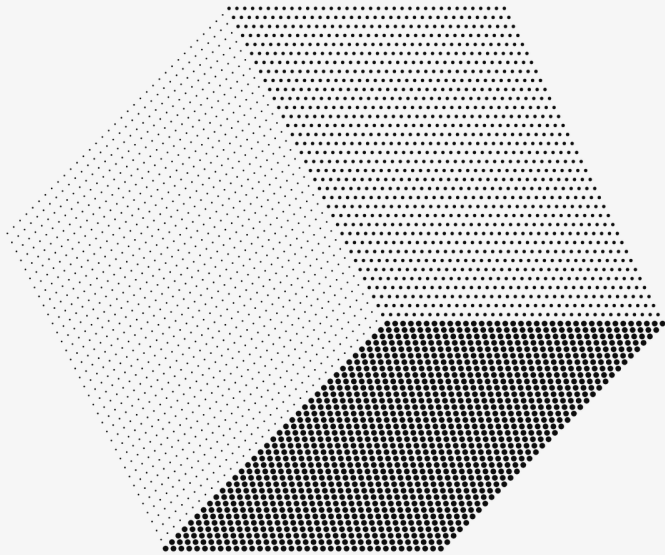




GUIDE 11 DEVICE SOFTWARE

When the software is the suspect

How device software is built and recorded under IEC 62304, what the manufacturer must have on file for each version, and how to read it against a failure.

AD

ABOUT

Most devices are now computers with a sensor and an actuator attached. When one fails, the question is often not whether a part broke but what the software did, in which version, and whether the manufacturer knew it could.

This guide explains the software life cycle the standards require, the records it leaves for every release, and how to request and read those records when a failure may have been in the code. It is written for readers who will not read the code, and do not need to.

FOR

Legal counsel who must establish what version was running, what it was known to do, and what was tested.

Manufacturers — software, quality, and regulatory teams who maintain the software file and decide what a problem report means.

Hospitals — clinical engineering and IT staff who manage versions, updates, and the devices on the network.

NOTE

This guide is general technical information. It is not legal advice. Terms follow IEC 62304:2006 with Amendment 1:2015 and FDA's 2023 guidance on premarket software documentation; a manufacturer's own procedures may name the same records differently.

CONTENTS

In this guide

01	Software has a life cycle	04
	Figure: the record a release leaves	05
02	What FDA saw	06
03	Requesting the record	07
04	Reading the record	09
05	What software failures look like	10
06	Where the record is weak	11
	Checklist	12
	Sources	13
	How Alpha Devices helps	14
	Terms	15

SERIES

This guide is one of fifteen practical guides from Alpha Devices. The full list is on the back cover; all are free at alphadevices.io/resources.html.

01 **Software has a life cycle**

IEC 62304 is the standard FDA recognises for the life cycle of medical device software. It does not say how to write software. It says what activities must happen, what each must produce, and how much of it is required by the harm the software could cause.

CLASS A, B, C	Each software system is classed by the worst harm a failure could contribute to after other risk controls are taken into account: A, no injury; B, non-serious injury; C, death or serious injury. The class sets how much design, verification, and documentation is required. The classification is itself a recorded decision.
ACTIVITIES	Planning, requirements, architecture, detailed design, implementation and unit verification, integration and testing, system testing, and release. Then maintenance, which repeats the cycle for every change.
SOUP	Software of unknown provenance: operating systems, libraries, and components the manufacturer did not develop under the standard. Each must be listed with its version, its known anomalies reviewed, and its failure modes considered in the risk analysis.
PROBLEMS	A problem resolution process: every reported problem recorded, investigated, classed for safety, and either fixed or closed with a reason. The problem report file is the software equivalent of the complaint file (guide 13).
VERSIONS	Every item under version control; every release identified, with the exact contents, build, and tools that made it. A version number on a label should lead to one and only one set of files.

NOTE

The safety class and the documentation level (next pages) decide how much of this record exists. Ask for the classification decision and its reasoning first.

