

An empirical analysis of software life spans to determine the planning horizon for new software

William Richmond · Paul Nelson · Sanjog Misra

© Springer Science + Business Media, LLC 2006

Abstract This paper presents an empirical analysis of the life span of over 180 systems aimed at developing a model for determining the planning horizon for new software at the business case stage of software acquisition. At this early stage, the firm has limited knowledge about the project, but must make crucial decisions, such as scope (breadth of requirements), approach (both in-source vs. outsource and custom vs. package) and technology, including fit with standards (adhere to current vs. adopt new technology). These decisions are associated with different system lifetimes that, in turn, impact both the costs incurred and benefits received from the system. The failure to explicitly and properly address these differences can lead to the implementation of systems better left undone or to unintended consequences, such as the Y2K problem.

We find that technology and approach, but not scope decisions are strongly related to system lifetime. In particular, systems that use an operating system that conforms to the firm's standard or are acquired using a blended team entail longer system life. On the other

hand, shorter system life is indicated if the system is technically complex, custom developed or uses an older programming language. Furthermore, modified packaged software is shorter lived than is a vanilla package. In addition, environmental variables also impact the appropriate horizon. For example, as one would expect, strategic systems are used longer. On the other hand, somewhat surprisingly, systems sponsored by executives last less long and despite the quickening pace of technological and business process advancement, a small trend toward longer lived systems is uncovered.

Keywords Software · Planning horizon · Software life · Project justification

September 6, 2001: New operating software will be implemented across Kmart's supply chain beginning this quarter. . . . Kmart expects to record special charges . . . approximately \$ 130 million of the charges relate to the impairment of supply chain software and hardware that will no longer be utilized [14].

October 4, 1999: Motorola estimates the total cost for Y2K remediation from 1997 through 2000 will be in the range of \$250 million to \$300 million. These costs do not include estimates for potential litigation [5].

Every year billions of dollars are spent acquiring information systems. These investments can have unintended consequences as the lives of systems are

W. Richmond (✉)
Computer Information Systems, Western Carolina
University, Cullowhee, NC 28723
e-mail: brichmond@email.wcu.edu

P. Nelson · S. Misra
William E. Simon Graduate School of Business
Administration, University of Rochester,
Rochester, New York 14627
e-mails: {nelson, misra}@simon.rochester.edu

extended beyond their original projection—as with the Y2K problem—or as systems are used for shorter periods than expected—as with Kmart's supply chain management system. To minimize these financial and operational risks, Information Technology (IT) management must define the planning horizon (expected life span) for each potential system at the project initiation stage in order to accurately assess its business case. Since this time horizon defines the length of the expected cash flows derived from using the system, it directly influences the project's Go/No-Go decision. The planning horizon also influences budget decisions, since typically only 10% to 15% of a system's total cost pertains to its development and implementation—the rest covers the maintenance and upgrading of the system over its life [8].

This paper presents an empirical investigation of 181 systems that looks at how factors both outside of and under an IT manager's control are related to the length of a software system's life. In particular, environmental variables such as application area characteristics, organizational differences and technology advances affect a system's life span but are outside of the IT manager's purview. More importantly, management can shorten or lengthen system life through its choice of the project's scope (breadth/complexity), approach (insource vs. outsource and custom vs. package) and technology used (technical complexity and whether or not to adhere to the firm's technology standards).

The paper proceeds as follows. First, a model depicting the factors related to system life is presented that is appropriate for use when evaluating a system's economic feasibility. This development also briefly discusses related literatures concerning software performance, maintenance and replacement. Next, a series of hypotheses are presented relating to how a system's life is influenced by factors under IT's control—system scope, approach and technology. This is followed by a description of our survey data covering the life spans and system characteristics of a wide variety of replaced or retired systems as well as systems still in use. The influences of environmental variables outside of IT's control also are discussed. Econometric analysis is used to validate the hypotheses and identify the key correlates with observed life spans. We estimate a model relating the above design and development decisions as well as environmental characteristics of actual systems to their observed length of use. This allows us to identify the appropriate planning horizon for any particular

proposed system as well as investigate the relationship of early project design and development decisions with this horizon. In sum, we offer an intuitive view of what is associated with a longer or shorter expected life span for a proposed system, and provide an easily utilized methodology to estimate this time period. We conclude with a summary of our findings and their managerial and academic implications.

1. Model of software life

System evaluation is one of the steps within the Information Systems Planning Process—the process of defining the portfolio of systems an organization uses to achieve its business objectives [4]. The system planning literature finds that determining an appropriate planning horizon significantly affects the quality of the planning process and the resulting plans [20, 21]. In particular, the organizational factors examined in the system planning literature have been incorporated into earlier cost-benefit models that address when to rewrite/modify software [3, 7, 9], with the aim of modeling the economic tradeoff between the cost of maintaining existing software and the cost of rewriting the software.

