

Legacy Systems and their evolution through Cloud Services

Manoj Kumar¹, Dr. Aman Jain²

¹Research Scholar (Singhania University),
Pacheri Bari, Jhunjhunu, Rajasthan, India
E-Mail : mkdhull89@gmail.com

²Professor at Maharishi Arvind Institute of Science and Management,
Jaipur, Rajasthan, India
E- mail:- amanjain.jpr@gmail.com

Abstract - Computer technology has changed rapidly in the last few years. The emphasis shifted toward more distributed computing power residing on an end user's desk or on a local network as opposed to the more traditional centralized mainframe-oriented culture of the 1960s and 1970s.

Most mission-critical data has been traditionally stored on mainframe applications set up years ago. Over time these applications have become known as legacy systems. Most of the applications are old and have been updated in a patchwork fashion. They also tend to be very proprietary and stovepipe.

Many organizations today are struggling with how to make best use of these systems. On one hand, there is the realization that most of the legacy applications are fairly inexpensive and easy to maintain and in many cases serve their users well. On the other hand, there is the user demand for easier ways to access and use legacy data, and a greater need to share this information both within the organization and, increasingly, outside the organization.[1]

This paper suggests how several organizations and firms are dealing with the need to get better access to data stored on legacy systems. There are a variety of ways to deal with the issue and our research found that it depends on a number of key variables which vary from organization to organization. Some of these variables include: budget, centralized or decentralized information resources management organization, end user requirements, management support, and the legacy application itself.

While there is no one right way to deal with legacy systems, we are trying to find out a solution for Legacy Software evolution through Cloud Services.

Key Words : Legacy Software, Software Evolution, Cloud Computing

1. WHAT IS LEGACY SYSTEM?

Organizations, particularly those with programs of longstanding, are often using application programs that have been in existence in many years. These older application programs have become known generically as "legacy systems." Although there is not a precise, agreed upon definition of exactly what a legacy system is, the term "Legacy System" [2] typically applies to software applications having the following characteristics:

- More than 10 years old
- Changed over a period of years
- Mainframe based
- Proprietary language
- Flat file or hierarchical database
- Difficult to port
- Poor documentation
- Limited flexibility
- Non-optimal run times
- Limited access to data

1.1 Characteristics of Legacy Applications

In addition, many of these applications were developed in

response to a specific need or piece of legislation. The applications often do not communicate or share data with other applications in use within the organization. This makes it difficult for the systems to keep pace with business changes and changing information needs within the organization.

2. ASSESSING LEGACY SYSTEMS

It cannot be assumed that just because an application is "old" or a "legacy system," that it is no longer good or should be discarded. Each application must be examined separately. The organization may then determine what changes, if any, need to be made for the systems to better support the current needs of the organization.

2.1 Reasons to keep a Legacy System

- The system meets the needs of the organization.
- New investments cannot be justified.
- Changing the platform does not always improve the system.
- Disagreement within the organization over whether a system should be changed.
- Low maintenance of many legacy systems.

2.2 Reasons to change or get rid of a Legacy System

For the legacy applications that are not fully meeting the current needs of the organization, most have one or more of the following limitations:

2.2.1 Modifications are difficult

Many of the older legacy systems were poorly documented when the application was written. Over the years changes and patches have been made to the system. These changes were often poorly documented as well. In some organizations, the IRM personnel are extremely hesitant to make any additional changes, even small ones, to these systems. There is the risk that the changes will either make the system stop working altogether or that the changes made will have some other, unknown effects on another part of the system that cannot be accurately anticipated.

When faced with this situation, organizations must look at what the system does and the alternative viable solutions. This includes examining not only the individual application and the functions it performs, but addressing the issue of the hardware platform on which the application should run.

Issues regarding how the functions of the application should be changed and how much, if any, of the original coding should be salvaged should also be considered.[3]

2.2.2 More flexibility is needed

One of the key issues in deciding whether to move away from a legacy system is the need for a more flexible platform to use or manipulate data. The diversity of organization missions has fostered the growth of proprietary "stovepipe" systems that cannot be easily changed or share information. Many organizations are focusing on the need to share data across applications to eliminate redundant data entry, provide better customer service, and to share data across organizations. The flexibility and structure needed to do these things are often lacking in organizations legacy applications.

Until recently, one of the key advantages of mainframe based legacy systems was that they could perform multitasking and do more than one operation at a time. Now that desktop systems are capable of multi-tasking, new opportunities to shift legacy systems to alternative hardware platforms are open to organizations.

2.2.3 More timely information is required

With the increased emphasis on customer service and the higher level of expectations from customers, many organizations are no longer satisfied with the level of support possible using their legacy applications. Customers are no longer content to wait long periods of time for changes to be executed or to receive information they have requested. Organizations that are relying on systems using batch processing for their transactions are often frustrated in their efforts to provide an improved level of customer service. In addition, it is less efficient and more costly for the organization to deal with the customer, or data regarding the customer, multiple times rather than once. As a result, many organizations are migrating their applications to an environment that will accommodate on-line transaction processing and also provide good quality software. [4]

2.2.4 The operating system software, or the hardware platform it runs on, are no longer supported

Some legacy applications are residing on hardware that is obsolete, unreliable, no longer supported by the manufacturer, and difficult to repair. In addition, in some cases, the operating system software is no longer being supported.