FIGURE

The record a release leaves

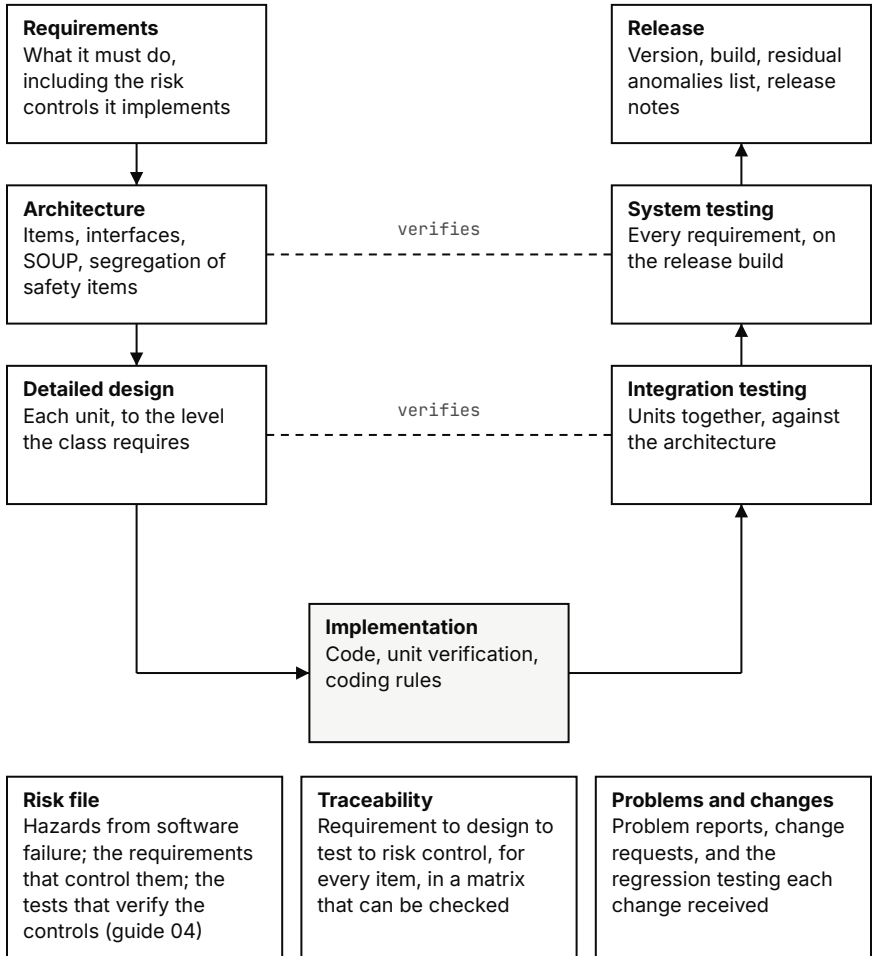


FIG 01

Each box is a document the standard requires, to a depth set by the software class, with a version. A failure is traced from the right side of the V to the left: which test should have caught it, which requirement it violates, which hazard it belongs to.

02

What FDA saw

A premarket submission contains part of the software record, at a depth FDA's guidance calls the documentation level.

01

Basic and enhanced

FDA's 2023 guidance replaced three levels of concern with two documentation levels. Enhanced applies where a failure or latent flaw could, before risk controls, present a probable risk of death or serious injury, and to Class III devices, combination products, and blood-establishment software. Enhanced submissions include the detailed design and unit and integration test evidence; basic ones summarise them.

02

What is in every submission

A software description; the risk management file; the requirements; the architecture; the development and maintenance process; verification and validation evidence; the revision history; and the unresolved anomalies, each with its effect and the reason it was left.

03

What is not

The code, the full test records, the problem reports, and everything after the submission: later versions, patches, and the letters to file that decided a change needed no new submission (guide 09).

KEY POINT

The unresolved anomalies list is the manufacturer's own statement of what it knew the software did wrong at release, and why it shipped.