Similarly, when making a new software investment decision, firms typically conduct a cost-benefit analysis among alternative solutions in order to determine the most attractive system for installation. Depending on the firm and situation, this analysis ranges from quite qualitative to very quantitative and from cursory to exhaustive. Qualitative approaches include Weill and Vitale [1] and Swanson [27]. Quantitative approaches typically monetize the criteria evaluated in the qualitative approaches and then utilize expected cash flow concepts such as net present value [18], return on investment [12], earned value analysis [23] and option pricing [29] to identify the best alternative. In whatever manner the analysis is conducted, the expected system life—the length of time the organization expects to receive benefits from and incur costs for the system—is crucial to the evaluation.

All cost-benefit analyses essentially are based on the premise that the firm: (a) has a set of perhaps changing business functions it wishes to carry out over time; (b) articulates, through a needs analysis, the requirements needed to carry out these changing functions; (c) assesses how well each particular alternative can fulfill

the desired requirements and how much each will cost to operate (i.e., maintain and upgrade) over time; (d) compares the net benefits or cash flows over time for each alternative software solution (the value pertaining to how well each performs the desired requirements less operating costs) to its upfront acquisition cost (i.e., development, installation and conversion costs); and (e) chooses the alternative for which a weighted sum of these net benefits relative to the acquisition cost is largest. In the net present value (NPV) approach used in this paper, the weights used to sum the net benefits depend on the time value of money and, in such, the weighted sum equals the sum of the present values of the net benefits. This approach is used in a wide variety of academic literatures including economics, finance and information technology, and commonly is recommended and used to make IT decisions [23]. It is formally used by two of the firms in our sample.

As an example, consider a firm that is pondering implementation of a system to support telemarketing. The system will identify who to call and which products to pitch as well as provide a script for the call. The NPV for the system is: $NPV = \sum_{t=0}^{\text{End of Life}} \frac{\text{Benefit}_t - \text{Operating Cost}_t}{(1 + \text{Cost of Capital})^t} - \text{Acquisition Cost}$. If the Acquisition Cost is \$ 1,000,000, the Net Benefit ($NB_t = \text{Benefit}_t - \text{Operating Cost}_t$) is \$ 200,000 in all periods t , and the Cost of Capital is 10%, then the project has a positive NPV only if the system life is greater than seven years. Hence, to make a well informed decision about whether or not to install a particular system the firm needs a good estimate of system life—that is, how long the system is expected to outperform replacement alternatives (i.e., when planned obsolescence occurs or when its functions are more cost effectively performed by a replacement software system or manual labor).

Figure 1 depicts the cost-benefit analysis that determines a proposed system’s expected life. This figure is meant to be illustrative. In actuality, the relationships in the figure are likely to be non-linear. The only assumption needed for the theoretical discussion that follows is the net benefit from replacing the system, ΔNB_t , minus the upfront cost of replacing the system, $A_t^{\text{alternative}}$, increases with time. This is reasonable, since over time the firm’s business needs change due to an evolving business environment. This necessitates system modifications that typically result in increased operating costs and less effective performance of the desired requirements (Laws 1 and 7 of software evolution [16]).

In addition, since a replacement system can be designed to specifically address the changed requirements, it will perform the requirements more completely and/or with less operating expense. As a result, at each time t following the installation of a proposed system, the superiority of the best replacement alternative’s net benefits (and, hence, their weighted sum or present value, $NB_t^{\text{alternative}}$) is expected to increase relative that of the proposed system (NB_t^{proposed}).¹ The difference in these present values $\Delta NB_t = NB_t^{\text{alternative}} - NB_t^{\text{proposed}}$ depicts the benefit of switching at time t and is depicted in Fig. 1.

When the expected benefit of switching (ΔNB_t) finally exceeds the expected cost of switching, replacement of the proposed system is expected. The cost of switching is the upfront acquisition cost of the replacement system (development, implementation and conversion costs) and is represented by $A_t^{\text{alternative}}$ in Fig. 1.² That is, the expected system life for the proposed system corresponds to the time T in Fig. 1 when ΔNB_t first exceeds the cost of switching, $A_t^{\text{alternative}}$.³ If ΔNB_t increases more slowly over time, the proposed system’s life span is longer. If the upfront acquisition cost of the replacement system, $A_t^{\text{alternative}}$, is reduced or declines more quickly over time, a shorter system life is indicated.

What factors, then, affect ΔNB_t and $A_t^{\text{alternative}}$? While Fig. 1 provides the intuition behind the expected longevity of a proposed system, our interest is in how design and development decisions as well as environmental characteristics are related to the length of this time span. In particular, at the project initiation stage of the software acquisition process, IT managers have limited information upon which to either base their estimate of a proposed system’s life or decide upon the system characteristics for a system with a particular desired life span (planned obsolescence). Some environmental factors, such as the application area and sponsor, are easily observable and are likely tied to

¹ The present value of the alternative’s net benefits, $NB_t^{\text{alternative}}$, excludes the onetime cost of developing, implementing and switching to the system. These costs are captured separately by $A_t^{\text{alternative}}$ and discussed later.

