A Framework to Assess Legacy Software Systems

Basem Y. Alkazemi
Department of Computer Science, Umm Al-Qura University, Makkah, Saudi Arabia
Email: bykazemi@uqu.edu.sa

***Abstract*—Enterprise organizations require flexible software systems that can handle emerging market needs from time to time, chiefly to ensure the organization has a competitive edge in the market. On the other hand, several organizations still possess legacy software systems, which have definite functionalities but may limit their development progress to comply with emerging e-business needs. Off-the-shelf systems might be one possible option if a decision for replacement is made. Equally, improving the architecture of a legacy system can be a valid option to consider as well. The decision on which approach to follow when dealing with a legacy software system must be made based on a thorough investigation of the nature of the system. This paper develops a conceptual framework to analyse legacy software systems in order to support CEOs in making informed decisions about the appropriate approaches to follow in order to keep their organizations functional and competitive.**

***Index Terms*—legacy system, software architecture, decision making**

I. Introduction

Legacy software systems still represent valuable assets to many organizations, even with the pioneering of new technologies and wisdom. One of the main aspects that forces an organization to maintain its legacy system in business is the highly customized functionality provided by the system, which cannot be easily implemented in many brand-new enterprise solutions. However, such reluctance to put new systems in place might negatively affect the evolution of the organization in terms of its compliance with the latest business needs (e.g. e-commerce models). Therefore, CEOs must be able to make informed decisions on how their organizations can proceed towards satisfying new business requirements. This paper presents our analysis of the Student Information System (SIS) at Umm Al-Qura University (UQU) [1] in order to identify its appropriateness to stay in business and to subsequently help the CEOs at UQU to make a decision about the most suitable strategy to follow in order to keep the organization functional in a business context.

The remainder of this paper is organized as follows: section 2 describes the key background work in the field of assessing legacy systems. Section 3 presents the proposed framework for assessing legacy systems. Section 4 describes the case study with which we conducted the experimentations. Section 5 presents the application of our framework to assess the legacy system we selected as the case study. A discussion about the results obtained from analysing the legacy system and the implications of these results are given in section 6. Finally, section 7 presents the conclusions of the paper and suggests avenues for future work.

II. Background Work

The term "legacy system" in this paper refers to a running system that still meets a considerable percentage of an organization's business needs in terms of functional aspects but does not comply with emerging architectural standards. Although this view might somewhat differ from common definitions in the literature, such as that of McGee [2], our definition considers more specifically the architectural aspects of the legacy system, in addition to other factors such as support, functionality, and technology. Based on the architectural style adopted by an organization, we can classify the system in terms of whether it is a legacy system or not regardless of the functional aspects, as we assume that all systems available in an organization are being utilized; otherwise, they would be shut down. For example, desktop applications that are based on a client–server architectural pattern can be considered legacy systems if they are still in use among large enterprise organizations such as universities. The reason for this is that client–server solutions have many limitations with respect to extensibility and the adaptation of new emerging e-business requirements. Thus, a system can be considered a legacy system in one context but not in another smaller one (e.g. for student projects).

Summerville [4] identifies two main dimensions for assessing legacy systems, namely environment and application. Environment assessment is concerned with certain vendor- and technology-related aspects. Application assessment, however, examines the internal non-functional attributes of the system itself. A more refined version of this assessment framework is proposed by Ransom et al. [5], who consider the technical, business, and organizational infrastructure aspects of the system. Grace et al. [6] established the SMART methodology for analysing legacy software systems in order to migrate to an SOA-compatible [6] environment. Their analysis is mainly concerned with identifying potential business logics from legacy systems in order to refactor and expose these as service components. Seacord et al. [7] identified three main dimensions for assessing legacy systems, namely technical, programmatical, and organizational considerations, which must be thoroughly

doi:10.4304/jsw.9.1.111-115

investigated before modernizing any legacy system. Bennett et al. [8] established a two-phased decision model for assisting organizations in making informed decisions about their legacy systems. Their model is concerned with the business strategy and the technical aspects of the legacy system.

We observe that the above-described work focuses mainly on the business aspects of legacy systems in order to support CEOs in making informed decisions on whether to replace the entire system with a brand new one or to keep the legacy system functional and maintain it over time. However, none of the extant research describes an assessment for the purpose of re-architecting an organization's legacy system, as described in our prior work [9], apart from the work by Grace et al. [6], who identified the requirements to migrate into an SOA-compatible environment. Thus, the aim of this work is to assess legacy systems from the perspective of modifying their architectural styles to comply with new ones that can easily accommodate newly emerging e-business needs. Our assessment is built on combining the previous work described above and considering the key factors of it as a baseline with some modifications to fit the context of our study.