03

Requesting the record

These are the items to name in a request, in the terms of IEC 62304, for the version that was running and every version since. The form of the request is in guide 05.

01

Classification and plan

The software safety classification with its rationale; the software development plan and the maintenance plan.

02

Requirements, architecture, design

The software requirements specification, the architecture, and the detailed design for the items involved in the failure mode.

03

SOUP list

Every SOUP item with version, the review of its published anomalies, and the risk analysis of its failure.

04

Verification records

Unit, integration, and system test protocols and reports for the release in question, and the regression testing for each change since.

05

Traceability

The matrix linking hazards, requirements, design, and tests. A missing link is a finding on its own.

06

Configuration and release

Release records with version, build, and contents; release notes and the residual anomalies list for each version; the record of which units received which version, and when.

07

Problems and changes

All problem reports and change requests for the software, with their safety assessment and closure, from first release to the present.

NOTE

Request each item for the version that was running and for every version since. A file that covers only the current release answers a different question.

04 Reading the record

Software records are read from the version backwards.

01 Fix the version

From the device log, the label, and the service record (guide 07). Version, build, and configuration. If they disagree, that is the first finding.

02 Read the anomaly lists

The residual anomalies for that version, and the release notes for every later version. “Fixed” entries in later notes describe what the earlier version did.

03 Find the requirement

The requirement the failure violates, and the hazard it was meant to control. No requirement means the hazard was not analysed, or the control was not specified.

04 Find the test

The test that verifies the requirement: its protocol, its result, the version it ran on, and whether it was repeated after later changes.

05 Read the problem reports

Every report with similar symptoms, before and after the event: how it was classed, what was found, and how it was closed. “Could not reproduce” is a closure, not a finding.

06 Check the SOUP

Operating system, drivers, and libraries: their versions, their published defects, and whether any match the failure.

05

What software failures look like

Software fails differently from hardware. It does not wear, and it does not fail at random; it fails every time the same conditions recur, and the conditions may be rare. Field reports describe the symptoms; these are the common ones.

TIMING

A fault that appears under a sequence of actions, a timing coincidence, or a particular load, and not on the bench. Timing and concurrency faults are software faults until shown otherwise.

RESETS

The device restarts on its own. A watchdog has caught the software not responding. The log shows the reset; the problem reports may show why.

MISMATCH

The display, the alarm, and the actual output disagree: a value shown that was not delivered, an alarm limit that did not act, a setting that reverted.

ARITHMETIC

A dose, a rate, a volume, or a time calculated wrongly for a particular unit, range, or rounding. Reproducible exactly, once the inputs are known.

NETWORK

Time-outs, lost messages, and stale data between the device and a network, a pump module, or a sensor. Each side logs its own view.

■ CAUTION

■ "No fault found" on a returned device is the expected result of a
■ software fault tested under the wrong conditions. It closes
■ nothing. The conditions in the log and the clinical record are the
■ test.

06

Where the record is weak

These are the findings an engineering reviewer looks for first in a software file.

01

The fielded version was not the tested version

System testing ran on one build; a later build shipped, with changes and no full regression.

02

A known anomaly, deferred

The failure mode is in the residual anomalies list of the release, with a rationale that rests on probability rather than on a control.

03

A hazard with no requirement

The risk file lists the hazard; nothing in the requirements controls it; nothing in the tests checks it.

04

A test that was never repeated

The requirement was verified once, before several changes to the items that implement it.

05

SOUP without a review

An operating system or library listed with no version, or with no review of its defects.

06

Problem reports closed without analysis

Similar reports closed as “cannot reproduce” or “user error” with no record of what was tried.

07

A patch outside the process

A change made quickly in the field with a shortened record: no requirement change, no risk review, limited testing.

NOTE

Each finding is a comparison between what the standard requires and what the file contains. The standard is the measure; the file is the evidence.

