appendix

Guidelines for Medical Device Software Registration Review

( Revised 2022 )

The purpose of this guideline is to guide registration applicants to standardize the life cycle process of medical device software and prepare medical device software registration application materials, and to standardize the technical review requirements of medical device software , so as to provide a reference for the system verification of medical device software and quality management software.

This guideline is a general requirement for medical device software. The registration applicant shall determine the applicability of the specific content of this guideline according to the product characteristics and risk level , and explain the reasons if it is not applicable. Registration applicants can also use other alternative methods that meet the regulatory requirements, but they need to provide detailed research materials.

This guiding principle is formulated under the current regulations, mandatory standard system and current scientific and technological capabilities and cognition level. With the continuous improvement of regulations and mandatory standard system and the continuous development of scientific and technological capabilities and cognition level The content will also be adjusted from time to time.

This guideline serves as a guiding document for registration applicants , reviewers and inspectors. It does not include administrative matters involved in review and approval, nor is it enforced as a regulation . The guideline should be used under the premise of meeting the requirements of the law.

This guideline is the basic guideline of the digital health (Digital Health) guideline system, and is also a general guideline for medical device software. Other guidelines for medical devices that contain or involve software can be targeted based on this guideline. Adjust, modify and improve.

1. Scope of application

This guideline applies to the registration application of medical device software, including the second and third types of independent software and medical devices containing software components (including in vitro diagnostic medical devices) ; it is applicable to the registration application of self-developed software and ready-made software.

This guideline can also be used as a reference for system verification of medical device software and quality management software .

2. Main concepts

(1) Medical device software

Medical device software includes software that is itself a medical device or software contained in a medical device. The former is medical device independent software (referred to as independent software) , and the latter is medical device software components (referred to as software components), see Figure 1 for details .

Standalone software (SaMD) refers to software that has one or more medical purposes/uses, can complete its intended use without medical device hardware, and run on a general-purpose computing platform [[1]](#footnote-1). The general computing platform meets the safety requirements of information technology equipment (including electromagnetic compatibility), and complies with GB 4943.1 , GB/T 9254 and other standards.

Independent software can be divided into general-purpose independent software and special-purpose independent software. The former is usually used in conjunction with multiple medical devices based on general data interfaces, such as medical image processing software and patient monitoring software; the latter is based on general-purpose, dedicated data interfaces and specific medical equipment. The combined use of the devices can be regarded as accessories of medical devices, such as dynamic ECG data analysis software and ophthalmic microscope image processing software.

software component (SiMD) refers to software that has one or more medical purposes/purposes , controls/drives medical device hardware, or runs on a medical computing platform . The medical computing platform meets the safety requirements (including electromagnetic compatibility) of medical electrical equipment (GB 9706 series), laboratory electrical equipment (GB 4793 series) or active implantable medical devices (GB 16174 series); The general computing platform is used jointly to form a system, which is regarded as a medical computing platform as a whole .

Software components can be divided into embedded software components and external control software components. The former runs on the medical computing platform and controls/drives the hardware of medical devices , such as the embedded software (ie firmware) contained in ECG machines and EEG machines ; The operator runs on a general computing platform to control/drive medical device hardware , such as CT , MRI image acquisition workstation software [[2]](#footnote-2).

Figure 1 Types of medical device software

Independent software, as a medical device or medical device accessory, is usually registered separately. In special cases, it can be registered with the medical device. At this time, although it does not control/drive the medical device hardware , it runs on the medical computing platform from the product point of view , so it is regarded as a software component , such as Dedicated stand-alone software can be registered as an accessory with the medical device .

As a component of a medical device or medical device components and accessories, software components should not be registered separately, but must be registered with the medical device as a whole .

(2) System software, application software, middleware, support software

System software refers to software designed to ensure the normal operation of computer systems, such as operating system software and virtual machine software . Application software refers to software designed to realize the specific needs of computer users, such as browser software, database software , and security software . Middleware is between system software and application software, relying on the support of system software, and at the same time providing support for application software , such as distributed computing platform software . Support software refers to software designed to develop and test other software , such as software development tools and software testing tools.

Medical device software belongs to application software, and its normal operation usually needs to be based on system software, or at the same time requires the support of application software (including other medical device software), middleware, and support software.

Some registration applicants will develop medical middleware as a public support platform for medical device software. Since these medical middleware are necessary for the normal operation of medical device software, they should be considered as an integral part of medical device software.

Some supporting software (such as VTK, ITK) have their own algorithm library, and the medical device software development process has integrated the relevant content of the algorithm library into itself , so the normal operation of medical device software requires the support of these supporting software .

In this guideline, other medical device software and medical middleware necessary for the normal operation of medical device software are referred to as necessary software, and the system software, general application software, general middleware, and support software necessary for their normal operation are collectively referred to as external software environment . The necessary software is registered separately as medical device software, and it is enough to clarify the mutual interface relationship and technical characteristics. The external software environment does not contain prerequisite software , nor is it medical device software .

(3) Software life cycle

Software life cycle (also known as software life cycle) refers to the time period from concept definition to decommissioning of a software system , including software development planning, software requirements analysis, software design, software coding, software testing, software release, software deployment, and software maintenance. , software outage and other stages . Among them, the time period from software requirement analysis to software release is called software development life cycle .

Software development planning mainly determines the goals and feasibility of software development. Software requirements analysis is to convert regulations, standards, users, products and other requirements into software requirements specification/software requirements specification (SRS) . Software design is the transformation of software requirements specification into software design specification /software design specification (SDS) through design activities . Software coding is the transformation of software design specifications into software systems by writing source code . Software testing is to ensure the quality of software systems through various testing activities . Software release is the finalization of a software system into a product . Software deployment refers to the delivery, installation, setup, and configuration of software systems . Software maintenance is to realize the update requirements after the software system is released . Software outage (ie software delisting ) refers to the termination of sales and after-sale services of the software system , and the after-sale service is usually stopped later than the sale .

A software life cycle model refers to a set of frameworks including processes, activities and tasks, spanning the software life cycle process from software requirements analysis to software outage, each process can be subdivided into several activities, and each activity can be subdivided into several tasks. Among them, the software development life cycle model is an important part of the software life cycle model. Common models include waterfall model, iterative model, incremental model, V model and so on .

Agile development is a software development method that combines human-centered, iterative and incremental . Common software development life cycle models include SCRUM , extreme programming, etc. Agile development is based on four philosophies: human interaction is better than process and tools, usable software is better than detailed documentation, customer collaboration is better than contract negotiation, and responding to change is better than following a plan. Therefore, the use of agile development should take into account the relevant requirements of the quality management system, focusing on control requirements such as software updates, documents and records.

Applicants for registration can choose an appropriate software life cycle model based on the product characteristics , risk level and quality management system requirements of the software, and establish the corresponding software life cycle process with reference to relevant international, national and industry standards.

(4) Software testing, software verification, and software validation

Software testing is the basic measure of software quality assurance, and it is also an important method for software verification and software validation. There are different classification methods from different angles.

From the perspective of test basis, it can be divided into black box testing and white box testing. Among them, black-box testing refers to testing based on input and output , while white-box testing refers to testing based on source code . Black-box testing and white-box testing can be used in combination, that is, gray-box testing. White-box testing can be divided into static and dynamic analysis/ testing according to whether or not to run the source code.

From the perspective of the testing process, it can be divided into unit testing, integration testing, and system testing. Among them, unit testing is to test software units, usually using white-box testing; integration testing is to test software items (composed of several software units , namely software modules ), combining white-box testing , black- box testing , and gray-box testing ;System testing is to test the software system (composed of several software items), usually using black box testing, which can be divided into functional testing, performance testing, concurrency testing, stress testing, interface testing, memory testing, Compatibility testing, user interface testing, installation and uninstallation testing , security testing , etc.

From the perspective of test implementation, it can be divided into internal testing, user testing , and third-party testing. Among them, internal testing refers to the testing performed by the registration applicant , including unit testing, integration testing , system testing, a combination of white-box testing , black- box testing , and gray-box testing ; user testing refers to the expected user testing in real or simulated usage scenarios. Software systems are tested using black-box testing; third-party testing refers to the testing of software systems by third-party organizations, usually using black-box testing.

Regression testing refers to software testing that is used to determine that software updates have had no adverse effects and have not introduced risky unacceptable new defects. Regression testing needs to carry out appropriate unit testing, integration testing, system testing, user testing, third-party testing and other testing activities according to the type, content and degree of software update .

Software verification means that the output of a certain stage of software development and software maintenance meets the input requirements by providing objective evidence. Software verification includes a series of activities such as source code review, static and dynamic analysis/testing , unit testing, integration testing, system testing, and design review, and is the basis for software validation.

Software validation refers to providing objective evidence that the software meets the user's needs and intended use . Software validation is a design validation based on process control, including a series of activities such as user testing, clinical evaluation, and design review. It is necessary to ensure that the software meets user needs and intended use , and that the risks of known remaining defects of the software are acceptable.

software testing requirements in combination with the product characteristics and risk level of the software , and clarify the test coverage requirements such as statements, judgments, conditions, and paths, so as to ensure the quality of software verification and software validation. All source code should be tested, which can be implemented by combining white box testing, black box testing, gray box testing and other methods.

(5) Software traceability analysis

Software traceability analysis, as one of the important activities of software verification and software validation, refers to tracking the relationship between software requirements, software design, source code, software testing, and software risk management, and analyzing the correctness and consistency of the identified relationships. , completeness and accuracy.

The software life cycle process needs to carry out traceability analysis activities, as shown in Figure 2. The software requirements analysis stage retrospectively analyzes the relationship between software requirements and product requirements , software requirements and risk analysis . The software design stage retrospectively analyzes the relationship between software design and software requirements , software design and risk control . The software coding stage retrospectively analyzes the relationship between source code and software design, source code and test cases . The internal testing stage retrospectively analyzes the relationship between unit testing, integration testing, and system testing, test cases and software design at all levels , system testing and software requirements , and system testing and risk management . The user testing stage retrospectively analyzes the relationship between user testing and product requirements, user testing and risk management. Software updates also require appropriate software traceability analysis activities .

Figure 2 Software traceability analysis

registration applicant shall establish a software traceability analysis process and standardize the requirements for relevant activities of software traceability analysis to ensure the quality of software verification and software validation. Taking into account the actual situation of the industry, the source code traceability analysis activities can be traced back to the software unit (namely listed).

(6) Software update

1. Main Concepts

Software update refers to any modification made by the registration applicant to the software during the whole process of the software life cycle, also known as software change and software maintenance. Software maintenance generally has the same meaning as software updates, but can specifically refer to post- release software updates.

Software updates are classified differently from different perspectives. From the perspective of update results, it can be divided into major updates and minor updates. Major updates refer to software updates that affect the safety or effectiveness of medical devices, and vice versa.

From the perspective of update content, it can be divided into enhancement type update and correction type update (ie software defect repair). Enhanced updates can be divided into perfect updates and adaptive updates . Perfect updates refer to software updates that change software attributes such as functions, performance , and interfaces , and adaptive updates refer to software updates that adapt to the new operating environment . From the perspective of update results, it can be further divided into major enhancement updates and minor enhancement updates . Corrective updates can be divided into corrective updates and preventive updates. Corrective updates refer to software updates that repair software that exists and have caused operating failure defects, and preventive updates refer to repairing software that exists but has not yet caused operating failure defects. software updates ; these are generally minor updates unless they affect the safety or effectiveness of the medical device.

This guideline focuses on the safety and effectiveness of medical device software and categorizes software updates into:

(1) Major software updates: Enhanced updates that affect the safety or effectiveness of medical devices, that is, major enhanced software updates, should apply for change registration .

(2) Minor software updates: enhancement updates and correction updates that do not affect the safety and effectiveness of medical devices, including minor enhancement software updates and correction software updates , are controlled through the quality management system, and there is no need to apply for change registration. The corresponding registration application materials shall be submitted when the registration is changed next time.

In addition, software construction refers to the compilation of software to generate a working version, which conforms to the definition of software update and is controlled by the quality management system. The requirements for registration application materials are the same as those for corrective software updates . Recall-related software updates include software updates that lead to recalls and software updates used for recall measures. Whether enhancement updates or correction updates are major software updates, they are handled in accordance with the requirements of relevant laws and regulations on medical device recalls and are not within the scope of this guideline .

Software updates follow the principle of high risk, that is, major and minor software updates at the same time are handled as major software updates , and simultaneous enhancement and correction software updates are handled as enhanced software updates . Software updates need to consider introducing a rollback mechanism to ensure the continuity of medical business, especially for high-risk software.

Software redevelopment means that the registration applicant abandons the original software and develops new software , which does not belong to the scope of software update and is treated as the initial release.