III. Assessment Framework

Generally, there are four conceptual strategies that might be followed to identify the most appropriate solution for systems evolutions, including:

- Replacing the legacy system with a new enterprise solution;
- Maintaining the legacy system in order to proceed with the organization's business with limited evolution;
- Re-architecting the legacy system to comply with modern architectural styles, while keeping the functionality untouched; and
- Extending the legacy system by wrapping it as a black box and exposing its standard interfaces to interact with new systems.

Each one of these strategies incurs some limitations and possibly a number of benefits, if implemented carefully. However, the decision to proceed with one strategy or another is relatively challenging and requires thorough investigation of the business value and the quality of the legacy system at hand. We identify a number of dimensions for this assessment, as follows:

- **Support:** This dimension is concerned with the hardware and software support provided to the legacy system. In addition, this dimension examines the availability of source code and the team that supports it.
- **Business:** The business requirement is the key aspect by which an organization can decide whether a system should be kept functional, replaced with a new system that satisfies their needs, or simply shut down completely. Therefore, this dimension covers the requirements of an organization in addition to the modelling technique used and the documentation of the business in the legacy system.
- **Architecture:** The building blocks of the system are defined in this dimension through the style used in the legacy system and the integration pattern between the different entities of that system and with external systems. It also relates to the way in which the functionality can be consumed by clients.
- **Technology:** This dimension discusses the type of technology adopted by the legacy system and whether the technology is still appropriate or not compared to emerging business needs.

This framework is abstract in nature and therefore is not exhaustive. However, it can be utilized as the basis for analysing the underlying environment of an organization, as will be seen in section 5.

IV. Case Study

UQU is a typical Saudi organization in that it runs a legacy system and thus is in need of an urgent update. UQU was therefore selected as the case study for the proposed model with the purpose of creating a fully integrated environment that supports e-government business. The institution needs a fundamental solution to cope with the changing environment without interrupting the routine working activities. Funding is also a major consideration that influences any decision regarding major development plans.

From a historical perspective, UQU launched its information systems in early 2001 to serve around 3,600 employees and just over 40,000 students. The system runs an outdated code procedure based on Oracle 6i for forms and reports, which are built entirely on a client–server pattern [3]. The major in-house subsidiaries include Enterprise Resource Planning (ERP), Student Information System (SIS), Library Information System (LIS), and Healthcare Information System (HIS). Twelve years later, the system has started to struggle with the new environment because of an increase in the number of users and the pressure to support e-government models. Today, the system serves around 75,000 students and more than 7,000 employees: an almost two-fold increase compared to 2001, although there have been only minor changes to the original core functionality. Moreover, outmoded software systems lack many capabilities that are, these days, considered core requirements, including compatibility with different environments (e.g. mobile devices) and other services provided to students and faculty members in the university. Additionally, due to the emerging e-government movements in Saudi Arabia, organizations need to apply major changes to their core systems in order to accommodate new requirements. One such requirement is process automation, which solely requires the splitting of the functional aspects of an application from the business aspects.

V. Assessment of Legacy System in UQU

We utilized the framework described earlier to assess SIS at UQU in order to build a conceptual understanding of the current state of that system. The different dimensions, together with their corresponding factors and the status of SIS at UQU, are described in Table I.

It is apparent from table I that UQU does not have many problems with the current support provided to their SIS, as its servers are up to date and the source code is completely owned and managed by the UQU team in the IT deanship. With respect to business, it seems that SIS is currently used by the university and hence constitutes part of their active systems. This is why the system is highly necessary for the university, even though it may lack some functionality that seems to be available in a number of competitive solutions in the market. However, the availability of a dedicated IT team does help in terms of adding extra functionality as per the emerging functional requirements of the business owner, and this seems to be a great advantage to the university in a business sense. We can identify a slight concern with respect to the modelling and documentation of the SIS business logic, which seem to be lacking at the moment. When we investigated this further, we observed that the system is dependent mainly on personnel who are working in the environment, including the business owner and senior developers. This might represent a real threat to the university business, as it could suspend the operational continuity of the system in the case of unforeseen circumstances.

