Migration of Legacy Information Systems

Teta Stamati
National and Kapodistrian University of Athens, Greece

Panagiotis Kanellis
National and Kapodistrian University of Athens, Greece

Konstantina Stamati
National and Kapodistrian University of Athens, Greece

Drakoulis Martakos
National and Kapodistrian University of Athens, Greece

INTRODUCTION

In recent years, the accelerated competition in the global marketplace rendered the corporate environment more volatile than ever. The businesses are heavily relying on technological advancements to deliver a vast array of initiatives across a variety of industries. The firms' main partner in this increasingly complex and unpredictable journey is considered to be their information systems. Although the relevant industry offers an unprecedented rate of technological innovations, nevertheless there are cases where the information systems carry significant baggage from the past (Kelly, Gibson, Holland, & Light, 1999). There are aged systems that often form the central hub of the information flow within the organisation and are responsible for consolidating information about the business (Bisbal, Lawless, Wu, & Grimson, 1999; Sommerville, 2001) and thus they are called *mission-critical legacy information systems*.

The term "Legacy", according to the Oxford Dictionary, refers to any long-lasting effect of an event or process. The Legacy System describes an old system that remains in operation within an organisation. These systems often represent a massive, long-term business investment. Ulrich (1994) defined them as "stand-alone applications built during a prior era's technology, but they are perhaps more widely understood as software systems whose plans and documentation are either poor or non-existent" (Connall & Burns, 1993). Bennett (1995) referred to the legacy systems as, "large software systems that we do not know how to cope with but that are vital to the organisation", while Brodie and Stonebraker (1995) as "any information system that significantly resists modification and evolution to meet new and constantly changing business requirements". Finally, O'Callaghan (1999), drawing on the characteristics of legacy systems, described them as "a large system delivering significant business value today from a substantial pre-investment in hardware and software that may be many years old. Characteristically, it will have a long maintenance tail. It is, therefore, by definition a successful system and is likely to be one that is, in its own terms, well engineered. It is a business critical system which has an architecture which makes it insufficiently flexible to meet the challenges of anticipated future change requirements."

Legacy systems as a subject area is often overlooked in favour of areas such as new technology developments and strategic planning of information technology. In this context, the following sections present an overview of the legacy information systems problems in terms of their scale and definition. The legacy system issues include the required man-effort and costs of maintaining and evolving existing systems and the current methods of migrating complex legacy systems to new technology. It is shown that legacy systems present a critical area of study in both software engineering and business information systems. Taking into account that the role of technology is not merely supportive but affects the way enterprises conduct their business, it is shown that it is outdated to consider the migration process as the simple replacement of aged or problematic hardware and software. Thus, the migration should be approached as a planned change process that first and foremost requires an understanding and a methodology that covers the range of issues and organisational entities involved.

BACKGROUND

Legacy Information Systems

O'Callaghan (1999) refers to the adoption of an informational culture within the organisations in which "point solutions" were developed due to the widespread use of computer technology over several decades. There are cases where different divisions of the same organisation have developed individual applications in order to meet their perceived needs

in an application-by-application basis (O'Callaghan, 1999). In a similar way, there are applications in the same company that are running on different operating systems. Subsequently, such "point solutions", according to O'Callaghan (1999), became subject to localised optimisation, and uncontrolled maintenance, exacerbating the position further (Zou & Kontogiannis, 2002). These applications are unambiguously hard to maintain, improve, and expand because there is a general lack in their understanding. In addition, integration with newer systems may also be difficult because new business software may use completely different technologies (Wu, Lawless, Bisbal, Grimson, Wade, O'Sullivan, & Richardson, 1997). Due to the aforementioned reasons, there is a significant number of software engineers and practicing managers that consider the legacy systems to be potentially problematic (Bisbal et al., 1999).

On the other hand, according to Brodie and Stonebraker (1995), legacy systems do not always fit this stereotype. They propose that if a system was recently developed but cannot be readily modified to adapt to the constantly changing business requirements, then such a system can be regarded as a legacy system. Similarly, Randall (1999) stresses that "Legacy" is not just a problem encountered by organisations with aging mainframes and dated software, it is an issue from the moment a computer system becomes an integral part of any organisation's work (Randall, 1999).

The common rule is that if the legacy systems cannot support the business requirements, the business will not be able to remain competitive for long (Brodie & Stonebraker, 1995). Both a significant budget and person-hours will be monopolised by legacy systems maintenance. De Palma and Woodring (1993) referred to over 40% of the IT costs within an organisation being spent on maintaining its legacy systems, while Brodie and Stonebraker (1995) considered that the process of keeping these systems running takes 80-90% of the IT budget. Slee and Slovin (1997) gave an estimation in the area of 80% just for the routine maintenance activities.

Migration of Legacy Information Systems

The common sense solution to the legacy problem is migration. The definition of a successful information system migration according to Brodie and Stonebraker (1995) is as follows: "it begins with a mission-critical legacy system of a significant size in full operation and it ends with a fully operational, mission critical target application (or applications components) that replaces the essential aspects of the original legacy system." This involves replacing the problematic hardware and software, including the interfaces, applications, and databases that compose an information system infrastructure. Brodie and Stonebraker (1995) claim that legacy information system migration involves starting with a legacy information system and ending with a comparable target information system. This target system is significantly different from the original, but it contains substantial functionality and data from the legacy system.

The target system must be built using technological advancements in place of the legacy technology. For practical reasons, the target system may contain legacy components for which there is no adequate justification for their migration. According to Bisbal et al. (1999) the essence of legacy system migration is to allow the organisations to move their legacy systems to new environments, retaining the functionality of existing information systems without having to completely redevelop them.

In recent years, a significant number of organisations have initiated large-scale migration projects in order to improve their operations performance and to be compatible with the latest technological advancements. Particularly, the introduction of the Web browser technology, the so-called first-wave of the Internet (Dreyfus, 1998), forced the organisations to undertake migration projects in order to exploit the benefits of the shared information resources (Zou & Kontogiannis, 2002). Afterwards, the convergence of the Web and the distributed-object technologies extended the information Web-based applications to the services-based worldwide applications which was referred to as the Internet's second-wave (Dreyfus, 1998), and where the provided services and the content were distributed over the Internet (Zou & Kontogiannis, 2002). Moreover, the object-oriented technologies provided some valuable tools for the realisation of the services-based Web due to their inherent properties of encapsulation, polymorphism, and specialisation. In addition to the object orientation as a design paradigm, n-tier object computing was gradually being adopted by organisations as the preferred architecture for distributed applications because it allowed for the clear separation of business logic, representation logic, and back-end services (Zou & Kontogiannis, 2002).

Considering the case where the stakeholders of a large organisation (for instance, in the banking sector) have decided to maintain organisation's competitive edge and achieve conformation to the requirements posed by the need for flexibility and the minimization of time to market, the migration of company's legacy systems towards a new operating Web-based environment will be a considerably effective system evolution strategy. The strategic objectives will focus on leveraging the existing legacy software assets while minimising the risks involved in implementing from scratch its large scale mission-critical legacy applications (Umar, 1997). Thus, following the new era of distributed component-based applications, the organisation will face pressures to evolve its existing system (for instance, a large mainframe) in response to its customer expectations. The transformation from the mainframe computer to a multi-tier architecture will force the migration engineers to separate the integration logic and the legacy services to be stored in the middle-tier and the back-end tier, respectively. This architec-