* Software Maintenance

[Software maintenance](https://en.wikipedia.org/wiki/Software_maintenance) is the process of changing, modifying, and updating software to keep up with customer needs. Software maintenance is done after the product has launched for several reasons including improving the software overall, correcting issues or bugs, to boost performance, and more.

Software maintenance is a natural part of SDLC (software development life cycle). Software developers don’t have the luxury of launching a product and letting it run, they constantly need to be on the lookout to both correct and improve their software to remain competitive and relevant.

Using the right software maintenance techniques and strategies is a critical part of keeping any software running for a long period of time and keeping customers and users happy.

**software maintenance important**

Creating a new piece of software and launching it into the world is an exciting step for any company. A lot goes into creating your software and its launch including the actual building and coding, licensing models, marketing, and more. However, any great piece of software must be able to adapt to the times.

This means monitoring and maintaining properly. As technology is changing at the speed of light, software must keep up with the market changes and demands.

* Basics of Software Maintenance

Software maintenance is the process of changing, modifying, and updating software to keep up with customer needs. Software maintenance is done after the product has launched for several reasons including improving the software overall, correcting issues or bugs, to boost performance, and more.

Software maintenance is a natural part of SDLC (software development life cycle).

**Types of software maintenance**

The four different types of software maintenance are each performed for different reasons and purposes. A given piece of software may have to undergo one, two, or all types of maintenance throughout its lifespan.

The four types are:

1. Corrective Software Maintenance
2. Preventative Software Maintenance
3. Perfective Software Maintenance
4. Adaptive Software Maintenance

Corrective Software Maintenance

Corrective software maintenance is the typical, classic form of maintenance (for software and anything else for that matter). Corrective software maintenance is necessary when something goes wrong in a piece of software including faults and errors. These can have a widespread impact on the functionality of the software in general and therefore must be addressed as quickly as possible.

Many times, software vendors can address issues that require corrective maintenance due to bug reports that users send in. If a company can recognize and take care of faults before users discover them, this is an added advantage that will make your company seem more reputable and reliable (no one likes an error message after all).

Preventative Software Maintenance

Preventative software maintenance is looking into the future so that your software can keep working as desired for as long as possible.

This includes making necessary changes, upgrades, adaptations and more. Preventative software maintenance may address small issues which at the given time may lack significance but may turn into larger problems in the future. These are called latent faults which need to be detected and corrected to make sure that they won’t turn into effective faults.

Perfective Software Maintenance

As with any product on the market, once the software is released to the public, new issues and ideas come to the surface. Users may see the need for new features or requirements that they would like to see in the software to make it the best tool available for their needs. This is when perfective software maintenance comes into play.

Perfective software maintenance aims to adjust software by adding new features as necessary and removing features that are irrelevant or not effective in the given software. This process keeps software relevant as the market, and user needs, change.

Adaptive Software Maintenance

Adaptive software maintenance has to do with the changing technologies as well as policies and rules regarding your software. These include operating system changes, cloud storage, hardware, etc. When these changes are performed, your software must adapt in order to properly meet new requirements and continue to run well.

The Software Maintenance Process

The software maintenance process typically involves the following steps:

1. **Problem identification:** The first step is to identify the problem that needs to be addressed. This may be a bug, a performance issue, or a change in the environment.
2. **Analysis:** Once the problem has been identified, it is necessary to analyze the problem and determine the root cause. This may involve reviewing the code, testing the software, or talking to the users of the software.
3. **Design:** Once the root cause of the problem has been determined, it is necessary to design a solution. This may involve modifying the code, adding new features, or improving the user interface.
4. **Implementation:** The next step is to implement the solution. This may involve writing new code, testing the changes, and deploying the changes to the production environment.
5. **Testing:** Once the changes have been implemented, it is necessary to test the software to ensure that the problem has been fixed and that there are no new problems.
6. **Documentation:** The final step is to document the changes that have been made. This is important for future reference and for ensuring that the software is maintained properly.

**Software Maintenance Cost**

The cost of software maintenance can be high. However, this doesn’t negate the importance of software maintenance. In certain cases, software maintenance can cost up to two-thirds of the entire software process cycle or more than 50% of the SDLC processes.

The costs involved in software maintenance are due to multiple factors and vary depending on the specific situation. The older the software, the more maintenance will cost, as technologies (and coding languages) change over time. Revamping an old piece of software to meet today’s technology can be an exceptionally expensive process in certain situations.

In addition, engineers may not always be able to target the exact issues when looking to upgrade or maintain a specific piece of software. This causes them to use a trial and error method, which can result in many hours of work.

There are certain ways to try and bring down [software maintenance costs](https://www.forbes.com/sites/forbestechcouncil/2020/04/10/reducing-maintenance-costs-with-functional-programming/?sh=27b37957f643). These include optimizing the top of programming used in the software, strong typing, and functional programming.

When creating new software as well as taking on maintenance projects for older models, software companies must take software maintenance costs into consideration. Without maintenance, any software will be obsolete and essentially useless over time.

**Software maintenance strategies**

All software companies should have a specific strategy in place to tackle software maintenance in an effective and complete manner.

Documentation is one important strategy in software development. If software documentation isn’t up to date, upgrading can be seemingly impossible. The documentation should include info about how the code works, solutions to potential problems, etc.

QA is also an important part of a software maintenance plan. While QA is important before an initial software launch, it can also be integrated much earlier in the process (as early as the planning stage) to make sure that the software is developed correctly and to give insight into making changes when necessary.