S. No.	Product	Process
8.	Product patents are thought to offer a greater level of protection than process patents.	A process patent provides the inventor only limited protection.

External Standards:

The British standard for Quality Management systems was previously called BS 5750.

Remember that these standards were originally designed for all kinds of production - not just software development.

BS EN ISO 9001

At IOE, a decision might have been made to use an outside contractor to produce the maintenance group accounts subsystem rather than develop the software in-house. As a client using the services of an outside contractor, IOE would be concerned that the contractor is following the best quality practices. It is now common to include in contracts terms covering the types of technique that a contractor will use. Various national and international standards bodies, including the British Standards Institution (BSI) in the United Kingdom, have inevitably become involved in the creation of standards for quality management systems. The British standard is now called BS EN ISO 9001:1994, which is identical to the international standard, ISO 9001:1994. Standards such as the ISO 9000 series aim to ensure that a monitoring and control system to check quality is in place. They are concerned with the certification of the development process, not of the end product, as in the case of crash helmets and electrical appliances with their familiar CE labels. The ISO 9000 series govern quality systems in general terms and not just those in the software development environment.

There has been some controversy over the value of these standards. Stephen Halliday, writing in The Observer, had misgivings that these standards are taken by many customers to imply that the final product is of a certified standard although as Halliday says 'It has nothing to do with the quality of the product going out of the gate. You set down your own specifications and just have to maintain them, however low they may be'. It has also been suggested that obtaining certification can be an expensive and time-consuming process that can put smaller, but still well-run, businesses at a disadvantage. Finally, there has been a concern that a preoccupation with certification might distract attention from the real problems of producing quality products.

Putting aside these reservations, let us examine how the standard works. A primary task is to identify those things that are to be the subject of quality requirements. Having defined the requirements, a system must be put in place to check that the requirements are being fulfilled and that corrective action is being taken where necessary.

An overview of BS EN ISO 9001 QMS requirements

In order for a quality management system (QMS) to meet the standard it has to conform to certain requirements which are summarized below.

(a) The management must define and document the policy concerning quality and must ensure that this policy is communicated to all levels of the organization.

(b) All quality control procedures must be documented.

(c) All contracts to supply goods or services must contain mutually agreed requirements that the developer is capable of delivering.

(d) There must be procedures to control and verify the design of the system to be supplied so that it meets the requirements agreed with the customer.

(e) There must be procedures to approve design and other documentation.

(f) Where components of the system to be supplied to the client are obtained from third parties there must be procedures to ensure, check and maintain the quality of these components.

(g) Individual products must be identifiable as should their components.

(h) The process by which the final product is created must be planned and monitored.

(i) Inspection and testing must take place during the development phase, at its completion and before delivery. Tests and inspections must also be carried out on components obtained from third parties.

(j) The equipment used in the production process itself must be properly controlled with respect to quality.

(k) The testing status of all components and systems must be clearly recorded at all times.

(1) Care must be taken to ensure that items that are known to be defective are not inadvertently used.

(m) When a defect is detected, measures must be undertaken to remove the defective part and to ensure that the defect does not occur again.

(n) Satisfactory procedures must be in place to deal with correct handling, storage, packaging and delivery of the product.

(o) Sufficient records must be maintained to demonstrate that the quality system is working satisfactorily.

(p) The software quality management system must be audited on a regular basis.

(q) Servicing and support activities must be subject to the quality management system.

(r) The developer must establish appropriate statistical techniques to verify the acceptability of the final product.

Identify specific instances in a software development environment where the requirements about the control of equipment (j), the recording of the testing status of all components (k), and the correct handling, storage, packaging and delivery of the product (m) would be relevant. What procedures would apply in a software environment in relation to these requirements?

Bearing in mind the criticisms of BS EN ISO 9001 that have been mentioned, what precautionary steps could a project manger take where some work of which the quality is important is to be contracted out?

TickIT

However, some parts do The ISO 9000 standards refer to quality management systems in general but in the refer to software, for United Kingdom, the Department of Trade and Industry (DTI) have formulated example, ISO 9000-3. the TickIT standards which give an interpretation of these standards, which applies specifically to software development. This includes such requirements as:

• a detailed development plan is required before development is embarked upon;

• change control procedures should be used at all stages of development;

• design reviews must take place;

• the suitability of the design methodology must be reviewed;

• progress must be reviewed on a systematic basis;

• it must be possible to trace back the features of software design to specifications and requirements;

• designs must be properly documented;

• suitable test plans, specifications and records must be produced;

• a code of practice must be in place which governs the way the software is developed.

The code of practice must include the requirements that:

• the design must be broken down into levels, each with identifiable inputs and outputs;

• software must be organized into modules;

• a module must normally perform a single function or a set of related functions;

• a plain language description must exist for each module.

Exercise 12.8

Exercise 12.9

A TickIT auditor can certify that a particular organization conforms to these standards. This is called certification. The bodies doing this certification have to be accredited by the National Council for Certification Bodies (NACCB) on behalf of the DTI. The scheme is now administered by DISC, a part of the British Standards Institution.