² Like ΔNB_t , the acquisition cost of the alternative, $A_t^{\text{alternative}}$, may be non-linear. It even may increase over time.

³ Note that it may require years to develop or acquire a replacement system. Therefore, the firm is advised to start the replacement process well before time T .

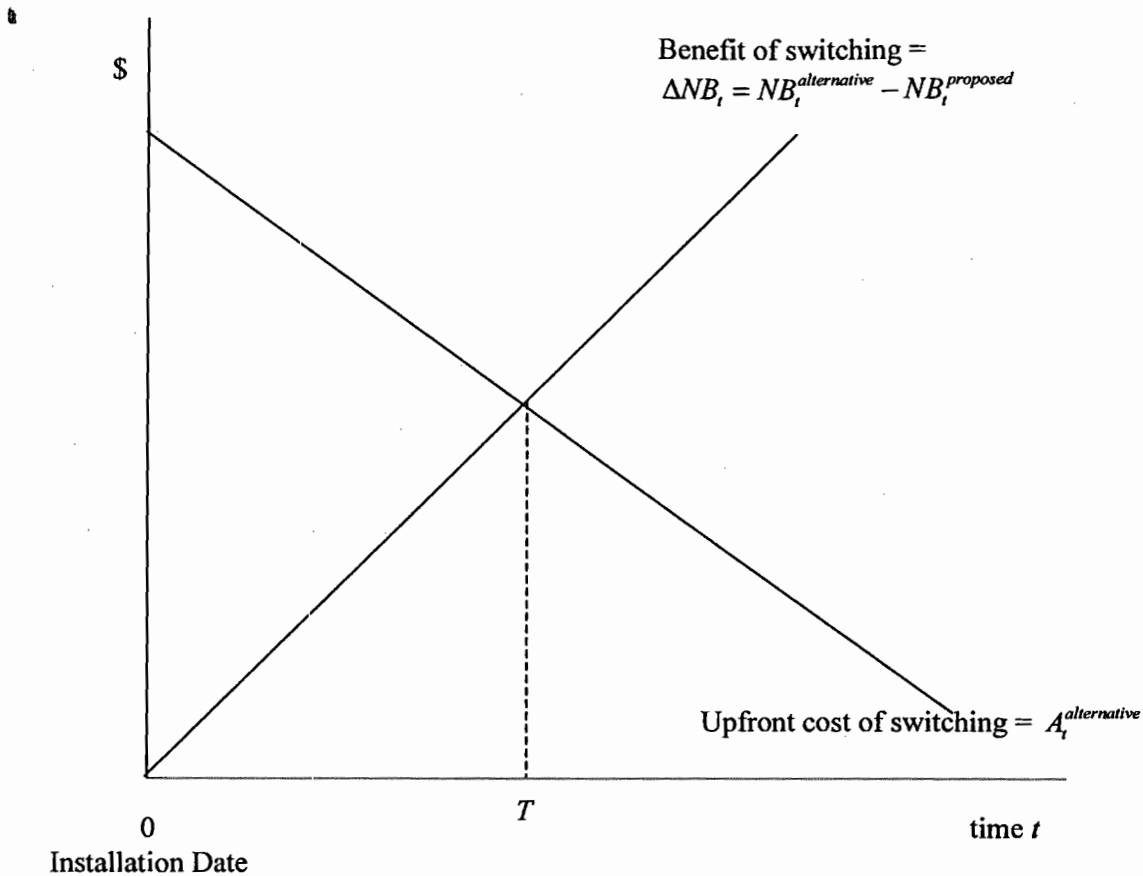


Fig. 1 Benefit of switching vs. upfront cost of switching determines planning horizon

ΔNB_t and $A_t^{alternative}$ and, hence, system life. More important from an IT manager's perspective are system scope, approach and technology decisions that also may be tied to the level or change over time in ΔNB_t and $A_t^{alternative}$. These scope, approach and technology issues are outlined in Fig. 2 and described below. Note that the relationship between these managerial decisions and system lifetime is of importance regardless of which is identified first. Whether these choices provide an expected lifetime or are chosen with a specific lifetime in mind, the relationship between these choices and lifespan is the same.

