Book Questions
Chapters 15
Devayan Mandal

15.2 Suggest why the cost savings from reusing existing software are not simply proportional to the size of the components that are reused.

Ans: Savings in cost from reusing existing software is not necessarily proportional to the size of the reused software. It is not feasible to assess the size of the software or code in a quantitative manner.

For example, a software development team have written many thousands of lines of code but cannot seem to correct a certain bug. The total money spent by the software company on its software engineers to write the many thousands of lines of code was approximately $15x. A software engineer, specialized in a certain mode of software testing was recruited. He corrected the bug within a few hundred lines of code for $2x. The existing software engineers were using existing code and it didn't solve the problem. The new software engineer used an approach to coding which was novel to the existing software engineers. Hence, there were no savings and in fact a temporary loss for the company using existing software.

Another example would be, a software development team only has to update the user interface for the existing software and add in a few new features to the software. The original code is left predominantly untouched and the user interface subgroup revamps the software's look and feel. In this case, there was a lot of savings in reusing existing code.

15.3 Give four circumstances in which you might recommend against software reuse.

Ans: Four circumstances in which the reuse of software is not recommended are:

1. If the reuse of a certain component of a software or the software itself is receiving several complaints from the end users as well as the software developers, it would be best if the existing code for the component or software be replaced.

2. If the provider of the software does not have a good reputation it might not be in the best interest of the software development team to reuse the provided software. Reasons behind this could be because the software providers may not be able to provide reliable technical support or because the providers may go out of business.

3. The reuse of software is predominantly done when savings are given priority by the software development team. In the case of software, which has to give higher priority to performance or for providing individualized solutions specific to the client's

requirements, the reuse of software may not necessarily meet the end goal of the software development project.

4. Reusing software without obtaining permission from the software code providers is usually not recommended. In an academic or professional setting, unauthorized reuse of software can be challenged on ethical grounds.

15.8 Identify six possible risks that can arise when systems are constructed using existing application systems. What steps can a company take to reduce these risks?

Ans: Six possible risks that can arise when systems are constructed using existing application systems include:

1. The risk of the newly constructed system not specifically meeting the system requirements may be present because the nature of requirements addressed by the existing application systems, which were used did not match with the new system's requirements.

2. The risk of constructing a new system which cannot guarantee a long shelf life because of the unstable shelf life of the existing applications systems used. If technical support and continuous upgrades are not available for the existing applications systems, the length of the new system cannot be guaranteed.

3. The risk of spending excessive time by the software development team to understand, learn and incorporate the existing application systems into the new system. Also, if excessive man hours spent trying to understand the existing code may contribute to excessive money spent to train the development team too.

4. The risk of the existing application system provider discontinuing business with the new system providers. This would directly impact the outcome and system run time of the new system.

5. The risk of the existing application system providing software support which sourced from other vendors illegally. It is very important to determine this as the new system would be directly affected by such services.

6. The risk of increased expenditure to support the newly constructed system. Such a case may arise if existing application systems were chosen in such a way that they proved to be the least expensive 'out-of-pocket' options but later down the system life cycle were the reason for continuous software system breakdown.

It is important to conduct a thorough market research and ascertain whether the existing applications system providers hold a good reputation. This will prevent abrupt discontinuation of technical support and services in the future.

15.9 Explain why adaptors are usually needed when systems are constructed by integrating application systems. Suggest three practical problems that might arise in writing adaptor software to link two application systems.

Ans: Adaptors are usually needed when systems are constructed by integrating application systems because the adaptors act like connections or bridges between the application systems. Suppose an application system is created for the sole purpose of acting as a database to store many thousands of data entries. Another application system is created for the sole purpose of providing an interface for a website associated to the medical setting. Adaptors can be used to integrate the user experience functionality provided by the second application to the data storage provided by the first application and this would allow the end users in the medical setting to better focus on their medical responsibilities as opposed to talking to tech support.

Three practical problems that might arise in writing adaptor software to link two application systems are:

1. The adaptor software was able to link the two application systems appropriately in a controlled, test setting but failed in the real life setting. Reasons for this may include inadequate testing during the development and quality assurance phases and because all client requirements were not met.

2. The adaptor software does not integrate the technical (high level) and non-technical languages used by the individual application systems. This may lead to breakdown of the newly integrated system and would require re-assembly of the system shortly.

3. The adaptor software does not fully address the scope of changes and connections which need to be made. For example, one of the applications may require a lot of code which needs to be written in order to connect and support the other application system. Hence, the adaptor software development team will need to ensure that any missing connections are appropriately covered before delivery.