Capability process models

Rather than just checking that a system is in place to detect faults, a customer might wish to check that a supplier is using software development methods and tools that are likely to produce good quality software. Even the TickIT recommendations can be regarded as fairly minimal. A customer will feel more confident, for instance, if they know that the software supplier is using structured methods. In the United States, an influential capability maturity model (CMM) has been developed at the Software Engineering Institute (SEI), a part of the Carnegie-Mellon University. This attempts to place organizations producing software at one of five levels of process maturity to indicate the sophistication and quality of their software production practices. These levels are defined as follows.

• Level 1: Initial The procedures followed tend to be haphazard. Some projects will be successful, but this tends to be because of the skills of particular individuals including project managers. The is no level 0 and so any organization would be at this level by default.

• Level 2: Repeatable Organizations at this level will have basic project management procedures in place. However, the way an individual task is carried out will depend largely on the person doing it.

• Level 3: Defined The organization has defined the way in which each task in the software development life cycle is to be done.

• Level 4: Managed The products and processes involved in software development are subject to measurement and control.

• Level 5: Optimizing Improvement in procedures are designed and implemented using the data gathered from the measurement process.

For each of the levels, apart from the default level 1, key process areas (KPAs) have been identified as distinguishing the current level from the lower ones. These are listed in the Table 12.2.

The assessment is done by a team of assessors coming into the organization and interviewing key staff about their practices using a standard questionnaire to capture the information. A key objective is not just to assess, but to recommend specific actions to bring the organization up to a higher level.

A criticism has been made of the approach that it is unrealistic to try and assess an organization as a whole - in reality there will be major differences between the way that individual projects are conducted. Bootstrap, which is a European initiative along the same lines as CMM, does allow assessment to be done at a project level.

See Watts Humphrey, Managing the Software Process, Addison-Wesley, New York, 1989.

The SEI originally developed CMM for the US Department of Defense who wanted to be able to assess the capability of contractors from whom they procured software.

Bootstrap also caters for ratings between the major levels, for example, at 2.6 which indicates that the project is better than level 2 but not yet up to a level 3 standard.

Table 12.3 CM M Key process areas

Level

Key process areas

1. Initial

2. Managed

3. Defined

4. Managed

5. Optimizing not applicable configuration management, quality assurance, sub-contract management, project tracking and oversight, project planning peer reviews, inter-group co-ordination, software product engineering, integrated software management, training programme, organization process definition and focus quality management, process measurement and analysis process change management, technology innovation, defect prevention

Assessing software products

The concern in this section has so far been with the assessment of organizations and the processes that they use to produce software, but many purchasers of software, including project managers contemplating the purchase of software tools are more directly worried about the quality of the software product itself. Compilers for some programming languages, for example, are subject to certification. Much progress, however, has still to be made in this area.

Techniques To Help Enhance Software Quality:

Listed below are 15 strategies that will drastically improve software quality.

1. Test Early

Testing is important when learning how to improve software quality and shouldn't be neglected. Testing aims to catch defects early during the design phase so they don't snowball and grow into bigger issues later. Software development teams resort to manual testing for many problems, but companies leverage automated testing strategies for non-UI tasks. Testing early also invariably reduces the money spent on bug fixes. A defect that could cost your company $100 in the requirements phase can cost $10,000 or more in the product implementation stages.

2. Implement Cross Browser Testing

Cross browser testing checks if the software runs seamlessly across different web browsers, screen sizes, and mobile apps. With multiple devices and models coming out in the market, cross browser testing is becoming integral for every developer. There are many upsides associated with cloud-based cross browser testing, and effective testing solutions lead to a flawless user experience.

You can cut costs, optimize the speed and performance of your products, and ensure that testing environments stay scalable and dynamic using various cross-browser testing tools. For best results, combine parallel testing and testing automation tools.

BrowserStack allows you to perform cross browser testing on 3000+ devices seamlessly. You can perform both manual and automation tests on any browser or device.
Logikcull, a legal discovery platform uses BrowserStack and has reduced its test time by 73%. See how they were able to achieve this.

3. Test on Multiple Devices

Multi-device testing will help you make better decisions regarding software development and quality improvement processes. There is an overwhelming number of devices, screen sizes, and OS configurations in the market, so it's important to test as many variations as possible. Windows and Mac are the two most popular operating systems used for testing purposes, and web browsers such as Chrome, Safari, Opera, and Firefox cover most of the users. Using the same configurations as end-users when testing software on these browser and OS environments is vital. Testing tools like BrowserStack gives developers access to real-time environments without having to install additional hardware on machines.