1.1. System decision factors: Scope

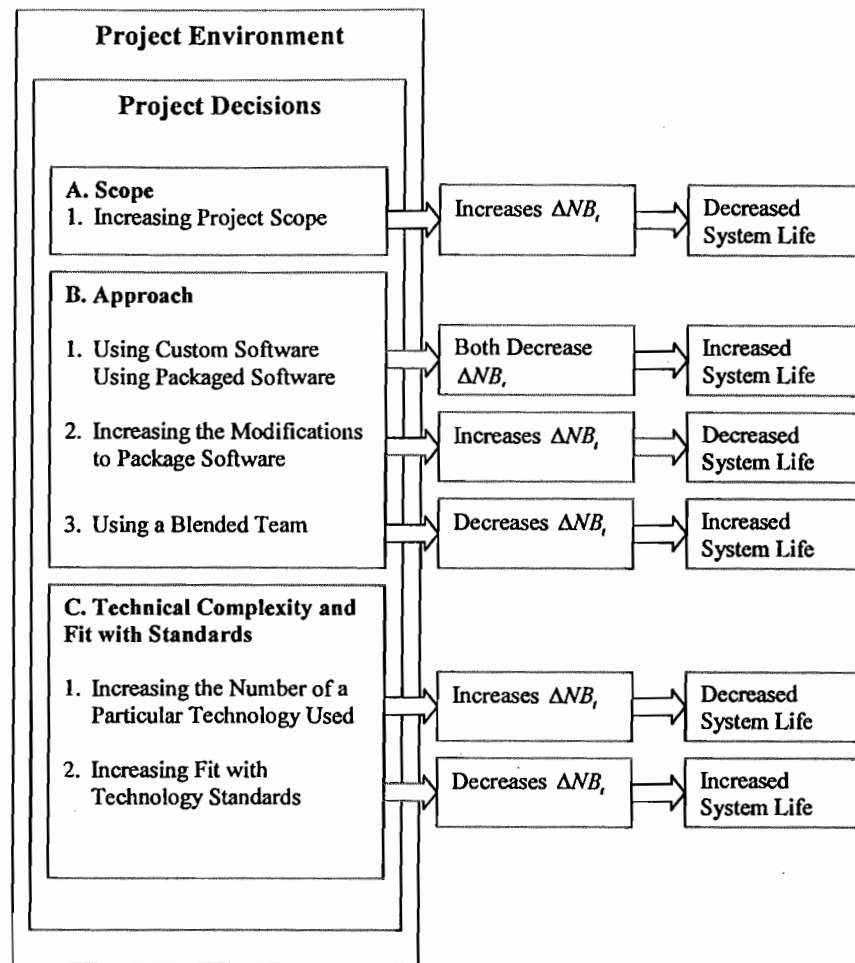
At the project initiation stage, a decision is made about whether to address the set of desired business functions using a combination of small systems or a single more complex system. Clearly the number of requirements supported positively impacts the value of the system as

well as the amount of maintenance needed as the system's requirements change over time. As modifications are made, the installed system's ability to perform the revised requirements deteriorates and/or its operating costs rise [1, 16]. It follows that the greater the system's scope, the faster is this deterioration. Hence, the net benefit of switching from the proposed system (ΔNB_t) increases more quickly over time (see Fig. 2.A.1) and, thereby, shortens system life.

1.2. System decision factors: Approach

The approach the firm takes to acquire a system is central to its design and development. It is generally seen as two dimensional [19]. The first dimension concerns the system's source—whether to custom develop the system, purchase packaged software or purchase and modify packaged software. This decision, in turn, defines the scope and nature of the work necessary to implement the system. Custom developed software

Fig. 2 Impact of project decision variables on system life



typically entails requirements gathering, system analysis, system design, coding, testing and implementation among other tasks. Implementing packaged software requires requirements gathering, package evaluation and selection, testing and implementation and so on. Modified packaged software entails both the software selection process and the process of identifying, designing, coding and testing changes to the packaged software. Note that this requires an in-depth understanding of the internals (code) of the packaged software.

The second acquisition approach dimension corresponds to who does the design and installation work at the firm—an internal, external, or blended team that utilizes both internal and external resources. This acquisition team decision concerning who does the work is independent of how the software is acquired. If the software is custom developed, this development can be done in-house, be outsourced to a firm such as Perot

Systems or people from both the outsourcer and the client can work on the project. In reality, this dimension is a continuum based on the relative effort of the outsourcer (if any) and the firm. Similarly, for packaged and modified packaged software, the team doing the work can be internal, external or a combination although, as mentioned previously, some of the tasks the team executes may differ from those in a custom development. Note that with packaged software, the initial development of the software is done by an external entity (which is captured by the other approach dimension).

These approach decisions are important because the better designed and developed a system is, the better it fulfills its desired business requirements and the more maintainable it is. A system is at an immediate disadvantage if design, development and implementation faults interfere with its ability to satisfy

the firm's (possibly undefined) requirements.⁴ In this case, an alternative system (one that does efficiently carry out the requirements) has a greater net benefit even at time 0 when the proposed system is installed. That is, for a poorly defined and developed system, ΔNB_t in Fig. 1 either may not equal zero at time 0 or may increase more quickly over time. This shift causes the intersection of ΔNB_t and $A_t^{\text{alternative}}$ to occur earlier (i.e., T is reduced). Therefore, the proposed system's expected length of use is shortened.