3. ALTERNATIVE APPROACHES

In response to issues such as those discussed in the previous section, many organizations have taken steps to provide more user-friendly, and more efficient access to the data contained in legacy applications. Not every organization is using the same approach to enhance their access to legacy application information. Some of the different approaches are:

3.1 Keep the existing legacy application without changes.

Many agencies using legacy applications are taking no actions to improve access to the information in the legacy applications. Even when it is clearly recognized the systems are not meeting all of the agency's needs, significant changes are not being made at this time. Budget and staffing constraints are most often cited as reasons for continuing with the status quo.

3.2 Significantly change or re-write the legacy application.

Several organizations are taking this approach. After examining the problems being encountered with the use of their legacy systems, and the increasing need to share data across programs, the decision can be taken that a new application system could better meet the needs of the organization. Sufficient support and funding for such a large scale program needs security and the writing of new application systems can be started.

4. SOFTWARE EVOLUTION AND CLOUD COMPUTING

Now a days, software is produced at an ever-increasing rate and the demand from customers is coming in at an even faster rate, which means that almost all newly developed software can become legacy soon after they are released. Few trends in software development are that existing software is looked at before a new system is designed and developed. In a sense, this is to evolve a piece of software on a gradual basis. This can also be very important for popular paradigms as the essence of these paradigms is to achieve software under the special paradigm that should have all desired features. On one hand, developing new software and evolving software are needed, and on the other hand evolving existing software into existing paradigms can be equally important. Therefore, it is agreed that software always needs evolving to remain useful and which can cope with users' frequent changing requirements. The complexity and dynamic nature of changing requirements [5][6] will make software evolution more difficult, as failing to evolve

correctly may result in catastrophic consequences, e.g. loss of money, time, even including human life. In addition, due to the sheer size of software systems, it is often impossible for a single individual to conduct evolution and therefore the process should be a systematic and group-system approach.

Software evolution [7] is a set of activities, both technical and managerial, that ensures that software continues to meet organizational and business objectives in a cost-effective way. As a combination of reverse engineering and forward engineering, software evolution is continuous software reengineering, i.e., operating->reverse engineering->re-specifying->forward engineering... An evolution process should be able to realize better software systems that can not only reuse the valuable components of legacy systems but also provide large profits to enterprise and software maintenance.

Cloud computing [8] has emerged as an important trend in information technology in recent years. In this study, the paper quotes a less technical and more customer appealing definition from Marks and Lozano [9], —Cloud computing is a style of computing where computing resources are easy to obtain an access, simple to use, cheap, and just work. Meanwhile, an academic definition is also introduced by NIST [10], —Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Cloud computing is becoming the preferred paradigm for those applications with large scalability, dynamic collaboration and flexible resource requirements. The characters of resource reuse and time cost saving have attracted enterprises to consider migrating their systems into the popular paradigm, and the academic discussion on the paradigm. Hence, the research on Cloud-oriented evolution (COE) has been triggered.[12]

5. CONCLUSION

This paper presented the concept of legacy software and their need in the organization. But due to new technology and the need of data sharing with newly developed software, legacy software/systems must be integrated with newly developed systems. A new and cost effective approach to meet these requirements is Cloud Computing,. But evolution of legacy systems through cloud computing is a challenging task though may require new algorithms [11] and architecture. That's why we have adopted this topic for our research work.

References

- [1] Cannon, R. 2001. Theory building in applied areas. Handbook of Industrial and Organisational Psychology. New York: John Wiley. Chap. Chapter 1, pages 17-39.
- [2] J. Lavery, B. Boldyreff, B. Ling and C. Allison, "Modeling the Evolution of Legacy Systems to Web-based System", Journal of Software Maintenance and Evolution : Research and Practice-Special issue : Web site evolution, John Wiley & Sons, Inc., Vol. 16, No. 2, pp. 5-30, 2004.
- [3] J. Kim, S. Park and V. Sugumaran, "A Framework of Domain Requirements Analysis and Modeling Architecture in Software Product Lines", Journal of Systems and Software, Elsevier Science Inc., Vol. 81, No. 1, pp. 37-55, 2008.
- [4] ISO, "Systems and software engineering--Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models", TechnicalReport : International Organisation of Standardisation, 2011.
- [5] IBM Software Group, "Managing Legacy Integration with IBM Rational Software", TechnicalReport : TP902, IBM, USA, 2003.
- [6] J. S. Hennumann, "Requirements Discovery for Legacy Systems", TechnicalReport : IBM Software Group, 2003.
- [7] K. H. Bennett and V. Rajlich, "Software Maintenance and Evolution: A Roadmap", Proceedings of the Conference on The Future of Software Engineering, New York, USA, pp. 73-87, 2000.
- [8] R. Buyya, C. Yeo, S. Venugopal, J. Broberg and L. Brandic, "Cloud Computing and Emerging IT Platforms: Visson, Hype and Reality for Delivering Computing as the 5th Utility", Future Generation Computer Systems, Elsevier Science Publishers, Vol. 25, No. pp. 599-616, 2009.
- [9] E. Marks and B. Lozano, Executive's Guide to Cloud Computing, John Wiley & Sons, USA, 2010.
- [10] B. Navneet and Rai, Strategic Decision Making: Applying the Analytic Hierarchy Process, Springer-Verlag, London, 2004.
- [11] Y. Guo, M. Ghanem and R. Han, "Does the Cloud need new algorithms? An introduction to elastic algorithms", IEEE 4th International Conference of Cloud Computing Technology and Science, Taipei, pp. 66-73, 2012.
- [12] L. Q. Chen, "Integrating Cloud Computing Services Using Enterprise Service Bus (ESB)", Business and Management Research, Sciedu Press, Vol. 1, No. 1, 2012.