CHECKLIST

Device software at a glance

- ☐ Version, build, and configuration established from log, label, and service record
- ☐ Software safety classification and rationale obtained
- ☐ Documentation level and the submission's software section read (guide 09)
- ☐ Residual anomalies list for the version, and release notes for every later version
- ☐ Requirement and hazard for the failure mode located; traceability checked
- ☐ Test protocol and report for that requirement; version it ran on; regression since
- ☐ Problem reports with similar symptoms, with classification and closure
- ☐ SOUP list with versions and anomaly reviews
- ☐ Changes since release: change requests, letters to file, regression records
- ☐ Which units received which version, and when
- ☐ Conditions from the log and clinical record written as a reproduction test

SOURCES

Sources and further reading

IEC 62304:2006 + A1:2015 — Medical device software, software life cycle processes

iec.ch

FDA — Content of Premarket Submissions for Device Software Functions (June 2023)

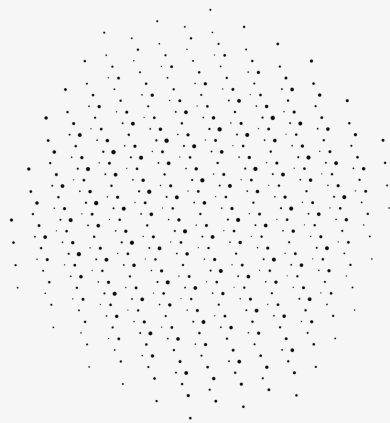
fda.gov

FDA — General Principles of Software Validation (2002)

fda.gov

eCFR — 21 CFR Part 820, Quality Management System Regulation; design and development under ISO 13485 clause 7.3

ecfr.gov



SERVICES

We read the software file as the engineers who would have built it

We establish the version, trace the failure mode through requirements, risk controls, and tests, and reproduce the conditions on an exemplar device running the same build.

01

Version and record

The exact version from the device, and the software record for it requested by name under IEC 62304.

02

Trace and test

The failure mode traced to its requirement, hazard, and test; the test repeated on the fielded build.

03

Reproduction

The conditions from the log and the clinical record set up on an exemplar, with the device's own log as the witness.

TERMS

Terms used in this guide

Software system

The software of a device, or a part of it, considered as one item for classification and documentation.

Software safety class

A, B, or C: the class assigned to a software system by the worst harm a failure could contribute to, after other risk controls.

SOUP

Software of unknown provenance: components the manufacturer did not develop under the standard, such as an operating system or a library.

Software requirements

The statements of what the software must do, including the risk controls it implements, each one verifiable.

Verification

Confirmation by examination and objective evidence that specified requirements have been fulfilled: unit, integration, and system testing.

Residual anomaly

A known defect left in a release, listed with its effect and the reason it was accepted.

Problem report

The record of a reported problem with the software: its investigation, its safety classification, and its closure.

Documentation level

FDA's basic or enhanced level, which sets how much of the software record a premarket submission must contain.

CONTACT

Tell us about the device.

Send a short description of the device, the event, and any deadlines. We will tell you what to preserve now and recommend the next steps.

EMAIL info@alphadevices.io

WEB alphadevices.io

GUIDES

- | | | | |
|----|-------------------------------------|----|---|
| 01 | A device failed. What happens next? | 09 | How a device reaches the market |
| 02 | Searching MAUDE with purpose | 10 | Why the alarm did not help |
| 03 | Understanding waterfall design | 11 | When the software is the suspect |
| 04 | Reading an ISO 14971 risk analysis | 12 | Use error is a design question |
| 05 | The manufacturer's record | 13 | Complaints, MDRs and CAPA |
| 06 | Using LLMs to review case documents | 14 | Recalls, corrections and quiet fixes |
| 07 | What the device recorded | 15 | What did they know, and when? |
| 08 | The joint inspection | | |

Alpha Devices

2101 Brandywine Street, Suite 200
Philadelphia, PA 19130

Independent medical device engineering. © 2026 Alpha Devices.