Regarding architecture, the system complies with a client–server architectural pattern that we believe is somewhat outdated. Thus, extending the university business to satisfy some of the emerging e-business requirements might be very minimal. Our deep analysis of the current environment has identified that SIS is split into a front end, which is available on the web, and a back end. The front end is programmed purely in Java Server Faces (JSF), which can be considered a fairly modern technology. However, the current building blocks of JSF applications built for UQU are based mainly on object, rather than web-service, interaction. This might limit any potential utilization by different types of environments (e.g. mobile phone applications), as all objects need to be exposed as services beforehand. The back-end system, on the other hand, seems to be very old as it is programmed using Oracle 6i for forms and reports. One common problem in the current back-end system that may hinder its potential extendibility is that business logics are embedded inside Oracle forms. Hence, no application-to-application interaction is possible at the moment without embedding glue code inside the corresponding forms. With respect to data integration, it seems that UQU owns a unified database for most of its applications. However, we observed minor cases where the same data were presented differently on different screens, which can indicate the unnecessary availability of redundant data sources.

Moreover, report generation is a very time-consuming task as it requires manual intervention for creating views and queries. We observed that a considerable amount of IT time is consumed in responding to daily requests for reports due to the lack of a sophisticated tool that end users can utilize for report generation.

TABLE I.
Assessment dimensions of SIS legacy system

Dimension	Factors	Status at UQU
Support	Hardware	Virtual servers supported by CISCO
	Application server	Web-logic currently supported by Oracle
	Source code	Source code fully owned by UQU IT development team
	Database	Currently fully supported by Oracle
Business	Validity	SIS currently represents the core business of UQU
	Modeling	No business modeling technique used at the moment
	Documentation	No documentation of the business – just high-level descriptions of some aspects.
Architecture	Application integration	Ad-hoc interaction between applications using mainly glue-code
	Consumption	- Web application for users (e.g. students, faculties). - Desktop application on 6i forms for business owner
	Extensibility	The system cannot be extended by adding plug-ins to it. Extension must be done in the core functional classes.
	Interoperability	Conducted at the DB level through procures calls and triggers
	Data integration	Currently no data-centralized warehouse, and reports are generated manually by writing queries
	Style	Client-Server
Technology	Vendor	Oracle
	Version	Varies according to layers: - Database: Oracle 11g - Backend: Oracle 6i - Frontend: JSF 2.0
	License	Yearly Subscription for support only
	Data center	Good support for virtualization

From a technology point of view, UQU seems to be in good shape as it currently has many modern environments, such as Oracle 11g for databases and JSF 2.0. However, its back-end systems technology is almost obsolete and may shortly lose support from Oracle. Licences do not seem to be an issue as the IT department only pays yearly subscription fees for support during high-traffic seasons such as registration and admission. Apart from this, no other licences are paid as SIS is completely owned by the university. All the applications in the university are hosted in a centralized data centre that fully supports virtualization.

VI. Discussion

Based on the analysis described in the previous section, we have identified some strong points and weaknesses that the CEO must be aware of in order to make a decision regarding the system. The system's strong points can be summarized as follows:

- The current SIS satisfies most of the functional requirements of the UQU business owner, and the system is used frequently and effectively during the academic year.
- UQU currently has full control over the system. It can add new features or even disable unnecessary ones without the need to report back to any vendor.
- The UQU team fully understands the business logic of the system and can deal with any business-related modifications required by the business owner. Moreover, the team is skilled enough to modify and build new features smoothly.
- UQU owns a state-of-the-art environment in which to host different applications. Moreover, the current SIS complies with the new application of the server's technologies, as the university is using a WebLogic application server at the moment.
- The database technology is up to date and fully supported by Oracle.
- No costly licences need to be paid by UQU, as the system is fully owned by the university.

On the other hand, weaknesses in the current SIS environment are derived mainly from shortcomings in the overall environment with respect to emerging e-business needs, as described in [9]. We observed that these characteristics might significantly influence the CEO's decisions. The weaknesses can be summarized as follows:

- There is a lack of web-service capabilities for the back-end systems.
- The look and feel of the system is somewhat basic, as Oracle 6i forms do not support user-friendly interfaces.
- Report generation is difficult and time consuming as it involves specifically requesting that the IT department generates a report, due to the lack of tools and environments to support report generation. Consequently, some of the university's business processes might be negatively affected by this delay.
- The functional screens are not integrated properly into the workflow of the business process.
- The architectural style that is used by the university is somewhat out of date, and this type of client–server style is only normally used nowadays in small organizations. Currently, most of the enterprise solutions on the market have moved towards service-based architecture such as SOA [10] in order to ensure that their environments are extensible to accommodate future development; thus, UQU seems to be lagging behind in this dimension.
- Business logic is embedded inside Oracle forms. This can hinder flexible integration patterns with other applications or environments. Moreover, extending functionality might be an extremely time-consuming task, as the entire application must be put offline for full recompilation.

