ASSIGNMENT 3 (MAINTENANCE)

Name: Omkar shrikant gawade
ID:1001967237

Challenges in software maintenance

INTRODUCTION

Software development has different phases including requirement phase, design, implementation, testing and maintainance.Software systems are always getting updated for new changes and enhancements.Maintenance is the last stage of software development life cycle.Complex software requires more maintenance for smooth and continued functioning. The definition of software maintenance is “Modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a modified environment.

Let's take an example of a car for better understanding of maintenance terms, as a car requires oil change, tire alignment, engine tuning etc.Similarly software needs some fixes performed to ensure optimal performance. But what if every single component of your car is built by a different manufacturer? In that case your brakes could wear because your gas cap was upgraded to a new version that reminds you to tighten it.

The concept of car maintenance makes sense as a car moving day to day, which needs its physical parts to be replaced but why do we need maintenance in software as there are no moving parts? Well, not physically but things do happen that require certain actions be taken that are intended use of the software.

When a company releases any project to its client within a fixed timeline, then the actual maintenance phase begins. Basically the software maintenance phase keeps the software up to date with new upgrades, updates , environmental changes and improved performance after delivery of the software product.

The main aim of this report is to highlight the major challenges of the maintenance phase. In the first article we are going to explain and discuss the major problems and in the second article we are going to talk about the solution which can be offered for those problems.

FIRST ARTICLE

Legacy Code:

Software maintenance is critical since it improves the product's efficiency and reduces the margin for error. An application can become inefficient and difficult to use if it is not properly maintained. So, to begin with, maintaining a legacy system might be tough because the code utilized is outdated in comparison to new software. The majority of old code is bulky, long, and incompatible with most newer systems.

High maintenance costs:

Another issue with most legacy systems is their expensive maintenance costs, which are often out of reach for most businesses.The entire process of software maintenance entails numerous moving pieces of varied degrees of complexity and resource demands, as well as varying costs. A regular maintenance package might comprise a variety of services, depending on the company's software needs and current market conditions. These services can be divided into three categories.

Lack of sufficient skill sets:

To maintain old software, you'll need a developer who understands how it works. Most developers, on the other hand, are using new technologies to future-proof their programs. As a result, finding someone who can work with an old system can be difficult.Furthermore, maintaining and regulating changes in software can be challenging. Maintaining the systems requires a significant amount of time and effort, which is both costly and time-consuming.

Incompatibility with other IT solutions:

There are now sophisticated tools that can be utilized to make software maintenance quick and painless. However, the majority of legacy IT infrastructure is incompatible with such solutions, making their upkeep more difficult.Developers may find it challenging to incorporate legacy systems into their environments if their features are incompatible with those of contemporary IT solutions.The difficulty of adding new features to legacy systems adds to the difficulties of maintaining them. Trying to restructure and make old systems more manageable may not function as planned because most legacy systems break readily.

SECOND ARTICLE

Legacy technology must be modernized in order to compete in today's competitive IT landscape. Legacy apps that have been updated result in increased user productivity, lower maintenance costs, and more helpful experiences.According to Avanade's recent research, updating IT systems can result in a 14 percent increase in revenue. As a result, selecting various software modernisation choices could provide considerable benefits to your company.

It's crucial to remember that just because software is old doesn't mean it's obsolete. Some 'ancient' software may still have advanced capabilities that are beneficial to an application's maximum performance.As a result, developers can rework the system's source code to tackle the issues of maintaining older software. They can utilize clean, modern code that is reusable and easy to debug in this way.Refactoring is the process of improving the internal structure of a software system. You do not, however, meddle with the code's exterior behavior. As a result of the code's internal enhancements, the software's functionalities are optimized.

Updates and alterations should be thoroughly tested while restructuring legacy code to avoid application breakages and poor performance. Regression tests, for example, can be used to confirm that everything is working properly.

Maintenance cost can be reduced by hiring skilled employees who can identify as well as manage code easily and more efficiently. Also if code is commented and explained very well it will be easy to understand that code and debug or enhance.Using cloud platforms is the best way to handle incompatibility within software systems.

CONCLUSION

Although updating old software is difficult, time-consuming, and costly, the rewards are usually well worth the effort. Continuing to rely on legacy IT systems is similar to sending an urgent communication via the post office when an email could be sent with the press of a button.So the point of this report is that the software maintenance industry required some enhancements and upgrades which can definitely benefit the industry in the upcoming future.

REFERENCE

1.Sharon, D. (1996). Meeting the challenge of software maintenance. , 13(1), 122–125. doi:10.1109/52.476304

2.Chapin, N. (1993). [IEEE Comput. Soc. Press 1993 Conference on Software Maintenance - Montreal, Que., Canada (27-30 Sept. 1993)] 1993 Conference on Software Maintenance - Management problems seen in software maintenance: An empowerment study. , (), 329–336. doi:10.1109/icsm.1993.366929

3.alfrick opidi = https://www.freecodecamp.org/news/legacy-software-maintenance-challenges/