Judgment principles for major software updates

If a software update affects the intended use, usage scenarios or core functions of a medical device, it is in principle a major software update, and its determination principles include but are not limited to:

(1) Complete software update: if it affects the user's decision-making (including decision-making ability, decision-making results, decision-making process, user actions) or the safety of personnel (including patients, users, and other related personnel), it is a major software update , such as software input Changes in output data types , architecture, user interface relationships, physical topology, core algorithms, core functions, diagnosis and treatment processes or intended uses, etc., software systems, high-risk software items/software units undergo code refactoring, security level changes, adjustments The method of alarm, etc.; while the operation efficiency is simply improved, the diagnosis and treatment process or working language can be configured (that is, the user can retain the original diagnosis and treatment process or working language) , the textual modification of the user interface , and the code reconstruction of low- and medium-risk software items/software units Such conditions are generally not considered major software updates unless they affect the safety or effectiveness of the medical device.

(2) Adaptive software update: if the software operating environment spans incompatible computing platforms (including hardware configuration, external software environment, necessary software, and network conditions ) , it is a major software update . Change to iOS, replace browser kernel, necessary software, network conditions change from local area network to wide area network, computing platform change from general computing platform to medical computing platform, etc.; the compatibility version of expected system software, support software , and general middleware Updates, patch updates are generally not considered major software updates unless they affect the safety or effectiveness of a medical device.

In summary, the types of medical device software updates are shown in Figure 3.

Figure 3 Types of Medical Device Software Updates

The scope of major software updates is dynamically adjusted as awareness levels and technical capabilities improve, as well as analysis of adverse events and recalls .

(7) Software version

Software has no physical entity, and can only manage software updates through status identification to ensure software quality. The software version uses different fields to distinguish the software update type, and then identifies the software status. Therefore, the software version and the software have a one-to-one correspondence between the table and the inside, and are also an indispensable part of the software identification.

The software version can be divided into software release version and software complete version [[3]](#footnote-3). The software release version only reflects major software updates, that is, it is limited to major enhancement software updates, and its changes mean major software updates, and vice versa; software complete versions reflect major software updates. , All types of minor software updates, including major enhancement software updates, minor enhancement software updates, corrective software updates, software builds (if applicable), changes in different fields mean different types of software updates, and vice versa.

If the software release version changes , it means that the software has undergone a major update and should apply for change registration . If the full version of the software changes but the software release version does not change, it means that the software is only slightly updated. At this time , it is controlled by the quality management system, and there is no need to apply for change registration . For example, the software version naming convention is XYZB, where X represents a major enhancement software update, Y represents a minor enhancement software update, Z represents a corrective software update, and B represents a software build, then the software release version is X, and the full software version is XYZB At this time, if X changes, you should apply for change registration , and if Y, Z , and B change but X does not change , you do not need to apply for change registration .

registration applicant should comprehensively consider the characteristics of software products , quality management system requirements , compliance and other factors to formulate software version naming rules and record them , clarify the number of digits, scope, and meaning of the fields, covering all types of software updates, and the field meanings are clear and non-existent. There is no ambiguity and no contradiction, it can distinguish major software updates from minor software updates, and ensure that the version changes of software updates meet the requirements of software version naming rules. At the same time, consider the requirements of guiding principles such as medical device cybersecurity and artificial intelligence medical devices.

software version naming rules also follow the principle of high risk , that is, if a field represents both a major software update and a minor software update, the field will be treated as a major software update and will be part of the software release version.

software with a user interface , the released version of the software and the full version of the software can be reflected in the login interface, main interface, "About" or "Help" interface . Software without a user interface needs to provide a method for obtaining the full version of the software to clarify the software version information [[4]](#footnote-4).

Product technical requirements indicate the software release version and software version naming rules, among which the software version naming rules must be consistent with the quality management system. The test report provides a photo of the software version interface or lists the software version information . The software with a user interface reflects the software release version and the complete software version, and the software without a user interface reflects the complete software version. The manual indicates the software release version.

If the software modules (including medical middleware) are independently version controlled, they must also meet the version control requirements, and the relationship with the software system version control must be clearly defined.

(8) Software Algorithms, Software Functions, and Software Uses

Software algorithms are the basis of software functions, and the two are in a many-to-many relationship, that is, a software algorithm can be used by one or more software functions, and a software function can contain one or more software algorithms. Similarly, the relationship between software functions and software uses is also the same.

From the user's point of view, software algorithms are invisible internally, and software functions and software uses are externally visible. Considering that software functions are the link between software algorithms and software uses, software functions are used as the main line of software security effectiveness evaluation. For example, region growing algorithms can implement image segmentation functions, which can be used for lesion contour identification .

Software functions have different classification methods from different perspectives. From the perspective of importance, it can be divided into core functions and non-core functions. The core functions refer to the functions necessary for the software to complete the intended use in the expected usage scenarios, and vice versa is the non-core functions.

From the perspective of technical characteristics , it can be roughly divided into processing functions , control functions , and safety functions . Processing functions refer to processing medical device data (ie, objective data generated by medical devices for medical purposes) or model-based calculations (such as radiation dose models). , Pharmacokinetic model) , the control function refers to the function of controlling/driving the hardware operation of the medical device, and the safety function refers to the function of ensuring the safety of the medical device .

The processing function can be divided into pre-processing function and post-processing function from the perspective of data flow. The former refers to the processing function in the process of collecting human anatomy and physiological information to generate medical device data , such as filtering, noise reduction, correction, reconstruction and other functions ; the latter It refers to the processing functions that use medical device data to generate diagnosis and treatment information or perform medical intervention , such as translation, scaling, rotation, filtering, measurement, segmentation , fusion , 3D reconstruction, treatment plan formulation, pharmacokinetic model calculation, gene sequencing, analysis , etc. function . From the perspective of complexity, post-processing functions can be divided into simple functions and complex functions. The former refers to functions that operate on existing medical information rather than generating new medical information, such as translation, zooming , and rotating functions; the latter refers to generating New medical information functions, such as filtering, measurement, segmentation , fusion , 3D reconstruction, treatment planning, pharmacokinetic model calculation, gene sequencing, analysis, etc.

The functions of the independent software are all post-processing functions, and the functions of the software components are mainly control functions, pre-processing functions, and also post-processing functions. Considering that the control function and pre-processing function are usually evaluated together with medical device hardware products, the processing function refers to the post-processing function unless otherwise specified; The functions are described side by side.

From the perspective of the scope of supervision, it can be divided into medical device functions and non-medical device functions. Medical device functions refer to software functions that can be used as the basis for the definition of medical devices, such as the measurement, processing, and modeling of medical images, physiological parameters, and in vitro diagnostic data. Functions such as calculation and analysis; on the contrary, it is a non-medical device function , such as charging and pricing, administrative office and other functions, which are not subject to supervision; the patient information management function necessary to realize the function of medical device belongs to the function of medical device. The two should be split as far as possible through modular design. If it is technically impossible to split , the non-medical device functions should be considered as an integral part of the medical device software .

Software algorithms can be divided into core algorithms and non-core algorithms from the perspective of importance. The core algorithms are the algorithms necessary to realize the core functions of the software, and vice versa. From the perspective of complexity, it can be divided into simple algorithms and complex algorithms. The former is simple and clear in principle or based on mature formulas, while the latter is usually based on model research, with many assumptions and many influencing factors . There are also differences in speed and so on . From the perspective of interpretability, it can be divided into white-box algorithms and black-box algorithms. The former can be associated with existing knowledge, while the latter is difficult to establish association with existing knowledge. The former is better than the latter in interpretability.

The use of software can usually be divided into assisted decision-making and non-assisted decision-making. Among them, assisted decision-making refers to assisting users (such as medical staff and patients) in medical decision-making by providing suggestions for diagnosis and treatment activities, such as identification of lesion characteristics, determination of lesion properties, medication guidance, and formulation of treatment. Planning, etc., is equivalent to the user 's "assistant" ; on the contrary , only providing medical reference information without medical decision-making is non-aided decision-making, including process optimization, diagnosis and treatment drive, the former such as simplified diagnosis and treatment process, the latter such as measurement, segmentation , 3D reconstruction, etc., which are equivalent to the user 's "tool". At the same time, assisted decision-making and non-assisted decision-making can be subdivided into real-time and non-real-time from the perspective of real-time, and the risk of the former is usually higher than that of the latter. Therefore, software functions can be divided into auxiliary decision-making functions and non-aided decision-making functions, real-time functions and non-real-time functions from the perspective of software use.

From the perspective of maturity, software algorithms, software functions, and software uses can be divided into two types: mature and brand-new. Mature refers to the situation where safety and effectiveness have been fully proven in medical practice, and brand-new refers to unmarketed or safety and effectiveness. Situations that [[5]](#footnote-5)have not been fully substantiated in medical practice . Similarly, software can also be divided into brand-new software and mature software. If one of the algorithms, functions, and uses of the software is a brand-new type, it belongs to brand-new software; otherwise, it belongs to mature software. Because new software has more potential unknown risks than mature software, this guideline is based on core functions and core algorithms, and focuses on the security and effectiveness of new software.

3. Basic principles

(1) Based on software features

With the rapid development of information and communication technology, the application of software in medical devices is becoming more and more common, and its role is becoming more and more important. However, the accompanying quality problems are also increasing. Medical device recall data shows that the number of software-related recalls continues to increase, which is significantly higher than the overall level of medical devices in the same period. At the same time, it is not uncommon for software failures to cause patient death or serious injury. Therefore, the seriousness of software quality problems cannot be ignored , and it is necessary to strengthen software quality assurance work based on software characteristics .

Software has no physical entity, and human factors are everywhere in the process of development and use. Software testing cannot exhaust all situations due to time and cost constraints, so software defects cannot be avoided. At the same time, software updates are frequent and rapid, minor updates can lead to serious consequences, and there are cumulative effects and regression problems (that is, every time a number of defects are fixed, a new defect is created), so software defects cannot be eradicated. Therefore, software defects can be regarded as one of the inherent attributes of software, which is also the root cause of the more prominent software quality problems.

There are many differences between software and hardware: hardware is a physical entity, and software is a logical relationship; hardware change cycles are long , and software updates are easy , fast , and frequent ; hardware has wear problems, but software has no wear and tear but has cumulative effects and degradation problems; hardware quality Depends on design and development and production, software quality depends on design and development, and has nothing to do with production; hardware failures have symptoms before they occur, software failures often occur suddenly without symptoms, and software failure rates are much higher than hardware; hardware components can be standardized, and software modules Standardization is more complicated; minor changes have limited impact on hardware, but may have serious impact on software; hardware inspection can basically guarantee quality, but software testing is not enough to ensure quality. These differences make traditional hardware quality control methods often fail to achieve expected results for software quality assurance .

In view of the characteristics of software, the security and effectiveness of software can only be guaranteed by comprehensively considering the requirements of risk management, quality management and software engineering. Registration applicants need to adopt good software engineering practices to improve the quality management system based on the degree of software risk, and carry out software quality assurance work as soon as possible, focused on, and comprehensively for the main reasons for software recalls such as algorithms, interfaces, updates , and exception handling .

(2) Risk orientation

have different life cycle quality control requirements and registration application data requirements.

The degree of software risk is expressed by the software security level. The higher the software security level, the stricter the life cycle quality control requirements and the more detailed the registration application materials. The software security level is divided into three levels : minor, moderate and severe based on the degree of software risk [[6]](#footnote-6). The minor level means that the software is unlikely to cause harm, the medium level means that the software may directly or indirectly cause minor (not serious) damage, and the serious level means that the software may cause harm. Serious injury or death, directly or indirectly.

The software security level can be comprehensively determined in combination with the intended use, usage scenarios, and core functions of the software [[7]](#footnote-7). Among them , the intended use mainly considers the type of use of the software (such as treatment, diagnosis, monitoring, screening), the degree of importance (such as important role, reference role, supplementary role), the degree of urgency (such as critical situation, serious situation, ordinary situation), Maturity (mature, brand-new) and other factors , the usage scenarios mainly consider the use place of the software (such as outpatient, surgery, hospitalization, emergency, family, transfer, public places), disease characteristics (such as severity, urgency, contagiousness), Applicable people (such as adults, children, the elderly , pregnant women ), target users (such as medical staff , patients) and other factors , the core functions mainly consider the functional type of the software (such as importance, technical characteristics , complexity, maturity), core Algorithms (such as importance, complexity , interpretability, maturity) , input and output (input data such as medical images, physiological parameters, in vitro diagnostic data, and output results such as processing, measurement, analysis, etc.), interfaces (such as applications Program interface (API), data interface, product interface) and other factors .

The software security level can also be determined according to the risk level determined by risk management. The software security level and the risk level can be classified differently, but there is a corresponding relationship between the two. Therefore, the software security level can be determined according to the risk level, but it should be Judgments are made before risk control measures are taken , and subsequent external risk control measures (including software measures and hardware measures) can be used to reduce the initial software security level .