We used a simplified version of the weighted decision-making grid (WDMG) to help derive a more accurate decision in light of the strong and weak points identified. The weights given for each dimension were generated based on the importance of that dimension to stakeholders, including the business owner (with a 40% overall score weight), the IT team (with 35%), the university administration team (with 15%), and key users (with 10%). Table II lists the different dimensions and their assigned weights, out of 5, for each aspect.

TABLE II.
Weighted Decision-Making Grid

Pros	Score	Cons	Score
System functionality	5	Lack of web-services	3
Modification delivery time	5	Usability problems	3
Team support	4	Report generation	3
Technical Requirements	2	Architectural Style	2
DB technology	3	Oracle 6i problems	3
License	3	Business process problem	5
Overall	**22**		**19**

The table shows that the advantages of the current system outweigh its potential disadvantages by about 3. There is an interesting trade-off between "system functionality" and "business process problem", as both recorded a score of 5. The former is concerned with the basic functionality that the system can provide in terms of screens, while the latter relates to placing a functional screen into context via a pre-defined workflow. Both dimensions are significant and complement each other. However, we believe that the implementation of functionality is a prerequisite for implementing workflows. Thus, satisfying functional requirements is more significant at this stage than integrating applications with workflows, as the integration is currently done at the

database level and seems sufficient to fulfil UQU requirements during high seasons.

In general, although the difference in the overall scores seems tiny, the advantages have more weight. This might give a good indication that the current system should be kept in business and that the CEO needs to start a number of projects to address the disadvantages of the system. In doing this, UQU would be able to obtain not only a fully functional system but also an enterprise environment that can accommodate emerging business needs.

VII. Conclusion and Future Work

This paper has described our research work and experimentation for evaluating the legacy system at UQU. The key outcomes of this study should be useful to help the CEO of the university to make an informed decision about the ability of the current SIS to serve its mission and subsequent objectives. The provisional evaluation of SIS at UQU was in favour of re-architecting the system rather than replacing it with a brand new one. The next step in this work is to investigate a possible model for modifying legacy systems in order to comply with emerging architectural standards. As such, a study could be utilized to build a more solid plan for the actions to be taken in the development of SIS at the university.

Acknowledgment

The author wishes to thank the IT deanship at Umm Al-Qura University (UQU) for supporting part of this research and helping to provide necessary information to conduct the field study.

References

[1] Umm Al-Qura University, http://www.uqu.edu.sa [Accessed 5 Oct 2012].
[2] J. McGee, "Legacy Systems: Why History Matters", *Enterprise Systems Journal*, Dec 2005.
[3] L. Bass, P. Clements, and R. Kazma, *Software Architecture in Practice*, 3rd Edition, SEI Series in Software Engineering, 2012.
[4] I. Summerville, *Software Engineering*, 2010.
[5] J. Ransom, I. Sommerville, and I. Warren, "A Method for Assessing Legacy Systems for Evolution", *IEEE Proceedings of the Second Euromicro Conference on Software Maintenance and Reengineering*. March 8–11, 1998.
[6] L. Grace, M. Edwin, S. Dennis, and S. Soumya, *SMART: Analyzing the Reuse Potential of Legacy Components in a Service-Oriented Architecture Environment (CMU/SEI-2008-TN-008).* Software Engineering Institute, Carnegie Mellon University, 2008.
[7] R.C. Seacord, D. Plakosh, and G.A. Lewis, *Modernizing Legacy Systems.* Boston, MA. Addison-Wesley, 2003.
[8] K. H. Bennett, M. Ramage, and M. Munro, "Decision model for legacy systems", *IEE Proceedings – Software* 146(3): 153–159, 1999.
[9] B. Alkazemi, A. Baz, and G. Grami. "Toward an Architectural Model to Facilitate Adopting E-Government Business Model", *Proceedings of the 3rd International Conference on e-business*, Hong Kong, 2012.
[10] T. Erl. *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall, 2005.

Dr **Basem Y. Alkezemi** is the currently holding the position of vice-dean of Umm Al-Qura University's IT Deanship for E-Government. He received his Bachelor degree in Electric and Computer Engineering in 1999. He then went to study his MSc and PhD in Software Engineering at Newcastle University in the UK which he received in 2004 and 2009 respectively. Dr. Alkazemi has published a number of articles in regional and international journals and participated in many specialized conferences around the world. His main research interests are in software engineering, wireless sensor networks, computer supported education, and e-government.