Custom developed software is likely to more closely match the desired business requirements than is packaged software since it can be tailored to match the firm's business process idiosyncrasies. Also, once written, the development team knows the system internals which should aid maintenance efforts. Together, these reduce the benefit of switching to a new system (i.e., decrease ΔNB_t) and, thus, lead to longer system life (Fig. 2.B.1). On the other hand, the functionality of packaged software matches industry standard practices which the firm may wish to adopt or already use. In addition, the vendor performs at least some of the packaged software maintenance (all for vanilla/unmodified implementations), and, due to scope economies, the vendor has a greater incentive to make changes to its software in a way that does not degrade either performance or the ability to make future changes. Each of these decreases ΔNB_t and, hence, is associated with longer system life (Fig. 2.B.1). Consequently, whether custom or packaged software has a stronger relationship with longer system life is an empirical question tied to the relative importance of firm specific knowledge versus industry-wide knowledge and technical know-how.

Since packaged software may not meet a company's exact business requirements, many firms purchase packaged software and then modify it to meet their needs. These modifications can be as simple as changing report headings or as complex as major revisions in functionality. However, if a company modifies packaged software, the vendor's maintenance efforts may conflict with those of the company. Since the firm typically has both less ability and less incentive to make changes in a way that does not degrade performance or the ability to make future changes, utilizing modified

packaged software should increase ΔNB_t and result in a shorter system life (Fig. 2.B.2).

Previous research on the team dimension of the acquisition approach shows that the team implementing a project can have a profound impact on its success [22]. An external team will have worked on similar applications at other businesses and likely know new applications (e.g., ERP in the early 1990s and CRM in the mid 1990s) and technologies better than an internal team. An external team also is motivated by market discipline, giving it a powerful incentive to maximize the likelihood of project success. This will decrease ΔNB_t , resulting in longer system life (Fig. 2.B.3). Alternatively, an internal development team knows the business' idiosyncrasies better than an external development team. In addition, Wang et al. [30] argue that the incentives of an internal team may be better than for an external one. Hence, an internal team also should decrease ΔNB_t , resulting in longer system life (Fig. 2.B.3). As with the custom vs. package choice, whether an internal or external team has a larger impact on system longevity is an open question. The obvious solution is to use a blended team that provides the benefits of both approaches.

1.3. System decision factors: Technical complexity and fit with standards

During the technical feasibility assessment in the project initiation stage, the firm identifies the technologies that will underlie the system, including the programming language(s), database management system(s) and operating system(s). These decisions are either explicitly or implicitly linked to the firm's decision of whether or not to adhere to its existing technology standards. These technology choices also determine the system's technical complexity in terms of the number of programming languages, database management systems and operating systems used within the system.

Systems using multiple variants of the same type of technology—multiple programming languages, database management systems, or operating systems are more complex. System maintenance, therefore, requires knowledge of multiple technologies and the ability to manage each of them. This added complexity increases maintenance costs which increases ΔNB_t and, thereby, shortens system life (Fig. 2.C.1).

Alternatively, an application that adheres to the organization's existing technology standards will be easier

⁴ The Standish Chaos study [26] reveals that a significant percentage of systems fail for this reason.

to maintain since there is no learning curve for the IT department. In addition, multiple types of expertise are not required to support on-going operations and maintenance which lowers costs. Together, these influences decrease ΔNB_t , implying that adhering to technology standards is associated with longer system life (Fig. 2.C.2).

2. Hypotheses

We operationalize our general model of system life by defining explicit measures for each factor affecting the benefit of switching, $\Delta NB_t = NB_t^{\text{alternative}} - NB_t^{\text{proposed}}$, and the upfront cost of switching, $A_t^{\text{alternative}}$. In so doing, we derive seven hypotheses as to how these controllable factors are associated with a system's lifetime.

2.1. System decision factors: Scope

System scope has two dimensions—the number of functional areas and types of decisions (e.g., transaction processing and management information) supported. Since a proposed system that services multiple areas or decision types is more complex, its expected system life is shorter (i.e., ΔNB_t increases). This idea coincides with Swanson and Dans' [28] hypothesis that business functional complexity affects both a system's maintenance requirements and its remaining life. While they do not find support for their propositions, this may be due to their focus on installed systems' remaining lives, their use of managerial estimates for these time spans and the fact that they do not incorporate the type of the decisions supported in their analysis. We offer two hypotheses:

Hypothesis 1: Systems supporting multiple functional areas have shorter lives.

Hypothesis 2: Systems supporting multiple decision types have shorter lives.

2.2. System decision factors: Approach

Based on our model of system life, both a pure custom and a vanilla package sourcing approach indicate a longer system life. This is supported by previous research by Barry et al. [2] that finds packaged software lowers software volatility and, hence, should increase system life. As with the sourcing decision, both an in-

sourced and an outsourced team approach provide a longer system life. Logically, then, a team which blends the advantages of both should imply even longer system life. Concerning modified packaged software, our model unambiguously predicts that the more a software package is modified, the shorter is system life. Correspondingly, we propose:

Hypothesis 3: Packaged software is associated with longer system life.

Hypothesis 4: The more packaged software is modified, the shorter is system life.

Hypothesis 5: A blended development team using both internal and external resources is associated with longer system life.