Software risk management needs to pay attention: the software itself is not dangerous, but it may cause dangerous situations; software failures appear as random failures, but they are actually systematic failures, and the probability of occurrence of dangers caused by software failures is difficult to calculate, so the degree of software risk is based on The severity of injury can be determined in combination with the probability of injury caused by dangerous situations ; the software components need to carry out risk management work with the medical devices they belong to .

software security level can also be judged by referring to the adverse events and recalls of similar medical device software that have already been listed. Moderate, no adverse events and only three or no recalls are minor.

registration applicant shall establish the corresponding software life cycle process in combination with the requirements of the quality management system , and carry out software quality assurance work that matches the software security level . At the same time, the corresponding registration application materials are submitted based on the software security level, and the registration application materials are all derived from the documents formed in the software life cycle process.

Although there are technical differences between independent software and software components , the quality control principles and requirements of the life cycle process are the same. Therefore, the requirements for registration and application materials of the two are basically the same , and the specific content is slightly different. Please refer to Chapter 8 for details.

Quality control of the whole life cycle

Due to the complexity of the software itself and the limitations of software testing, the quality assurance activities in the software development process are not enough to ensure the safety and effectiveness of the software. Therefore, the software quality control requirements should be considered in the entire life cycle of medical devices, and software risk management, Software configuration management, software defect management, and software traceability analysis run through the entire life cycle of medical devices .

Conduct adequate and effective software verification and validation activities prior to market launch to identify software foreseeable risks and reduce them to acceptable levels. Continue to carry out software quality assurance work after listing, identify unforeseen risks in the early stage in combination with user complaints, adverse events and recalls , and take necessary measures to ensure software quality; at the same time, based on the assessment of software update requirements, implement software update activities to meet the needs of users new requirements, and carry out appropriate software verification and validation activities to ensure the quality of software updates; in addition, software outages consider user notification and follow-up services, data migration, patient data and privacy protection requirements.

In short, digital medical technology is in a stage of rapid development, new technologies emerge in an endless stream, and the impact on the medical device industry is increasingly far-reaching, and its regulatory issues urgently need to be strengthened. No matter what kind of new digital medical technology, software as its technical foundation can still follow the above three basic principles to carry out the corresponding safety and effectiveness evaluation work.

4. Ready-made software

(1) Main concepts

that the registration applicant controls the complete life cycle is called self-developed software (if not expressly referred to as software ) . Finished software, outsourced software. Among them, the legacy software refers to the software developed by the registration applicant before but cannot obtain sufficient development records now, that is, the software that the registration applicant cannot prove that it has a complete life cycle control, involving application software and middleware. Finished software refers to software developed by a third party and usually available , that is, software for which the registration applicant has not carried out complete life cycle control, involving system software, application software, middleware, and supporting software, such as open source/closed source software, free/ Paid software, etc. Outsourced software refers to the software developed by a third party entrusted by the registration applicant, that is, the software for which the registration applicant only controls part of the life cycle, involving application software and middleware.

The relationship between off-the-shelf software and medical device software can be divided into two categories. One is ready-made software as a component of medical device software, that is, ready-made software components, including legacy software, finished software, outsourcing software, involving medical application software, medical middleware , and belonging to the supervision object; whether it is designed for medical use or not, ready-made software As an integral part of medical device software, the functions of software components belong to medical device functions, because if they are non-medical device functions, they should be directly deleted from the medical device software . The other type is ready-made software as a component of the operating environment of medical device software, that is, the external software environment, which is mainly finished software, involving system software, general application software, general middleware, and support software . Consider its impact on medical device software from a management perspective .

In summary, the types of ready-made software are shown in Figure 4.

Figure 4 Types of off-the-shelf software

embedded software components , medical device software needs the support of external software environment to run normally . At the same time, medical device software can be combined with self-developed software and off-the-shelf software components , some of which use off-the-shelf software components, that is, some of the usage methods; and all of the off-the -shelf software components can be used , that is, all the usage methods . Therefore, the use of off-the-shelf software has the characteristics of universality, flexibility, and complexity.

Since off-the-shelf software is usually not designed for medical use , especially in the external software environment, it may not be able to meet the relevant requirements of medical devices. Unused functions may adversely affect the medical device software through the coupling relationship, and the registration applicant does not fully survive the off-the-shelf software. Cycle control, so the risk of using off-the-shelf software is relatively high . In addition, off-the -shelf software components directly implement medical use, and the risks are relatively higher .

The use of ready-made software by the registration applicant shall ensure the safety and effectiveness of the medical device software. The developer of the finished software and the external software environment, as the medical device supplier, does not bear the relevant responsibilities of the registration applicant. Therefore, the registration applicant needs to distinguish the ready-made software components and the external software environment based on the type , characteristics and industry usage of the ready-made software, and comprehensively consider the ready-made software's extensive use, technology maturity, after-sales support, and functions, performance, and compatibility. , ease of use, reliability, maintainability, portability, network security and other requirements, adopt a risk-based full life cycle management method for quality control , especially in procurement, design and development, post-market monitoring, etc., while developing gaps Analyze and take remedial measures if necessary to ensure safety and effectiveness.

Due to the differences in the quality control requirements between self-developed software and ready-made software, ready-made software components and external software environment, and some use methods and all use methods, the corresponding registration application materials are different. For specific requirements , please refer to Chapter 8. In addition, medical device software can use multiple versions, multiple, and multiple ready-made software at the same time , which needs to be considered as a whole and provide corresponding registration application materials.

(2) General considerations for ready-made software

1. Off-the-shelf software components

The requirements for software updates and software versions of off-the-shelf software components are basically the same as those for self-developed software, and the principle of high risk is also followed.

If a major software update occurs to a ready-made software component, it should also apply for change registration . If a minor software update occurs, it is controlled by the quality management system, and there is no need to apply for change registration .

Registration applicants should formulate version naming rules for ready-made software components according to the requirements of the quality management system, and also need to consider compliance requirements. If the software version naming rules of off-the-shelf software component developers meet the compliance requirements, they can be used directly.

2. External software environment

There is a coupling relationship between medical device software and the external software environment, which needs to be considered as a whole.

From the perspective of medical device software, the software security level determination needs to consider the impact of the failure of the external software environment. The software requirements specification document and software test plan list the basic conditions of the external software environment , and the software test report lists the ready-made software contained in the external software environment. name, full version, patch version. Software verification and validation are implemented based on all ready-made software contained in the external software environment. If multiple versions of the same ready-made software are used, software verification and validation are required for each version . According to risk control requirements, medical device software must have the self-checking function of the external software environment when necessary to ensure that it can operate normally. Similarly, when necessary , the manual shall clarify the delivery, installation, setting, configuration, update, usage restrictions, warnings, etc. of all ready-made software contained in the external software environment .

From the perspective of external software environment, its own risk is relatively low. Due to the mutual coupling with medical device software, its security level is the same as that of medical device software, and the level of detail of registration application materials also depends on its security level .

External software environment update is an adaptive software update for medical device software , including product update (that is, replacing the ready-made software contained in the external software environment), version update, patch update, etc. Different external software environment updates have different effects on medical device software. Usually patch updates and version updates compatible with medical device software are minor software updates, while product updates and version updates to solve the incompatibility with medical device software are updated. It is a major software update, and it also follows the principle of high risk. Therefore, the registration applicant needs to carry out the impact assessment of the update of the external software environment according to the corresponding process .

5. Quality management software

(1) Main concepts

Quality management software refers to the application software used for the quality management of medical devices, non-medical device software, and does not need to be registered. Quality management software and medical device software have similarities at the technical level, so the confirmation requirements of quality management software can be considered with reference to the relevant requirements of medical device software.

Quality management software can be divided into quasi-independent software and quasi-software components with reference to medical device software. The former includes software used for design and development, software used for process management, etc., and the latter includes software included in production equipment and software included in inspection equipment.

Most of the quality management software is obtained through procurement, especially software components, which can be regarded as finished software, mainly used in full use, and partially used if secondary development; a small number of self-developed, mainly independent software, can be regarded as independent software. Research software. Therefore, the confirmation of quality management software can usually refer to the confirmation requirements of finished software and self-developed software.

The quality management software also needs the support of the external software environment (including system software, general application software, general middleware, support software) to run normally, so its confirmation also needs to consider the evaluation requirements of the external software environment.

( 2 ) Quality management software confirmation considerations

Quality management software confirmation is based on software functions, and comprehensively considers requirements such as requirements analysis, acceptance testing, and maintenance planning. Among them, the requirement analysis considers the functions, performance, interface and other requirements of the software, and evaluates the candidate software to determine the target object. The acceptance test is implemented based on the external software environment to ensure that the software can meet the needs of use, and the known remaining defects of the software and the impact of unused software functions on the quality of the medical device are acceptable. Maintenance plans take into account requirements related to software updates, especially maintenance plans for corrective software updates (ie, software defect fixes).

The evaluation requirements of the external software environment of the quality management software can refer to the corresponding requirements of the self-developed software. The quality management software update should be re-confirmed, and its external software environment update should also carry out impact assessment.

Six , medical device software life cycle process

The software life cycle process mainly includes software development process, software maintenance process, software risk management process, software configuration management process, and software defect management process .

The software development process includes software development planning, software requirements analysis, software design, software coding, software verification, software validation , software release and other activities .

software maintenance process is applicable to post-release enhanced software updates, including software update needs assessment, software update planning, software update implementation , software verification, software confirmation, software release, and user notification .

software risk management process includes activities such as risk analysis, risk evaluation, risk control , and comprehensive residual risk evaluation . It is implemented based on the medical device risk management process, and common risk management methods for medical devices can be used.

software configuration management process includes activities such as configuration item identification, change control, and configuration status recording . Software version control is performed based on software version naming rules, which can be implemented using configuration management tools or common office software. The software version naming rules specify the number of digits, range, and meaning of the fields. The meanings of the fields are clear, unambiguous and non-contradictory. It covers all types of software updates from self-developed software, ready-made software, and network security. It can distinguish between major software updates and minor software updates , ensuring that The version change of the software update conforms to the requirements of the software version naming rules [[8]](#footnote-8).

software defect management process is applicable to corrective software updates before and after release, including software defect assessment, software defect repair, regression testing and other activities , which can be implemented using defect management tools or common office software.

The software development process and software maintenance process are related before and after. The software risk management process, software configuration management process, and software defect management process run through the software development process and software maintenance process.

At the same time, software traceability analysis is also one of the important processes in the software life cycle process , which also runs through the software development process and software maintenance process , and can be implemented using traceability analysis tools or common office software.

In addition, the software life cycle process also needs to consider the requirements related to off-the-shelf software and network security. Off-the-shelf software considers the requirements of procurement control, design and development control, etc. The use of outsourced software requires a quality agreement with the supplier; the use of legacy software requires evaluation of existing documents, post-market use problems (including adverse events, recalls), etc. Use of open source software requires Follow the corresponding open source license agreement. The integration of network security design and development and software design and development can be implemented based on the requirements of network security capacity building, taking into account the requirements for emergency response to network security incidents.

The requirements of software life cycle process quality assurance activities should match the software security level. The higher the software risk level, the stricter the quality control requirements.

Agile development is special, and it is necessary to strengthen control requirements such as software updates, documents and records.

software life cycle process (including ready-made software and network security), please refer to the Quality Management Specification for the Production of Independent Software for Medical Devices and its on-site inspection guidelines [[9]](#footnote-9).

7. Technical considerations

(1) Registration unit and detection unit

1. Registration unit division principle

(1) Independent software

The independent software registration unit is divided into management categories, intended uses, and functional modules.

Independent software of different management categories is regarded as different registration units. If it is technically impossible to split , it can be used as a registration unit and applied for registration according to a higher management category .

Independent software with different intended uses, as different registration units, can be divided into assisted decision-making and non-assisted decision-making according to the intended use, and each category can be subdivided into treatment, diagnosis, monitoring , screening and other situations .

functional modules contained in each registration unit must be moderate. According to the type of functional modules , it can be divided into platform function software and specific function software [[10]](#footnote-10), among which platform function software provides basic functions and shared functions as a software platform, and supports multi - modal data (such as medical images , physiological parameters, in vitro diagnosis, etc. data); The functional software runs on the platform functional software and provides specific functions, supports single -modal data or uses multi-modal data to achieve a specific intended use.

For example, a PACS contains dozens of individual function modules, including auxiliary decision - making function modules, which need to be split into a platform function software and multiple specific function software, of which the auxiliary decision-making function module is used as a single registration unit.