4. Optimize Automation Testing

Increasing adoption of Automated testing and Agile methodologies has led to improvements in software quality. According to a 2020-21 World Quality Report, automation testing tools can save time, enhance coverage, minimize human errors, and improve testing capabilities. Some popular approaches are smoke testing, regression testing, cross-device and cross-browser testing, and load testing.
There are various automated testing tools, and it's critical to strike a balance between manual and automated testing methods. Most automated testing solutions can be integrated with Agile workflows and be a core component of DevOps practices.
Sainsbury's leveraged test automation and doubled its release frequency using BrowserStack.

5. Use Quality Controls from the Beginning

Quality management and control is an ongoing process, so testers must cooperate with developers and work together. A structured approach effectively improves test processes and cuts maintenance costs with native testing tools.

6. Leverage Continuous Delivery (CD) and Continuous Integration (CI)

CI-CD requires engineers to integrate changes and improvements to every step of the software development lifecycle. Continuous Delivery focuses on releasing changes to customers interactively, while continuous integration makes code more dependable by integrating modifications to a product multiple times daily.

7. Have Clear Communication

Clear communication with all team members is integral to the software quality improvement process. Having consistent KPIs (Key Performance Indicators) throughout the project and conveying accurate test reports helps in communicating. Everybody should be aligned when

setting testing requirements and sharing feedback. It's critical to get all stakeholders involved in meetings and ensure team members communicate with vendors and do not work in isolation.

Fluid communications can prevent project risks, ensure that processes run smoothly and that the software quality goals of senior management teams are met. Teams can use popular messaging apps like Slack, Discord, and Telegram for seamless communication.

8. Create a Risk Registry

Risk management is critical to software quality improvement, and project managers know they must monitor risks throughout the development lifecycle. A project risk register is also called a risk log, and it is used to identify, track, and analyze potential risks. Your team members should create a risk registry to record these risks, assess them, and assign appropriate priority levels for effective mitigation.

Examples of risks logged include:

- Legal compliance and regulatory risks
- Data security and breach risks
- Unforeseen events such as natural disasters, physical break-ins, and theft
- Supply chain disruptions

A risk register comprises the following elements:

- Identification number (for risks)
- Brief description and overview of each risk
- Risk categories (both internal and external)
- Probability
- Impact and Rating
- Risk analysis approach and specification
- Action plan
- Names of individuals responsible for overseeing, managing, or mitigating risks

9. Document Your Project Requirements

Good documentation defines the scope of the project, milestones, deliverables, and technical specifications, thus ensuring you meet deadlines and stay on track. The documentation also defines customers' needs and lists functional and non-functional requirements.

More importantly, it contains a list of all important features and processes with step-by-step breakdowns that help keep track of the development process. The first step to creating effective documentation is communicating to clients and collecting information about their expectations. All development processes and plans should follow the documentation based on them.

10. Think Outside the Box

Promoting innovation and thinking outside the box should be a no-brainer. Simply copying your competitor's strategy or growth hacks isn't enough. People crave "different," and you will stand out if you build a unique product that others cannot replicate. To improve software quality, think about what you stand for. Automate monotonous processes to free up time for productive tasks and use quality metrics with your testing structures.

11. Incorporate Employee Training

Your employees can play the role of your end-users. Tools, technologies, and techniques evolve, and it's important to stay on track with the latest trends. Employee training instills an awareness of what to look out for in leading software products. Flaws and vulnerabilities your team normally wouldn't notice during user journeys will crop up. Employees should also work on upgrading their coding skills to contribute to the software development process.

12. Create a Quality Management Plan

A quality management plan (document) outlines software quality expectations, defines roles and responsibilities, supports project managers, and organizes tasks to ensure that software development matches customer requirements and expectations. It has key components such as reporting tools and assurance policies, quality standards, testing strategies, and software quality objectives. Think of it as a roadmap for future improvements with the foundation set in stone.

13. Do Formal Technical Reviews

In a formal technical review, the stakeholders meet and discuss the logical and functional errors of a software program. These review rounds require a team of engineers and entail preparing reports for presentations.

The main objective is to give all reviewers a product walkthrough, examine source code, detect bugs, and make additional inspections. These reviews improve the software quality and help keep developers accountable for the production management process.

14. Try Ad Hoc and Exploratory Testing

Ad hoc and exploratory testing go into the manual side of testing. The main idea is to explore creativity and push the boundaries of testing practices. There are no rules to this, and exploratory testing is conducted on the fly at any moment. This approach benefits developers in testing software usability and user behaviours.

Ad hoc testing uses random data to generate tests and aims to break or disrupt software services. Its objective is to find vulnerabilities and is usually done near the end of the development process.

15. Produce Bug Reports

A good bug report can make software testing and improvement highly effective. It includes all possible scenarios, and use-cases and describes behaviours exhibited while testing new

features. You can add screenshots of failure exceptions in the report, list all possible solutions, and a bug summary.

To develop high-quality software and ensure optimal performance, it's important to have stringent software quality measures in place. Extensive testing across different environments using various methodologies is an effective way to improve software quality.

Making good quality software is not a black and white affair but a constant ongoing process. As a developer, you should be willing to make multiple iterations and additional tweaks based on analysis and feedback gathered in the quality management process.