2.3. System decision factors: Technical complexity and fit with standards

The impact of using multiple technologies on system life has not been studied, nor has the impact of adhering to technology standards. Based on our model, we expect that using more than one database, programming language or operating system will increase a system's technical complexity, which will shorten system life. We also expect that applications that adhere to the firm's technology standard—that use the same programming language, database and operating system as other systems at the firm—will have longer life. Correspondingly:

Hypothesis 6: Systems utilizing multiple variants of the same type of technology have shorter lives.

Hypothesis 7: Adherence to a firm's technology standards is associated with longer system life.

3. Data

We validate our seven hypotheses using survey data and a technology index. The survey data include system characteristics for a wide variety of replaced systems and a greater number of systems still in use. In particular, for five companies descriptive characteristics for every documented software system that each firm had installed were ascertained. Personal interviews by the authors with IT managers at each company were used to collect the data. The manager, with assistance

Table 1 Descriptive statistics for explanatory variables

Relevant Topic	Variable	Variable definition	Mean ^a	Minimum	Maximum
H1	<i>NumArea</i>	Number of functional areas supported	1.22	0	3
H2	<i>NumType</i>	Number of decision types supported	1.45	0	3
H3	<i>Package</i>	Packaged software used	0.18	0	1
H4	<i>PctMod</i>	Percent that packaged software is modified	0.25	0	.90
H5	<i>TeamBoth</i>	Acquisition team used both internal and external Resources (0/1)	0.17	0	1
H6	<i>NumDBMS</i>	Number of DBMS used	0.57	0	2
H6	<i>NumDBMS2</i>	Number of DBMS used squared	0.66	0	4
H6	<i>NumPL</i>	Number of programming languages used	1.22	1	4
H6	<i>NumOS</i>	Number of operating systems used	1.06	1	3
H7	<i>PctPL</i>	Percent of firm's systems using same programming language	0.28	0.01	0.92
H7	<i>PctDBMS</i>	Percent of firm's systems using same DBMS	0.45	0.01	1
H7	<i>PctOS</i>	Percent of firm's systems using same operating system	0.58	0.01	1
Application	<i>Finance</i>	Finance & accounting area supported	0.24	0	1
Application	<i>Strategic</i>	system is considered strategic	0.31	0	1
Organization	<i>Co1</i>	Company 1	0.10	0	1
Organization	<i>Co2</i>	Company 2	0.20	0	1
Organization	<i>Co3</i>	Company 3	0.09	0	1
Organization	<i>Co4</i>	Company 4	0.07	0	1
Organization	<i>Sponsor</i>	System Sponsored by an executive	0.36	0	1
Time	<i>Installed</i>	Log of date system installed (Julian date)	10.37	10.13	10.44
Time	<i>InstalledSquared</i>	Log of date system installed squared (Julian date)	107.52	102.54	108.95

^aMean value of the variable over the 181 sampled systems.

when needed by the authors, filled out a survey for each particular system. Managers also referred to system documentation in order to ascertain such things as implementation and cancellation dates. Most questions were objective (e.g., what development approach was used—vanilla package, modified package, custom development?). For the few subjective questions asked (e.g., is the system strategic?), the IT manager's opinion was recorded. Table 1 provides descriptive statistics for the data.

Data were captured for 181 systems. These systems ranged from small personal computer applications to large distributed systems and from common accounting systems to strategic operational systems. The firms do business in a broad cross-section of industries (a regulated utility, a long-distance telephone company, a payroll processing company, a rocket engine manufacturer and a building products manufacturer)

and vary significantly in size (annual revenues range from \$20 million to \$900 million).

The unit of analysis is the system. For each system, the respondent was asked when the system was installed and whether or not the system was still operational as of June 1, 1994. If the system had been replaced, the date it was canceled was ascertained. Installment dates ranged from February 1970 to June 1993. Thirty-three systems had been retired. For the retired systems, the elapsed time between its retirement date and the initial installation date provides the system's actual life in days. For those systems still being used, we know that their lives will equal or exceed the difference between the cut-off date, June 1, 1994, and their installation date. This "censoring" of the lifetime data for the systems still in use is explicitly incorporated into the empirical analysis.

The survey data include measures related to the scope, approach and technology decisions as well as environmental variables. Each hypothesis [H1–H7] corresponds with one or more of these variables, as do the environmental control characteristics.

The number of functional areas supported [H1] is measured using a simple count of the number of areas supported (*NumArea*). Six functional areas are measured: marketing, sales and customer service; distribution, operations and engineering; accounting and financial; human resources, training and payroll; information systems; and other. Other includes office automation as well as any application area that did not fit into the other categories. The second scope related issue—the number of decision types supported [H2]—also is measured by a simple count (*NumType*). The five possible decision types include Transaction Processing (TPS), Management Information (MIS), Decision Support (DSS), Executive Support (ESS) and other.

The custom-package sourcing approach [H3] variable is a binary variable (*Package*) equal to one if packaged software is utilized and zero otherwise. If packaged software was used, the IT manager responsible for the system was asked what percent of the application was modified [H4] (*PctMod*). Most of the systems were custom developed (81%). Of the systems based on packaged software, 67% were modified with the percent of modification ranging from 5 to 90%. The average amount of modification was 25%.