(2) Software components

software component registration unit is the same as the medical device to which it belongs , and medical devices with and without software components are regarded as different registration units . Special purpose stand-alone software is regarded as a software component in the same registration unit as a software component.

2. Detection unit division principle

The testing unit refers to the representative products used for testing in the same registration unit.

(1) Independent software

In principle, the independent software detection unit is the same as the registration unit , but if there are multiple operating environments or multiple release versions, each incompatible operating environment (including cloud computing) or each release version that does not cover each other must be a detection unit.

If the core functions of the software are the same but the types of core algorithms are different, the core functions corresponding to each type of core algorithm need to be detected (the detection object is the core function rather than the core algorithm). For example, the core algorithms used in the image segmentation function include conventional image processing algorithms and artificial intelligence algorithms, and the image segmentation functions based on these two types of algorithms need to be detected.

(2) Software components

The software component testing unit is in principle the same as the medical device to which it belongs , but if the medical device contains multiple software components or software components of multiple released versions , each software component or software component of each released version must be regarded as a testing unit. Unless the detection unit can completely cover the entire situation of the registration unit . Similarly, if the core functions of the software are the same but the core algorithm types are different, the core functions corresponding to each type of core algorithm need to be tested .

Dedicated independent software is regarded as the detection unit of a software component in principle the same as that of a software component, but if there are multiple operating environments, each incompatible operating environment (including cloud computing) must be used as a detection unit.

(2) Basic principles of clinical evaluation

This guideline only specifies the basic principles [[11]](#footnote-11)of clinical evaluation of medical device software. For specific requirements, please refer to the relevant guidelines for clinical evaluation of medical devices.

1. Standalone software

Independent software usually conducts clinical evaluation based on software function, and can conduct clinical evaluation based on software algorithm if necessary. The clinical evaluation needs to be comprehensively considered in combination with the expected use and maturity of the independent software, and similar software functions contained in the listed medical devices can be selected for comparison of the same variety of medical devices.

Non-assisted decision-making software functions are based on core functions to compare medical devices of the same type. New core algorithms, core functions, and intended uses all require clinical evaluation in principle . Simple operation and simple process optimization software functions can be evaluated by non-clinical evidence.

The auxiliary decision-making software functions are based on the core algorithm to compare the medical devices of the same variety, and the clinical evidence of the selected medical devices of the same variety must be based on clinical trials (including retrospective studies, the same below) in principle. New core algorithms, core functions, and intended uses all require clinical trials in principle. If a clinical trial adopts an active control design, independent software with the same intended use and equivalent core algorithms or core functions can be selected for comparison.

2. Software Components

software components is usually carried out with the overall evaluation of the medical device. The clinical evaluation of the post-processing function can refer to the requirements of independent software, or it can also be evaluated as a whole with the medical device it belongs to.

Dedicated stand-alone software is regarded as the clinical evaluation of the software component and the post - processing functional requirements of the software component are the same.

(3) Network security

Medical device network security needs to comprehensively consider information security and data security from the perspective of network security. If medical device software has one or more of the three functions of electronic data exchange, remote access and control, and user access , network security issues must be considered. For specific requirements, please refer to the relevant guidelines for medical device network security.

(4) Cloud computing[[12]](#footnote-12)

The application of cloud computing in the medical device industry is increasing. Although it has the advantages of reducing informatization costs, reducing duplication of construction, improving resource utilization, increasing business flexibility, and improving service professionalism, there is also a weakening of users' ability to control data. , service sustainability is reduced, data ownership is challenged, data protection is difficult, data residue is difficult to handle, unclear responsibilities between users and cloud service providers, jurisdiction issues, and cybersecurity threats. Therefore, registration applicants need to balance the use of The benefits and risks of cloud computing .

Cloud computing is regarded as ready-made software, and cloud service providers are regarded as medical device suppliers. Therefore , registration applicants can refer to the relevant requirements of off-the -shelf software and medical device suppliers , and consider cloud computing demand analysis, risk management, verification and validation, and maintenance plans. and other activity requirements.

The demand analysis needs to consider the technical characteristics of cloud computing and the selection of cloud service providers . The technical characteristics of cloud computing include service mode , deployment mode, configuration, core function, data interface, network security assurance, etc. The service mode is divided into software as a service (SaaS), platform as a service ( PaaS ) , infrastructure as a service ( IaaS ), the deployment mode is divided into private cloud, public cloud, and hybrid cloud . The configuration includes requirements such as computing resources and supporting software functions. The core functions include data storage , data processing, data analysis and other functions. The data interface considers transmission protocols and storage formats. Other requirements, network security assurance considers technical measures such as data anonymity, data encryption, data transmission verification , and security configuration . At the same time, the selection of cloud service providers should be considered based on factors such as cloud computing service qualifications and service agreements . The cloud computing service agreement should specify network security assurance, patient data and privacy protection and other responsibility requirements.

assurance of cloud computing . Verification and Validation Carry out the verification and validation of medical device software based on the cloud computing environment to ensure that the cloud computing meets the requirements for use and that the risks of known remaining defects are acceptable. The maintenance plan considers the maintenance plan for cloud computing updates, redefines the verification and confirmation of medical device software, and specifies the lossless data migration plan for the termination of cloud computing services.

If cloud computing is used, the registration applicant needs to reflect it in the relevant provisions of the self-developed software research report and the external software environment assessment report. For specific requirements, please refer to Chapter 8. If you build a cloud computing platform by yourself, the registration applicant should follow the relevant regulations of cloud service providers and submit relevant research materials with reference to the requirements of self-developed software.

(5) Mobile computing

Medical device software is a mobile medical device if it runs on a mobile computing terminal (including medical terminal and general terminal) for personal use. At this time, the technical characteristics and risks of mobile computing terminals should be comprehensively considered. For specific requirements, please refer to the relevant guidelines for mobile medical devices.

(6) Artificial intelligence

Medical device software is an artificial intelligence medical device if it uses artificial intelligence technology to achieve its intended use (i.e. medical use). At this time, it is necessary to comprehensively consider the technical characteristics and risks of artificial intelligence algorithms. For specific requirements, please refer to the relevant guidelines for artificial intelligence medical devices.

(7) Human factors and availability

It is recommended to strengthen the human factors design of the user interface of medical device software to improve usability and reduce the risk of user misuse to an acceptable level. For specific requirements, please refer to the relevant guidelines for human factors design of medical devices .

(8) Interoperability

Interoperability (aka interoperability) refers to the ability of a medical device to exchange and use information with other medical devices or general purpose equipment through electronic interfaces. The electronic interface includes a hardware interface and a software interface, and the information includes but is not limited to medical images, physiological parameters, in vitro diagnosis and other data and control instructions.

Interoperability focuses on the interface design and risks of medical device software. The interface includes internal interface and external interface. The former refers to the interface between software modules, and the latter refers to the interface for users to call. The independent parts of the split medical device The internal interface of the server is regarded as the external interface, such as the internal interface of the server and the client, the master and the slave. From the user's point of view, the software interface refers to the external interface unless otherwise specified.

Registration applicants need to consider the requirements for software interface needs analysis, risk management, verification and validation, maintenance planning and other activities, as well as the design requirements for instructions and labels.

Requirement analysis is based on the expected users of the software interface (such as medical personnel, patients, maintenance personnel), usage scenarios, and intended uses to clarify its technical characteristics and usage restrictions. Among them, the software interface includes application program interface for users to call, data interface ( including transmission protocol, storage format, such as DICOM, HL7, JPG, PNG, private protocol and format ), product interface ( other medical device independent software that can be used in combination) , medical device hardware products ) . Technical features include but are not limited to the requirements for connection objects, information content, communication protocols, performance indicators, and network security assurance. Usage restrictions take into account the expected range of users, connectivity requirements for each software interface.

Risk management is implemented based on the intended users of the software interface , usage scenarios, intended use and technical characteristics, and usage restrictions, and clarifies fault response measures. Verification and validation shall ensure that the software interface meets the design requirements and the risks of known remaining defects are acceptable. The maintenance plan considers the update requirements of the software interface, and the software interface update also needs to be re-verified and confirmed.

The manual describes the intended users, usage scenarios, intended uses, technical features, usage restrictions, and fault countermeasures of each software interface item by item. According to the risk control requirements, the labels clarify the technical characteristics and usage restrictions of key software interfaces.

The registration applicant can submit a separate interoperability research material, including basic information, requirements specification, risk management, verification and confirmation, and maintenance plan ; the software interface requirements can also be reflected in the relevant clauses of the self-developed software research report, and the specific requirements See Chapter 8 for details.

( 9 ) Measurement function

Measurement functions (also known as quantitative and quantitative functions) can be divided into graphic measurement functions and objective physical measurement functions. The former indirectly reflects the measurement results of objective things based on graphics, and the latter directly reflects the measurement results of objective things. Regardless of the measurement function, it is necessary to combine the measurement error, uncertainty and other factors to clarify the measurement accuracy indicators, such as linearity, precision, repeatability, reproducibility, range limit, display error, etc.

Registration applicants are required to provide research data on the measurement accuracy and inform users in the instructions. In addition, the objective physical measurement needs to specify the accuracy index in the product technical requirements, and the graphic measurement needs to provide warning information about the measurement accuracy in the manual.

(10) Remote access and control

For remote access and control (including remote maintenance and upgrade) functions, it is necessary to clarify the performance index requirements and network security requirements of related functions. For specific requirements, please refer to the relevant guidelines for medical device network security.

Registration applicants need to provide relevant research materials in software research materials and network security research materials, specify performance index requirements in product technical requirements, and inform users of the use requirements (including network security) of corresponding functions in the manual .

(11) General computing platform

The general computing platform itself does not belong to the object of supervision, and its impact on medical devices needs to be considered from the perspective of risk management. The specific requirements depend on whether it is a product structure of medical devices.

Independent software and dedicated independent software are regarded as software components, and the product structure of medical devices does not contain a general-purpose computing platform. Users are informed in the manual that the general-purpose computing platform must meet the safety requirements of information technology equipment (including electromagnetic compatibility), and list the requirements. List of standards complied with.

For externally controlled software components, the product structure of medical devices includes a general computing platform, and in "Other Research Materials", provide evidence that the general computing platform satisfies the safety requirements of information technology equipment (including electromagnetic compatibility), such as the relevant standards for automatic Inspection report, inspection report or related certification documents.

( 12 ) Non-medical device functions

If the medical device software can technically split the functions of non-medical devices, that is, adopt modular design to distinguish the functions of medical devices and non-medical devices, the product structure composition shall not include functional modules of non-medical devices . Functions should be deleted or marked as non-medical device functions, and product technical requirements should not contain non-medical device functions.

If the medical device software cannot technically separate the non-medical device functions, the non -medical device functions need to be considered as a part of its own as a whole, focusing on the impact and risks of the non-medical device functions on the medical device software. The registration applicant needs to consider the software life cycle process quality control requirements of non-medical device functions under the overall framework of medical device software design and development. In principle, it can be handled according to the requirements that the software security level is minor. The research materials of medical device software cover the functions of non-medical devices, and the instructions shall indicate the functions of non-medical devices.

( 13 ) Implant product design software

Some implantable medical devices do not contain medical device software, but their safety and effectiveness are significantly limited by the output of product design software, such as active implant design software (such as programmable logic control programming software, integrated circuit design software ), personalized passive implants (such as orthopedics, dental additive manufacturing prosthesis) design software, etc.

At the same time, some implantable medical devices need to use modeling simulation software for safety and effectiveness verification, such as magnetic resonance environment safety simulation software, computational fluid dynamics simulation software, finite element calculation simulation software, etc.

Therefore, from the perspective of medical device safety and effectiveness evaluation, these two types of implant product design software also need to submit software research materials when applying for implantable medical device registration. Due to the high risk of implantable medical devices, the implant product design software belongs to the quality management software, so the software research materials can be submitted with reference to the requirements of the finished software and self-developed software of the serious level. For specific requirements, please refer to Chapter 8.

(14) Term of use

The use period of independent software is the software life cycle time limit, which is determined by commercial factors and does not need to provide verification materials.

The term of use of a software component is the same as that of the medical device to which it belongs, and does not need to be represented separately. Dedicated independent software is regarded as a software component with the same service period requirements as independent software, which is reflected in the research data on the service period of the medical device.

(15) Exception handling

Exception handling is used to deal with abnormal conditions in the software, and notify users of software exception information in time, so as to take risk control measures to ensure safety and effectiveness. Registration applicants need to strengthen the design work of exception handling in the process of medical device software design and development, especially in the use scenarios such as surgery and first aid.

(16) Functional safety and software reliability

Taking into account the actual situation of the industry, functional safety and software reliability are not required for the time being, and will be taken into consideration when the time is ripe. It is recommended that the registration applicant refer to the relevant standards to strengthen the design work of functional safety and software reliability. If the corresponding work has been carried out, it can be provided in the software research materials.