The acquisition team approach [H5] is internal, external or both (blended). An internal team used only the company's IT resources to acquire the system. For an external team, the vendor was responsible for the project and no company IT resources were utilized full-time on the project. A blended team had both internal and external full-time resources as part of the project team. Correspondingly, a binary variable (*TeamBoth*) equal to one for a blended team and zero otherwise is used.

The first technology related issue (technical complexity)—the use of multiple variants of the same type of technology [H6]—is addressed using four variables. These variables consist of simple counts of the number of programming languages (*NumPL*), database management systems (*NumDBMS*) and operating systems (*NumOS*) used in the particular system. Because *NumDBMS* can be 0 if a file system such as VSAM is used, a squared term *NumDBMS2* is also included.

Thus, we capture both the move to a database technology as well as the complexity of this environment.

To assess how well the system fits with the firm's technology standards [H7], three variables are used which relate to the programming language, database management system and operating system used (*PctPL*, *PctDBMS* and *PctOS*, respectively). These percentage variables measure the percentage of a firm's systems that utilize the same technology (e.g., COBOL for programming language or Oracle for DBMS) as is used by the particular system in question.

3.1. Environmental factors: Application, organization and time

There are also numerous environmental factors outside of the project evaluation team's control that can affect system life. These include characteristics of the application itself as well as of the organization within which the system operates. The speeds of technological advancement and business change also play a part. Although not the focus of this study, these factors are discussed and modeled to lend face validity to the empirical results as well as to serve as covariates allowing for more accurate evaluation of the seven hypotheses.

An easily observed application characteristic is the area(s) supported. Some areas undergo more frequent and significant changes in their business requirements. This volatility is identified as a driver of both maintenance costs [2] and system replacement [1]. Specific functional areas, however, have not previously been identified as more or less volatile. Following Barry et al. [2], we distinguish between financial and non-financial systems using a binary variable (*Finance*). A second application characteristic is the system's perceived value. This can affect both project success (with more valued systems receiving more attention) as well as the level of maintenance effort. Correspondingly, higher value systems likely are longer lived. A binary variable (*Strategic*) indicating whether the IT manager viewed the system as strategic or not is used to proxy for perceived value. The assumption is that strategic systems are perceived as providing more value to the firm.

Binary variables (*Co1*, *Co2*, *Co3*, and *Co4*) capture idiosyncratic organizational differences across firms.⁵

⁵Because our empirical analysis utilizes an intercept term, for identification purposes one of the firm specific binary variables (*Co5*) is left out.

Different predispositions to replace systems may occur due to management style and quality differences (e.g., organizational structure changes or a tendency to follow management fads). Since executive sponsorship has been identified as a key driver of system success [26], a binary variable (*Sponsor*) identifying whether or not the system's sponsor was a company executive (director level or higher) also is utilized. Whether *Sponsor* is related to longevity, then, depends upon whether longevity is a key component of what is deemed success.

To take into account changes that occur due to pace of technological and business process change we also include two time variables (*Installed* and *Installed-Squared*) that specify the log of the Julian date on which the system was installed and this value squared, respectively. These dates range from February 1970 (25720) to June 1993 (34121) with the average date being May 1987 (31921). If the rate of change in the environment is increasing, as is claimed in *Future Shock*, then over time the systems will be replaced ever more quickly. Alternatively, if the technology and software development process changes are specifically aimed at enabling software to be more flexible, then software life will increase over time.

4. Empirical model

A straightforward approach to represent system longevity as a function of system characteristics is a multivariate regression model of the form:

$$lifetime_i = \alpha + \sum_{k=1}^K \beta_k x_{ik} + \varepsilon_i, \quad (1)$$

where $lifetime_i$ is the observed duration of the system i 's life. The x_{ik} are the K system design, development and environmental characteristics, while the β_k reflect the relationship of each characteristic k with system's life and α is a constant intercept term. In our situation, the x_{ik} pertain to the variables discussed in the data section. ε_i is an error term. Least squares minimization or maximum likelihood procedures are used to estimate α and the β_k .

Two problems, however, arise with this specification. First, the error term is normally distributed and, thereby, ranges from negative infinity to infinity.

Clearly, this assumption is not acceptable since a life can only be of positive duration. This necessitates an alternative assumption concerning the distribution of the error term ε_i . A number of options are available, including the Weibull, LogNormal, LogLogistic and Exponential.⁶

The LogNormal and LogLogistic distributions show positive duration dependence and then negative duration dependence, whereas the Weibull shows only positive duration dependence or only negative duration dependence depending on its scale parameter. Typically, the probability that a system is replaced, given that it has survived to time t , increases over time (positive duration dependence). There comes a point, however, when replacement becomes more difficult because the system is interwoven into the fabric of the organization and the knowledge of what it does and how it works is lost. At this point it is perhaps safer to muddle along with patches than to incur the risk of replacement. This corresponds to negative duration dependence. For this reason, we choose the LogNormal distribution.