(17) Implementation requirements of GB/T 25000.51

GB/T 25000.51 applies to medical device software, of which "requirements for product description" and "requirements for user documentation" apply to manuals, "software quality requirements" apply to the software itself, and "use quality" does not apply.

The registration applicant shall submit the GB/T 25000.51 self-test report in the software research materials, or submit the self-test report or inspection report instead of the self-test report, the same below.

(18) Imported medical device software

For imported medical device software, the certification documents or supporting materials that the declared software is approved for listing in the country of origin (that is, the country/region where the registration or production address is located) should be provided , indicating the complete version of the software .

If there are differences between China and foreign countries in the imported medical device software, such as language differences, deletion of software functions, reduction of application scope, etc., it is necessary to submit the Chinese and foreign differences between the application software and the software approved for marketing in the country of origin in the software research materials.

Considering that imported medical device software may not be registered simultaneously in China, that is, the software has been registered multiple times in the country of origin but is registered once in China (including product registration and change registration). For product registration, the registration application materials must cover the entire contents of the software declared this time ; for change registration, the registration application materials must cover all the changed contents of the software from the previous registration in China to this application .

Eight , medical device software research materials

The medical device software research data framework is shown in Figure 5, including self-developed software and ready-made software (including ready-made software components and external software environment). The specific requirements are as follows.

(1) Self-developed software research report

The self-developed software research report is suitable for the initial release and re-release of self-developed software. The content framework is detailed in Table 1, including basic information, implementation process, core functions, and conclusions. The degree of detail depends on the software security level. Inapplicable content details For reasons, all attachments must indicate the full version of the software .

Figure 5 Medical device software research data frame

1. Basic information

(1) Software identification

Specify the name, model specification, release version , HASH value (such as MD5 value) of the software, as well as the registration applicant , design and development address.

(2) Security level

Clarify the security level of the software and detail the reasons for the decision .

(3) Structure function

Based on the software design specification document, the software architecture diagram , user interface diagram and main interface diagram are provided. The architecture diagram distinguishes medical device software, necessary software, and external software environment. The user interface diagram clearly defines the main interface, first-level and Interrelationships of secondary user interfaces .

According to the architecture diagram , the functions, purposes, interfaces, necessary software , cloud computing, etc. of the illustrated software modules (that is, component modules ) are described in detail , and the security level of each component module is indicated . According to the user interface relationship diagram (if applicable) , the functions, purposes , and interfaces of the software modules (that is, functional modules ) are described in detail , and the layout, options, and functions of the main interface are described in detail according to the main interface diagram (if applicable) .

If applicable, the optional and module versions shall be indicated for the constituent modules and functional modules . The interface includes the application program interface, data interface, and product interface for users to call, and the expected users, usage scenarios, expected uses, technical characteristics, usage restrictions, and fault countermeasures of each interface are described item by item .

(4) Physical topology

(including cloud computing) based on the software design specification document , and describe the physical connection relationship between software/component modules, general computing platforms , medical device hardware products /components , and necessary software according to the physical topology diagram , including All peripherals .

(5) Operating environment

Identify the typical operating environment required for the normal operation of the software (software module) , including hardware configuration, external software environment, prerequisite software, and network conditions. Among them , the hardware configuration includes requirements for processors, memory, peripheral devices, etc .; the external software environment includes system software, general application software, general middleware, and support software, indicate the name, full version, and patch version of all software. Version" rather than "above version" and "higher version"; if applicable, the required software should specify the name, model specification, release version, registrant; network conditions include network architecture (such as BS architecture, CS architecture, hybrid architecture) , Network type (such as wide area network, local area network, personal area network) , network bandwidth and other requirements.

If cloud computing is used, specify the name, service mode , deployment mode, configuration of cloud computing, as well as the name, residence, and service qualification of the cloud service provider.

(6) Registration history

Clarify the registration of software in China and the country of origin, and list the date, release version, and management category of previous registrations. The software component specifies the registration status of the medical device to which it belongs. In addition, the registration information of the software in other major countries and regions can also be provided.

2. Implementation process

(1) Development overview

An overview of the software development methods (such as process-oriented, object-oriented, agile development, etc.), programming languages, development and testing environments (including hardware and software equipment, development and testing tools, network conditions , and cloud computing ), among which the development and testing tools have clear names and full versions. , developers ; provide an estimate of the total number of development and testing personnel, duration, workload (man-months), and the total number of lines of code .

(2) Risk management

Provide a software risk management flow chart, detailing the specific activities of the software risk management process according to the flow chart . Provide software risk analysis reports and risk management reports , covering functions, performance, interfaces, operating environment, necessary software, cloud computing, etc. original document .

Software components provide risk management documentation for the medical device to which they belong , and indicate where the software components are located .

(3) Requirements specification

Provide software requirements specification documents, clarify the functions, performance, interface, operating environment, necessary software, cloud computing and other requirements of the software, and attach the original documents formed by the software development .

product requirement specification document of the medical device to which it belongs can be provided , and the location of the software component shall be indicated .

(4) Life cycle

the specific activities of the software development process, software maintenance process, and software configuration management process .

Intermediate level: Provide software development, software maintenance, software configuration management flow charts, and detail the specific activities of the software development process, software maintenance process, and software configuration management process according to the flow chart.

Severity level: Provide software development, software maintenance, and software configuration management flow charts, and detail the specific activities of the software development process, software maintenance process, and software configuration management process according to the flow chart . Provides the software design history document set (DHF) index table, and attaches the software coding rules document.

In addition, the use of agile development requires document and record control documentation. Software life cycle processes and activities may also provide software life cycle process control program documentation or software life cycle process standard checklists to replace corresponding descriptions.

(5) Verification and confirmation

Minor level: Provides planning and reporting of system tests and user tests.

Intermediate level: Outline the quality assurance activities of the software development process , and provide planning and reporting of system testing and user testing.

Severity level: Provide a software development quality assurance flow chart, detail the specific quality assurance activities of the software development process according to the flow chart , and provide plans and reports for integration testing, system testing, and user testing.

In addition, the test plan and report cover the functions, performance, interface, operating environment, necessary software, cloud computing, etc. of the software, and the original documents formed by the software development are attached . The software development process quality assurance activities may also provide software development quality assurance plan documents in lieu of corresponding descriptions.

(6) Traceability analysis

Provide a software traceability analysis flow chart, and detail the specific activities of the software traceability analysis process according to the flow chart . Provide software traceability analysis report, summarize and list the software requirement specification document, software design specification document, source code (clear software unit name is enough), software test report, software risk analysis report, and attach software development The original document formed .

(7) Defect management

Minor level: Outline the software defect management process, clarify the total number of known software defects and the number of remaining defects.

Moderate and Severe: Provide a software defect management flow chart, detail the specific activities of the software defect management process according to the flow chart; clarify the total number of known software defects and the number of remaining defects, and list the content, impact, and risk of known remaining software defects. Make sure the risks are acceptable. Additional files can be attached for the known remaining defects of the software .

( 8) Update history

the full version, date, and type of software updates since the previous registration to the current application .

the full version, date, type and specific content of all software updates since the previous registration to this application .

the full version, date, type, and specific content of all software updates since the first registration to this application .

In addition, if software modules (including medical middleware) are separately version-controlled, the version naming rules should also be provided, and the relationship with the software version naming rules should be clearly defined; the version naming rules for software and software modules should be consistent with the quality management system. . Software update types indicate major or minor updates. The initial release lists the previous software updates in the software development stage. Software update history can be paid separately .

3. Core functions

Minor level: The name of the core function of the software, the core algorithm used, and the expected use are listed based on the manual. The new core function, core algorithm, and expected use need to be specified.

Moderate and Severe: Based on the manual, the name of the software 's core functions, the core algorithms used, and the intended use should be indicated. The new core functions, core algorithms, and intended uses should be indicated , and corresponding safety and effectiveness research materials should be provided . Among them, the new algorithm provides algorithm research reports, which usually include basic algorithm information, algorithm risk management, algorithm requirements specifications, algorithm quality control requirements, algorithm verification and confirmation, algorithm traceability analysis , conclusions , etc. The measurement function provides research data on measurement accuracy. Data resources (such as reference databases) clarify the types of data, the sample size and data distribution of each type of data.

4 Conclusion

Briefly describe the standardization of the software implementation process and the correctness of core functions, determine whether the safety and effectiveness of the software meet the requirements , and whether the benefits outweigh the risks .

(2) Self-developed software update research report

The self-developed software update research report is suitable for the re-release of self-developed software, including research reports such as perfect update, adaptive update, and correction update .

The perfect update research report is suitable for major or slightly perfect updates of self-developed software , or the merger of adaptive updates and correction updates . The content framework is shown in Table 2 and will not be repeated .

The adaptive update research report is suitable for major or minor adaptive updates of self-developed software , or merged and corrective updates, but no complete updates have occurred. The content includes software identification, security level, operating environment, risk management, and requirements specifications. , Life Cycle, Verification and Confirmation, Traceability Analysis, Defect Management, Update History, Conclusion , see Table 2 for specific requirements .

Corrective update research report is applicable to the situation where only corrective update occurs in self-developed software. The content includes software identification, security level, risk management, verification and confirmation, traceability analysis, defect management, update history, and conclusion . The specific requirements are detailed . See Table 2 for the corresponding description .

Considering the cumulative effect of software updates, the software update research report needs to cover the entire contents of the software update of the medical device software since the previous registration (except for the renewal registration).

Table 1 Self-developed software research report framework

|  |  |
| --- | --- |
| **Reporting terms** | **software security level** |
| **slight** | **medium** | **serious** |
| **Basic Information** | software identification | Specify the name, model specification, release version, HASH value, registration applicant, design and development address of the software |
| security level | Clarify the security level of the software and detail the reasons for the decision |
| structure function | According to the architecture diagram, the user interface diagram and the main interface diagram, the function, purpose, interface and necessary software of the component modules and functional modules are described in detail |
| physical topology | Detail the physical connection relationship of software/component modules, general computing platforms, medical device hardware products/components, and necessary software according to the physical topology diagram |
| Operating environment | Identify the typical operating environment required for the normal operation of the software, including hardware configuration, external software environment, prerequisite software, and network conditions |
| Registration History | Clarify the registration status of the software in China and the country of origin |
| **Implementation process** | Development overview | Provide an overview of the development methods, programming languages, and development and testing environments used in the software, and provide an estimate of the number of people, duration, workload, and lines of code used for development and testing |
| Risk Management | According to the flow chart, the software risk management process is detailed, and the software risk analysis report and risk management report are provided. |
| Requirements specification | Provide software requirements specification documents |
| life cycle | Overview of software development process, software maintenance process, software configuration management process | Detail the software development process, software maintenance process, and software configuration management process according to the flow chart | Detail the software development process, software maintenance process, and software configuration management process according to the flow chart, and provide the index table of software design history document set and software coding rule document |
| Verification and Confirmation | Provide system testing, user testing planning and reporting | Outline the quality assurance activities of the software development process, and provide plans and reports for system testing and user testing | Detail the quality assurance activities of the software development process according to the flowchart, and provide plans and reports for integration testing, system testing, and user testing |
| Traceability Analysis | Detail the software traceability analysis process according to the flow chart, and provide software traceability analysis report |
| defect management | Outline the software defect management process and clarify the total number of known software defects and the number of remaining defects | Describe the software defect management process in detail according to the flowchart, clarify the total number of known software defects and the number of remaining defects, and list the content, impact, and risk of known remaining defects to ensure that the risks are acceptable |
| update history | Clarify the software version naming rules, release version, and complete version, and list the complete version, date, and type of software updates since the previous registration to this application. | Clarify the software version naming rules, release version, and complete version, and list the complete version, date, type, and specific content of all software updates since the previous registration to this application. | Specify the software version naming rules, release version, and complete version, and list the complete version, date, type, and specific content of all software updates since the first registration to this application. |
| **Core functions** | List the name of the core function of the software, the core algorithm used, the intended use, and the type | List the name of the core function of the software, the core algorithm used, the intended use, and indicate the type. For new core functions, core algorithms, and intended uses, safety and effectiveness research materials must be provided |
| **in conclusion** | Briefly outline the standardization of the software implementation process and the correctness of core functions, and determine whether the safety and effectiveness of the software meet the requirements |

Table 2 Self-developed software perfect update research report framework

|  |  |
| --- | --- |
| **Reporting terms** | **software security level** |
| **slight** | **medium** | **serious** |
| **Basic Information** | software identification | Clarify the software status of this application and detail the changes |
| security level | Clarify the situation of this application software, if change the detailed reasons and submit registration application materials according to the new security level |
| structure function | Clarify the software status of this application and detail the changes |
| physical topology | Clarify the software status of this application and detail the changes |
| Operating environment | Clarify the software status of this application and detail the changes |
| Registration History | Clarify the software status of this application |
| **Implementation process** | Development overview | Clarify the software status of this application |
| Risk Management | According to the flow chart, the software risk management process is detailed, and the risk analysis report and risk management summary report of the software update part are provided. |
| Requirements specification | Provides requirements specification document for the software update section |
| life cycle | Outline the software maintenance process, software configuration management process | Detail the software maintenance process and software configuration management process according to the flow chart | Describe the software maintenance process and software configuration management process in detail according to the flow chart, provide the design history document set index table of the software update part, and the software coding rule document |
| Verification and Confirmation | Provide regression test plans and reports | Outline software maintenance process quality assurance activities and provide regression testing plans and reports | Detail the software maintenance process quality assurance activities according to the flowchart, and provide regression test plans and reports |
| Traceability Analysis | Detail the software traceability analysis process according to the flow chart, and provide the traceability analysis report of the software update part |
| defect management | Outline the software defect management process, and clarify the total number of known defects and remaining defects of the software declared this time | software defect management process in detail according to the flowchart, clarify the total number of known defects and the number of remaining defects in the software declared this time , and list the content, impact and risk of known remaining defects to ensure that the risks are acceptable |
| update history | Clarify the software version naming rules, release version, and complete version, and list the complete version, date, and type of software updates since the previous registration to this application. | Clarify the software version naming rules, release version, and complete version, and list the complete version, date, type, and specific content of all software updates since the previous registration to this application. | Specify the software version naming rules, release version, and complete version, and list the complete version, date, type, and specific content of all software updates since the first registration to this application. |
| **Core functions** | List the core functionality of the software update section | List the core functions of the software update part. The new core functions, core algorithms, and intended uses need to provide safety and effectiveness research materials. |
| **in conclusion** | Briefly describe the standardization of the software update implementation process and the correctness of the corresponding core functions, and determine whether the security and effectiveness of the software declared this time meet the requirements |

(3) Research data on ready-made software

1. Off-the-shelf software components research material

(1) Partial usage

components are explained in the applicable terms of the self-developed software research report .

Applicable terms include software identification, security level, structural functions, operating environment, risk management, requirements specification, life cycle, verification and validation, traceability analysis, defect management, update history, core functions, conclusions, and other terms as applicable. can be explained.

At this time, if the ready-made software components are updated, the research report of perfect update will explain the changes of the ready-made software components on the basis of the research report of self - developed software . The requirements are the same as the self-developed software.

(2) All usage methods

For all usage methods, the research data requirements for legacy software, finished software, and outsourced software are slightly different, and the ready-made software component research report and its type determination basis are submitted separately .

The terms of the ready-made software component research report are the same as those of the self-developed software research report, but they need to be explained based on the security level of the ready-made software components (in this case, medical device software). The applicable terms include at least software identification, security level, operating environment, risk Management, Requirements Specification, Verification and Validation, Defect Management, Core Functions, Conclusions, Not Applicable Clauses detail the reasons. In the verification and validation of legacy software , an evaluation report on post-market use problems (including adverse events and recalls) must also be submitted .

Type decisions are used to certify the type of off-the-shelf software components . Legacy software provides product registration certificate information or product listing certification materials before the full implementation date of the "Good Manufacturing Practice for Medical Devices" ( January 1, 2018) . The finished software shall provide supporting materials such as outsourcing contracts , and if it has been listed in China, provide product registration certificate information . Outsourcing software provides supporting materials such as outsourcing contracts or agreements .

At this time, if the ready-made software components are updated, the perfect update research report shall be based on the ready-made software component research report, and explain the changes of the ready-made software components, and explain the reasons for the non-applicable clauses; The research software is the same.

2. External Software Environment Assessment Report

external software environment evaluation report is used for the evaluation of the external software environment (including cloud computing) of medical device software , and is suitable for the initial release and re-release of medical device software . The content framework is shown in Table 3, including security level, software identification, functional use , operating environment, risk management, acceptance management, maintenance plan , conclusions, the level of detail depends on the software security level.

(1) Security level

According to the security level of the medical device software, the security level of the external software environment is clarified.

(2) Software identification

According to the four types of system software, application software, middleware, and support software, the name, full version, patch version, release date, and supplier of all ready-made software contained in the external software environment are classified and described.

(3) Functional use

According to the four types of system software, application software, middleware and support software, the functions, uses, relationship with medical device software, usage restrictions, and selection basis of all ready-made software contained in the external software environment are classified and described.

(4) Operating environment

According to the four types of system software, application software, middleware and support software, the operating environment of all ready-made software included in the external software environment is classified and described, and the basis for determining the operating environment of medical device software is considered in combination with compatibility .

(5) Risk management

Provide the risk analysis report of all ready-made software contained in the external software environment, and attach the original documents formed by the software development . The risk analysis report of medical device software can be provided , and the location of the external software environment shall be indicated .

(6) Acceptance management

Minor level: Outlines activities related to the acceptance management process for the external software environment.

Intermediate level: The activities related to the external software environment acceptance management process are detailed according to the flowchart.

Severity level: According to the flowchart, the relevant activities of the external software environment acceptance management process are detailed, and the external software environment compatibility test plan and report are provided, and the original documents formed by the software development are attached .

( 7) Maintenance plan

Minor level: Outline the activities related to the update management process of the external software environment, including patch updates, version updates, and product updates.

Medium level: The activities related to the update management process of the external software environment are detailed according to the flowchart, including patch update, version update, and product update.

Severity level: According to the flow chart, details related activities of the external software environment update management process, including patch update, version update, and product update; provide a follow-up maintenance plan for ready-made software outage, that is, after the ready-made software supplier stops after-sales service, the registration applicant will Maintenance solutions for off-the-shelf software , such as lossless data migration solutions after cloud computing services are terminated.

(8) Conclusion

Briefly describe whether the quality of all ready-made software contained in the external software environment meets the requirements.

Table 3 External software environment assessment report framework

|  |  |
| --- | --- |
| **Reporting terms** | **software security level** |
| **slight** | **medium** | **serious** |
| security level | Determine the security level of the external software environment according to the security level of the medical device software |
| software identification | Category describes the name, full version, patch version, release date, vendor of each off-the-shelf software |
| Functional use | Classification describes the functions, uses, relationship with medical device software, usage restrictions, and selection basis of each ready-made software |
| Operating environment | Classify and describe the operating environment of each ready-made software, and clarify the basis for determining the operating environment of medical device software |
| Risk Management | Provide risk analysis reports for each ready-made software |
| Acceptance management | Outline the external software environment acceptance management process | The external software environment acceptance management process is described in detail according to the flow chart | According to the flow chart, the external software environment acceptance management process is detailed, and the compatibility test plan and report are provided. |
| maintenance plan | Outline the external software environment update management process | Detail the external software environment update management process according to the flow chart | According to the flowchart, the external software environment update management process is detailed, and the follow-up maintenance plan for ready-made software is provided. |
| in conclusion | Briefly describe whether the quality of all ready-made software contained in the external software environment meets the requirements |

external software environment is updated, corresponding evaluation work should be carried out according to its security level, and the external software environment evaluation report of the medical device software should be updated for subsequent system verification or registration change.

9. Supplementary Instructions for Registration Application Materials

meeting the requirements of medical device registration application materials and other document requirements, the registration application materials shall focus on the following requirements.

(1) Product registration

1. Application form information

(1) Independent software

product name should comply with the requirements of the independent software common name naming specification, and usually reflect the characteristic words such as input data, core function, and intended use.

The model specification indicates the software release version, and does not need to reflect the English abbreviation V of the version.

structure consists of clear delivery content and functional modules, in which the delivery content includes software program files such as software installation programs, authorization files, external software environment installation programs, etc. The functional modules include client side , server side ( if applicable) , cloud (if applicable), if applicable Applicable indicate optional , module version.

The scope of application is usually specified based on the intended use, usage scenarios, and core functions. If applicable, the scope of application of each functional module is described. At the same time, ensure the standardization of terms, distinguish between "analysis" and "measurement", "surgery simulation" and "surgery planning", and use "display" and "receive" instead of "browse" and "collect".

( 2) Software components

Software components generally do not need to be reflected in the information contained in the registration certificate. The software function name can refer to the independent software requirements. Structural composition to ensure the normative terminology, the use of "host" and "workstation" instead of "computer" and "computer". If there are auxiliary decision-making software functions, the structural composition (if applicable) and the scope of application shall be reflected.

In the case of dedicated independent software being regarded as a software component , the software name is the same as the independent software requirements, the structure composition is clear about the name, model specification , and release version of the software, and the scope of application reflects the functions of auxiliary decision-making software .

2. Software research materials

Submit self-developed software research report, external software environment assessment report (if applicable) and GB/T 25000.51 self-test report.

If using off-the -shelf software components , submit appropriate research materials according to their usage. The specific requirements of relevant research materials are detailed in Chapter 8 .

In addition, the submission of medical device product marketing materials in software research materials is encouraged. This material is only used as a reference material for the review to supplement product information, not as a review object, nor as a basis for review decision-making.

3. Product technical requirements

(1) Independent software

technical requirements for independent software products "Product Model/Specification and its division description" specify the name, model specification, release version , and version naming rules of the software. If there are separate versions and version naming rules for software modules (including medical middleware), they must be explained. .

"Performance indicators" include general requirements, special requirements , and safety requirements, among which general requirements are regulated according to the characteristics of software products , and inapplicable contents are explained in the chapter on technical requirements for non-clinical data products; special requirements comply with relevant product standards (such as radiotherapy planning software ), and safety requirements comply with relevant safety standards (such as alarms, radiation therapy).

"Appendix" provides architecture diagram , user interface diagram and main interface diagram, physical topology diagram and necessary notes .

Please refer to the appendix for the template of technical requirements for independent software products.

(2) Software components

software components are regulated in the product technical requirements of the medical device to which they belong , in which the "Product Model/Specification and its Classification Description" specifies the software name, model specification (if applicable) , release version, version naming rules, software modules (including medical intermediates ) file) If there is a separate version, the version naming rules should be explained.

"Performance Metrics" include the software's functionality, usage restrictions, interface, access control, operating environment (if applicable), performance efficiency (if applicable) and other requirements. Among them, the function specifies the outline of all the core functions (including safety functions) of the software , indicating optional and automatic functions, among which the objective physical measurement function specifies the measurement accuracy index, and the data resources (such as reference databases) specify the data types and each type of data. Sample size, if the core functions are the same but the core algorithm types are different, each type of core algorithm needs to be remarked; usage restrictions include user usage restrictions and technical restrictions; interfaces include application program interfaces, data interfaces, and product interfaces for users to call; access control is clear Software user identification method, user type and user access authority; operating environment and performance efficiency are applicable to externally controlled software components and dedicated independent software are regarded as software components, in which the operating environment (including cloud computing) clearly defines the typical configuration, including hardware Configuration, external software environment, network conditions, and performance efficiency specify the time characteristics of software to complete typical core functions in a typical operating environment, and specify resource utilization and capacity if applicable. The non-applicable content of the above clauses is explained in the non-clinical data.

For dedicated independent software as software components, in addition to the above software component requirements, an architecture diagram , a user interface relationship diagram and a diagram of the main interface, a physical topology diagram and necessary annotations should be provided in the "Appendix" .

4. Instructions

Instructions should comply with relevant regulations, normative documents, national standards, industry standards.

The manual shall reflect the functions of the software, usage restrictions, input and output data types, necessary software and hardware, maximum concurrent number, interfaces, access control, operating environment (if applicable), performance efficiency (if applicable) and other information, and specify the software release version.

Among them, the software functions include all core functions (including safety functions), indicating optional and automatic functions, among which the measurement function specifies the measurement accuracy index, the graphic measurement function also needs to provide warning information about the measurement accuracy, and the data resources specify the data Type and sample size of each type of data. The interface describes each intended user, usage scenario, intended use, technical features, usage restrictions, and fault countermeasures for each user to invoke the software interface. The operating environment (including cloud computing) and performance efficiency are applicable to independent software, externally controlled software components, and dedicated independent software as software components. For specific requirements, please refer to the preceding section.

If applicable, inform the user that the general computing platform meets the safety requirements of information technology equipment (including electromagnetic compatibility), and list the corresponding standards.

5. Product label sample (for independent software )

For physical delivery methods, product labels should comply with the appropriate regulations . For the online delivery method, submit a photo of the product online delivery page, and the product registration information on this page should comply with the corresponding regulations .

In addition, it is recommended to reflect product registration information in the software user interface such as "About" or "Help".

(2) Change of registration

1. Software research materials

medical device change registration should submit research materials on the impact of software changes on product safety and effectiveness according to the software update situation :

(1) Involving perfect software update: it is applicable to the situation of perfect update of self-developed software, or merger of adaptive update and correction type update. At this time, submit the self-developed software perfect update research report (or self-developed software research report) , external software environment assessment report (if applicable) and GB/T 25000.51 self-test report;

(2) Involving adaptive software update: it is applicable to the situation where self-developed software has adaptive update, or merged corrective update , but no perfect update has occurred. At this time, submit the self-developed software adaptive update research report (or self-developed software research report);

(3) Only corrective software update occurs: it is applicable to the situation where only corrective update occurs in self-developed software, and the self-developed software corrective update research report is submitted at this time ;

(4) No software update occurs: a statement of authenticity is issued , and it is clear that it is legally responsible for this .

If using off-the-shelf software components, submit appropriate research materials according to their usage. The specific requirements of relevant research materials are detailed in Chapter 8.

Where applicable, the submission of medical device product marketing materials in software research materials is encouraged. This material is only used as a reference material for the review to supplement product information, not as a review object, nor as a basis for review decision-making.

2. Product technical requirements

(1) Independent software

independent software products reflect the status of software updates, including "product model/specification and its division description", "performance index" , and "appendix".

(2) Software components

Software components reflect the software updates in the product technical requirements of the medical device to which they belong, including the software information of "product model/specification and its division description", and the software requirements of "performance index" .

Special purpose stand-alone software has the same requirements as software components to be considered software components.

3. Instructions

If applicable, submit a description of changes to the specification.

4. Product label sample (for independent software )

If applicable, submit a product label sample of the declared product and a description of its changes.

(3) Renewal of registration

Submission of software-related research materials is generally not required for renewal of registration. If applicable, submit relevant software research materials according to the requirements stated in the "Remarks" of the registration certificate.

Product technical requirements "Product model/specification and its division description" specify the name , model specification, release version, and version naming rules of the software . If the original registered product standard (or original product technical requirements) and its change comparison table do not reflect the above software information, which needs to be specified in the declaration of conformity.

The technical requirements "performance indicators" and "inspection methods" of independent software products do not need to be modified in accordance with the requirements of the appendix of this guideline, and should be consistent with the original registered product standards (or original product technical requirements) and their change comparison table, and the "performance indicators" Relevant clauses delete the information contained in the registration certificate.

10. References

[ 1 ] Former State Food and Drug Administration . Regulations on the Administration of Instructions and Labels of Medical Devices (Order No. 6 of the General Administration) [Z] , 2014.7

[ 2 ] Former State Food and Drug Administration . Rules for the Classification of Medical Devices (Order No. 15 of the General Administration) [Z] , 2015.7

[ 3 ] Former State Food and Drug Administration . Measures for the Administration of Medical Device Recalls (Order No. 29 of the General Administration) [Z] , 2017.1

[ 4 ] State Administration for Market Regulation . Administrative Measures for Monitoring and Re-evaluation of Adverse Events of Medical Devices (Order No. 1 of the General Administration) [Z] , 2018.8

[ 5 ] State Administration for Market Regulation. Administrative Measures for Registration and Filing of Medical Devices (Order No. 47 of the General Administration) [Z] , 2021.8

[ 6 ] Former State Food and Drug Administration . Good Manufacturing Practice for Medical Devices (Announcement No. 64 of 2014) [Z] , 2014.12

[ 7 ] State Drug Administration. Rules for the Unique Identification System of Medical Devices ( Announcement No. 66 of 2019 ) [ Z], 2019.8

[ 8 ] State Administration for Market Regulation. Requirements for medical device registration application materials and format of approval documents ( Announcement No. 121 in 2021 ) [Z] , 2021.9

[ 9 ] State Drug Administration. Regulations on the Administration of Self-inspection of Medical Device Registration (Announcement No. 126 of 2021) [Z] , 2021.10

[ 10 ] Former State Food and Drug Administration . Guiding Principles for Technical Review of Medical Device Software Registration (No. 50 in 2015 ) [Z] , 2015.8

[ 11 ] Former State Food and Drug Administration . Guidelines for Technical Review of Medical Image Storage and Transmission Software (PACS) Registration (2016 Circular No. 27) [Z] , 2016.3

[ 12 ] Former State Food and Drug Administration . Guidelines for Technical Review of Medical Device Network Security Registration (2017 No. 13 Announcement) [Z] , 2017.1

[ 13 ] Former State Food and Drug Administration . Guidelines for the Division of Medical Device Registration Units (Circular No. 187 in 2017) [Z] , 2017.11

[14] Former State Food and Drug Administration . Guidelines for the Technical Review of Central Monitoring Software Registration (Notice No. 198 in 2017) [Z] , 2017.12

[ 15 ] Former State Food and Drug Administration . Guidelines for Technical Review of Mobile Medical Device Registration (2017 Circular No. 222) [Z] , 2017.12

[ 16 ] State Drug Administration . Guidelines for Technical Review of Active Medical Device Use Period Registration ( Circular No. 23 of 2019 ) [Z] , 2019.5

[ 17 ] State Drug Administration . Independent software of appendix of good manufacturing practice for medical devices (Notice No. 43 of 2019) [Z] , 2019.7

[ 18 ] State Drug Administration . Guidelines for the naming of generic names of medical devices ( Circular No. 99 of 2019 ) [Z] , 2019.12

[ 19 ] State Drug Administration . Basic Principles of Medical Device Safety and Performance (2020 No. 18 Notice ) [Z] , 2020.3

[ 20 ] State Drug Administration . Guidelines for naming common names of medical software ( Circular No. 48 in 2021 ) [Z] , 2021.7

[ 21 ] State Drug Administration. Technical Guidelines for Clinical Evaluation of Medical Devices ( Circular No. 73 of 2021) [Z] , 2021.9

[22] State Drug Administration. Guidelines for Compilation of Technical Requirements for Medical Device Products (Circular No. 8 of 2022) [Z], 2022.2

[ 23 ] National Medical Products Administration . Guidelines for the on-site inspection of independent software of medical device production quality management standards (Medical Administration Comprehensive Equipment Management [2020] No. 57) [Z] , 2020.5

[ 24 ] Center for Technical Review of Medical Devices, State Drug Administration . Guidelines for Technical Review of Human Factors Design of Medical Devices (Draft for Comment) [Z], 2020.5

[ 25 ] Medical Device Technology Review Center of the State Drug Administration . Key points of deep learning-aided decision-making for medical device software review (2019 No. 7 Announcement) [Z], 2019.7

[ 26 ] Center for Medical Device Technology Review of State Drug Administration . Key points of software review for CT image-assisted triage and assessment of pneumonia (trial implementation) ( Circular No. 8 of 2020 ) [Z] , 2020.3

[ 27 ] GB 4793.1-2007 Safety Requirements for Electrical Equipment for Measurement, Control and Laboratory Use Part 1: General Requirements[S]

[ 28 ] GB 4943.1-2011 Information Technology Equipment Safety Part 1: General Requirements [S]

[ 29 ] GB9706.1-20 20 Medical Electrical Equipment Part 1: General Requirements for Basic Safety and Basic Performance [S]

[ 30 ] GB 16174.1-2015 Active Implantable Medical Devices for Surgical Implants Part 1: General Requirements for Safety, Marking and Information Provided by the Manufacturer [S]

[ 31 ] GB / T 8566-2007 Information Technology Software Life Cycle Process [S]

[ 32 ] GB/T 9254-2008 Radio Disturbance Limits and Measurement Methods for Information Technology Equipment [S]

[ 33 ] GB/T 11457-2006 Information Technology Software Engineering Terminology [S]

[ 34 ] GB/T 18268.1-2010 Electromagnetic Compatibility Requirements for Electrical Equipment for Measurement, Control and Laboratory Use Part 1: General Requirements [S]

[35] GB/T 19003-2008 Software Engineering GB/T 19001-2000 Guidelines for the Application of Computer Software [S]

[ 36 ] GB/T 20157-2006 Information Technology Software Maintenance [S]

[ 37 ] GB/T 20158-2006 Information Technology Software Life Cycle Process Configuration Management [S]

[ 38 ] GB/T 20438.1-2017 Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems Part 1: General Requirements[S]

[ 39 ] GB/T 20438.2-2017 Functional Safety of Electrical / Electronic / Programmable Electronic Safety-Related Systems Part 2 : Requirements for Electrical / Electronic / Programmable Electronic Safety-Related Systems [S]

[ 40 ] GB / T20438.3-2017 Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems Part 3: Software Requirements[S]

[ 41 ] GB / T20438.4-2017 Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems Part 4: Definitions and Abbreviations[S]

[ 42 ] GB/T 20918-2007 Information Technology Software Life Cycle Process Risk Management [S]

[ 43 ] GB/T 25000.10-2016 System and Software Engineering System and Software Quality Requirements and Evaluation (SQuaRE) Part 10: System and Software Quality Model[S]

[44] GB/T 25000.12-2017 System and Software Engineering System and Software Quality Requirements and Evaluation (SQuaRE) Part 12: Data Quality Model [S]

[ 45 ] GB/T 25000.51-2016 System and Software Engineering System and Software Quality Requirements and Evaluation (SQuaRE) Part 51: Quality Requirements and Test Specifications for Ready-to-Use Software Products (RUSP) [S]

[ 46 ] GB/T 29832.1-2013 System and Software Reliability Part 1: Index System [S]

[ 47 ] GB/T 29832.2-2013 System and Software Reliability Part 2: Measurement Methods [S]

[4 8 ] GB / T 29832.3-2013 System and Software Reliability Part 3 : Test Methods [S]

[ 49 ] GB / T 31167-2014 Information Security Technology Cloud Computing Service Security Guide [S]

[ 50 ] GB / T 31168-2014 Information Security Technology Cloud Computing Service Security Capability Requirements [S]

[ 51 ] GB/T 32422-2015 Software Engineering Software Exception Classification Guide [S]

[ 52 ] GB/T 35278-2017 Information Security Technology Mobile Terminal Security Protection Technical Requirements [S]

[ 53 ] YY 0505-2012 Medical Electrical Equipment Part 1-2: General Requirements for Safety Collateral Standard: Electromagnetic Compatibility Requirements and Tests [S]

[ 54 ] YY 0637-2013 Safety requirements for radiotherapy planning systems for medical electrical equipment [S]

[ 55 ] YY 0709-2009 Medical Electrical Equipment Part 1-8: General Requirements for Safety Collateral Standard: Tests and Guidelines for Alarm Systems in Medical Electrical Equipment and Medical Electrical Systems [S]

[ 56 ] YY 0721-2009 Safety of Radiotherapy Recording and Verification System for Medical Electrical Equipment [S]

[ 57 ] YY 0775-2010 High-energy X (γ) beam dose calculation accuracy requirements and test methods for teletherapy planning systems [S]

[ 58 ] YY/T 0287-2017 Medical device quality management system for regulatory requirements [S]

[ 59 ] YY/T 0316-2016 Application of Medical Device Risk Management to Medical Devices [S]

[ 60 ] YY/T 0664-20 20 Medical Device Software Software Life Cycle Process [S]

[ 61 ] YY/T 0887-2013 Dose calculation requirements and test methods for radioactive seed implantation treatment planning system [S]

[ 62 ] YY/T 0889-2013 Intensity Modulated Radiation Therapy Planning System Performance and Test Methods [S]

[ 63 ] YY/T 0973-2016 Performance and test methods of radiotherapy planning system for automatic control brachytherapy equipment [S]

[ 64 ] YY/T 1406.1 - 2016 Medical Device Software Part 1: Guidelines for YY/T 0316 Applied to Medical Device Software[S]

[ 65 ] [YY/T 1630-2018 Basic Requirements for Unique Identification of Medical Devices](http://www.baidu.com/link?url=QWdGwquBYdapPo1ilF_j-8Ik2DhmrGT8tMtPeu1NO4D7ZefcxtiZ9oPGpTK6AI-2GlFE0DjvYP-H6JBvb5E7Ka) [S]

[ 66 ] [YY/T 1681-2019 Basic Terms of Medical Device Unique Identification System](http://www.baidu.com/link?url=T0y9X8Xq76lRzdwgK1rgXzxM1CNJ3QhpwDpS5CvbjhpZ21Gc98jwmjnKo4KQAmrkqCWxES0stZn_gDy1dXSlQVeZA0nhAgk5tWvo2UvH6La) [S]

[ 67 ] IMDRF/UDI WG/N7 FINAL: 2013, UDI Guidance: Unique Device Identification of Medical Devices [Z] , 2013.9

[ 68 ] IMDRF/ SaMD WG/N10 FINAL: 2013 , SaMD: Key Definitions [Z] , 2013.12

[ 69 ] IMDRF/ SaMD WG/N12 FINAL: 2014, SaMD: Possible Framework for Risk Categorization and Corresponding Considerations [Z] , 2014.9

[ 70 ] IMDRF/ SaMD WG/N23 FINAL : 2015, SaMD: Application of Quality Management System [Z] , 2015.10

[ 71 ] IMDRF/SaMD WG/N41 FINAL : 2017, SaMD: Clinical Evaluation [Z] , 2017.9

[ 72 ] IMDRF/CYBER WG/N60FINAL : 20 20 , Principles and Practices for Medical Device Cybersecurity [Z ] , 20 20.4

[ 73 ] FDA, General Principles of Software Validation [Z] , 2002.1

[ 74 ] FDA, Cybersecurity for Networked Medical Devices Containing Off-the-Shelf Software [Z] , 2005.1

[ 75 ] FDA, Guidance for the Content of Premarket Submissions for Software Contained in Medical Devices [Z] , 2005.5

[ 76 ] FDA, Modifications to Devices Subject to Premarket Approval(PMA) [Z] , 2008.12

[ 77 ] FDA, Acceptable Media for Electronic Product User Manuals [Z] , 2010.3

[ 78 ] FDA, Computer-Assisted Detection Devices Applied to Radiology Images and Radiology Device Data [Z] , 2012.7

[ 79 ] FDA, Considerations for Computer-Assisted Detection Devices Applied to Radiology Images and Radiology Device Data [Z] , 2012.7

[ 80 ] FDA, Content of Premarket Submissions for Management of Cybersecurity in Medical Devices [Z] , 2014.10

[ 81 ] FDA, Postmarket Management of Cybersecurity in Medical Devices [Z] , 2016.12

[ 82 ] FDA, Applying Human Factors and Usability Engineering to Medical Devices [Z] , 2016.2

[ 83 ] FDA, Deciding When to Submit a 510(k) for a Software Change to an Existing Device [Z] , 2017.10

[ 84 ] FDA, Design Considerations and Pre-market Submission Recommendations for Interoperable Medical Devices [Z] , 2017.9

[ 85 ] FDA, Medical Device Accessories - Describing Accessories and Classification Pathways [Z] , 2017.12

[ 86 ] FDA, [Technical Considerations for Additive Manufactured Medical Devices](http://www.fda.gov/media/97633/download) [Z], 2017.12

[ 87 ] FDA, Content of Premarket Submissions for Management of Cybersecurity in Medical Devices (Draft Guidance)[Z] , 2018.10

[88] FDA, Developing a Software Precertification Program: A Working Model[Z], 2019.1

[89] FDA, Technical Performance Assessment of Quantitative Imaging in Device Premarket Submission(Draft Guidance)[Z], 2019. 4

[ 90 ] FDA, Off-The-Shelf Software Use in Medical Devices [Z] , 2019.9

[ 91 ] FDA, General Wellness: Policy for Low Risk Devices [Z] , 2019.9

[ 92 ] FDA, Medical Device Data Systems, Medical Image Storage Devices, and Medical Image Communications Devices [Z] , 2019.9

[ 93 ] FDA, Policy for Device Software Functions and Mobile Medical Applications [Z] , 2019.9

[ 94 ] FDA, Changes to Existing Medical Software Policies Resulting from Section 3060 of the 21st Century Cures Act [Z] , 2019.9

[ 95 ] FDA, Clinical Decision Support Software (Draft Guidance)[Z] , 2019.9

[ 96 ] FDA, Multiple Function Device Products: Policy and Considerations [Z], 2020.7

[97] FDA , Artificial Intelligence and Machine Learning (AI/ML) Software as a Medical Device (SaMD) Action Plan[Z] , 20 21. 1

[98] FDA , Content of Premarket Submissions for Device Software Functions(Draft Guidance)[Z], 20 21 . 11

[99] FDA. Technical Considerations for Medical Devices with Physiologic Closed-Loop Control Technology (Draft Guidance) [Z], 2021.12

[100] FDA. Assessing the Credibility of Computational Modeling and Simulation in Medical Device Submissions(Draft Guidance)[Z], 2021.12

[ 101 ] FDA, FDASIA Health IT Report: Proposed Strategy and Recommendations for a Risk-Based Framework [ Z], 2014.4

[ 102 ] FDA, Report on Non-Device Software Functions: Impact to Health and Best Practices [Z], 20 20.11

[ 103 ] MEDDEV 2.1/6, Qualification and Classification of Standalone Software [Z], 2016.7

[ 104 ] MDCG 2019-11, Guidance on Qualification and Classification of Software in Regulation (EU) 2017/745 - MDR and Regulation ( EU ) 2017/746 - IVDR [Z] , 2019.10

[ 105 ] [MDCG 2020-1](https://ec.europa.eu/docsroom/documents/40323/attachments/1/translations/en/renditions/native) , Guidance on clinical evaluation(MDR)/Performance evaluation(IVDR) of medical device software [Z] , 20 20. 3

[106] AAMI TIR45:2012/(R)2018, Guidance on the use of AGILE practices in the development of medical device software [S]

[ 107 ] [ISO TR 17791:2013](https://webstore.iec.ch/publication/9274) , Health informatics - Guidance on standards for enabling safety in health software [S]

[ 108 ] IEC 62304 AMD1 : 2015, Medical device software - Software life cycle processes [S]

[ 109 ] IEC 62366-1:2015 Medical devices - Part 1: Application of usability engineering to medical devices [S]

[ 110 ] IEC 80001-1:2010, Application of risk management for IT-networks incorporating medical devices - Part 1: Roles, responsibilities and activities [S]

[ 111 ] ISOTR 80002-2 :2017 , Medical device software - Part 2: Validation of software for regulated processes [S]

[ 112 ] IECTR 80002-3:2014, Medical device software - Part 3: Process reference model of medical device software life cycle processes (IEC 62304) [S]

[ 113 ] IEC 82304-1 :2016 , Health Software - Part 1: General requirements for product safety [S]

[ 114 ] ISO TS 82304-2:2021 , Health software - Part 2: Health and wellness apps - Quality and reliability [S]

Appendix: Standalone Software Product Technical Requirements Template

Technical requirements for medical device products

**Medical device product technical requirement number:**

**product name**

**1. Product model/specification and its division description**

1.1 Software Model Specifications

Specify the model/specification of the software without reflecting the software release version.

1.2 Software Release Version

Specify the software release version. If the software module (including medical middleware) is independently version controlled, the release version should also be provided.

1.3 Software Version Naming Rules

the number of digits, range and meaning of all fields in the complete software version . If the software module (including medical middleware) is independently version controlled, its version naming rules should also be provided, and the relationship with the software version naming rules should be clearly defined . The version naming rules of software and software modules shall be consistent with the quality management system.

**2. Performance indicators**

2.1 General requirements

2.1.1 Function

According to the manual and user interface , specify the outline of all functions (including safety functions) of the software for users to call , and indicate optional and automatic functions. Among them, the objective physical measurement function specifies the measurement accuracy index, and data resources (such as reference databases) specify data types and The sample size of each type of data . If the core functions are the same but the core algorithm types are different, each type of core algorithm needs to be remarked.

2.1.2 Usage Restrictions

User limitations and technical limitations of the software shall be clarified according to the manual.

2.1.3 Input and output

Clarify the input data type of the software (such as medical images, physiological parameters, in vitro diagnostic data, etc.) and output result types (such as processing, measurement, analysis, etc.).

2.1.4 Interface

Specify the application program interface (API), data interface (including transmission protocol, storage format, such as DICOM, HL7, JPG, PNG, proprietary protocol and format), product interface (other medical device independent software that can be used in conjunction with the software) for users to call , medical device hardware products).

2.1.5 Necessary software and hardware

Identify other independent software for medical devices (name, model specification, release version), medical middleware (name, model specification, release version), and medical device hardware products (name, model specification) necessary for the normal operation of the software.

2.1.6 Operating Environment

Clarify the typical operating environment required for the normal operation of the software, including hardware configuration (including processor, memory, peripheral devices), external software environment (list all software names, full versions , patch versions, use "compatible version" instead of "Above Version", "Higher Version"), network conditions ( including network architecture, network type, network bandwidth ) , covering client-side, server-side (if applicable), and cloud (if applicable) requirements. There is no need to repeatedly describe the necessary hardware and software.

2.1.7 Performance Efficiency

time characteristics for the software to complete typical core functions in a typical operating environment (including cloud computing), and clarify resource utilization and capacity if applicable .

2.1.8 Maximum concurrent number

Specify the maximum number of concurrent users and/or patients for typical concurrent operations performed by the software in a typical operating environment (including cloud computing), and indicate the corresponding response time.

2.1.9 User Interface

Identify the type of user interface and user input for the software.

2.1.10 Messages

Identify the type and form of messages that the software provides to the user.

2.1.1 1 User Error Defense

Identify the software 's defenses against user error that can lead to serious consequences .

2.1.1 2 Access Control

Clarify the user identification method, user type and user access authority of the software .

2.1.1 3 Copyright protection

Clarify the copyright protection technology of the software and its impact on the normal use of the software .

2.1.14 Reliability \_

Clarify the ability of the software to save , restore , and continue running data in case of errors .

2.1.1 5 Maintainability

maintenance functions and types of maintenance information that the software provides to users .

2.2 Special requirements ( if applicable )

*Note: Describe one by one according to the applicable terms of the special standard (name, year of publication)*

2.2.1 YY 0775-2010 ( if applicable )

2. 2 .2 YY / T 0887-2013 ( if applicable )

2.2.3 YY /T 0889-2013 ( if applicable )

2.2.4 YY /T 0973-2016 ( if applicable )

...

2.3 Safety requirements ( if applicable )

*Note: Specify the name and release year of the safety standard*

2.3.1 YY 0709-2009 ( if applicable )

2.3.2 YY 0637-2013 ( if applicable )

2.3.3 YY 0721-2009 ( if applicable )

...

**3. Inspection method**

3.0 Describe the software testing environment (equivalent to the typical operating environment) according to the detection unit.

3.1 General requirements compliance inspection

The inspection method of each clause of 2.1 is explained one by one by checking the manual, actual operation, software testing and other methods, and the compliance of each clause of 2.1 is verified.

If the core functions are the same but the core algorithm types are different, the core functions corresponding to each type of core algorithm need to be detected (the detection object is the core function rather than the core algorithm).

3.2 Special requirements inspection method ( if applicable )

3.2.1 Test according to the method of YY 0775-2010 ( if applicable ).

3.2.2 Inspect according to the method of YY /T 0887-2013 ( if applicable ) .

3.2.3 Inspection according to the method of YY/T 0889-2013 ( if applicable ) .

3.2.4 Inspection according to the method of YY/T 0973-2016 ( if applicable ) .

...

3.3 Safety requirements inspection method ( if applicable )

3.3.1 Inspection according to the method of YY 0709-2009 ( if applicable ).

Inspection according to the method of YY 0637-2013 ( if applicable ).

3.3.3 Inspection according to the method of YY 0721-2009 ( if applicable ).

...

**4. Terminology** ( if applicable )

*Note: clarify the meaning of professional terms ( abbreviations ) used in the software*

4.1  …

4.2 ...

4.3 ...

...

(pagination)

**appendix**

*Note: All illustrations need to provide necessary notes*

1. Architecture diagram

2. User interface diagram and main interface diagram

3. Physical topology

1. See IMDRF/SaMD WG/N10 FINAL: 2013 for details . [↑](#footnote-ref-1)
2. For medical computing platforms and software components, please refer to GB 9706.1-2020 for definitions and requirements for medical electrical equipment / systems and programmable medical electrical equipment / systems. [↑](#footnote-ref-2)
3. From the perspective of Unique Device Identification ( UDI ), the full version of the software is part of the Manufacturing Identification ( PI ). [↑](#footnote-ref-3)
4. See IMDRF/UDI WG/N7 FINAL: 2013 for details . [↑](#footnote-ref-4)
5. See IMDRF/SaMD WG/N41 FINAL: 2017 for details. [↑](#footnote-ref-5)
6. The minor, medium and severe levels correspond to the A, B, and C levels defined in YY/T 0664, respectively. [↑](#footnote-ref-6)
7. See IMDRF/SaMD WG/N12 FINAL: 2014 for details. [↑](#footnote-ref-7)
8. CAPA ) should be taken and recorded if the software update does not match the software version naming convention . [↑](#footnote-ref-8)
9. See IMDRF/SaMD WG/N23 FINAL: 2015 for details . [↑](#footnote-ref-9)
10. Platform function software is the necessary software for specific function software. [↑](#footnote-ref-10)
11. See IMDRF/SaMD WG/N41 FINAL: 2017 for details . [↑](#footnote-ref-11)
12. Supersedes the requirements for cloud computing in the Mobile Medical Device Guidelines. [↑](#footnote-ref-12)