To verify this choice, we conducted a specification test. We adopted a Bayesian model selection approach as suggested by Raftery [23] and Kass and Raftery [14]. This involves estimating the candidate models, computing the Bayesian Information Criteria (BIC) where $BIC_j = -2\text{LogLikelihood} + p \ln(n)$ and then obtaining approximate posterior model probabilities (Π_m) using the formula: $\Pi_m = \frac{e^{-\frac{1}{2}BIC_m}}{\sum_{j=1}^J e^{-\frac{1}{2}BIC_j}}$. The advantage of this approach is that it allows the testing of alternative non-nested specifications. Based on our analysis we found the LogNormal to be most appropriate. In particular, the posterior probabilities of each distribution are as follows:

Density	Posterior Model Probability
LogNormal	73.08%
LogLogistic	26.79%
Weibull	0.07%
Exponential	0.06%

⁶ Advanced econometrics textbooks such as Greene [10] and Lawless [17] provide details on the lognormal and other distribution as well as the other concepts which follow (censoring, likelihood functions and their maximization, etc.).

In addition, percentile-based probability plots of the LogNormal and LogLogistic distributions show the LogNormal has a better fit.

The second problem is that we do not observe “completed lifetimes” for every observation. That is, many systems were still in use when the data collection process terminated in 1994 and, hence, we do not observe the true life of the system—the system’s lifetime, as measured, is incomplete. This problem is termed “censoring” in the statistical literature. In order to deal with this problem we use a likelihood function that takes this into account.

This likelihood function considers two types of observations. If the true duration of system i is observed, then its likelihood is simply $f(\omega_i)$ which is derived as follows:

$$lifetime_i = \left[\exp \left(\alpha + \sum_{k=1}^K \beta_k x_{ik} \right) \right] \omega_i, \tag{2}$$

where ω_i is an error term with a lognormal distribution. This relationship “linearizes” through a log transformation into:

$$\ln(lifetime_i) = \alpha + \sum_{k=1}^K \beta_k x_{ik} + \ln(\omega_i) \tag{3}$$

Setting $t_i = \ln(lifetime_i)$ and since $\ln(\omega_i)$ is normal it can be written as σw_i , where w_i is a standard normal error and σ is a scale parameter (i.e., σ^2 is the variance). In other words, for those cases where the lifetime is observed we have:

$$t_i = \alpha + \sum_{k=1}^K \beta_k x_{ik} + \sigma w_i. \tag{4}$$

On the other hand, if the true duration is unobserved (censored) then all we know is that the system lasted longer than a certain length $t_i^* = \ln(\text{days between June 1, 1994 and the installation date of system } i)$. In this case, the likelihood is expressed as the probability that the system is used longer than t_i^* . That is, $P(t_i > t_i^*) = 1 - P(t_i < t_i^*)$, where:

$$P(t < t_i^*) = P \left(w_i < \sigma^{-1} \left(t_i^* - \alpha - \sum_{k=1}^K \beta_k x_{ik} \right) \right). \tag{5}$$

The likelihood pertaining to a particular system i thus depends on whether or not the measurement for the system’s life is censored or not. Utilizing a dummy variable δ_i that takes on the value one if the duration is censored and zero otherwise, the likelihood for system i is

$$L_i = \phi \left(\sigma^{-1} \left(t_i - \alpha - \sum_{k=1}^K \beta_k x_{ik} \right) \right)^{(1-\delta_i)} \times \left[1 - \Phi \left(w_i < \sigma^{-1} \left(t_i^* - \alpha - \sum_{k=1}^K \beta_k x_{ik} \right) \right) \right]^{\delta_i}. \tag{6}$$

In the above equation, ϕ and Φ are the normal probability density (PDF) and cumulative distribution functions (CDF), respectively. The likelihood for the entire data set consisting of N observations is then

$$L = \prod_{i=1}^N L_i. \tag{7}$$

We maximize Eq. (7) to find the relevant estimates of the scale parameter σ , the intercept α and the variable parameters β_k (see Eq. (6) for details on how each parameter is related to the log likelihood). Note that β_k represents the relationship of characteristic x_{ik} with system life, holding the levels of all the other variables constant. Specifically, we estimate a model that contains the following explanatory variables: *NumArea*, *NumType*, *Package*, *PctMod*, *TeamBoth*, *NumPL*, *NumDBMS*, *NumDBMS2*, *NumOS*, *PctPL*, *PctDBMS*, *PctOS*, *Finance*, *Strategic*, *Co1*, *Co2*, *Co3*, *Co4*, *Sponsor*, *Installed*, and *Installed-Squared*. Table 2 presents the correlations among these variables. Overall, the correlations among the variables are relatively weak. There are a few exceptions. As expected, the variables *Installed* and *NumDBMS* are highly correlated with their squares. The correlation between *Package* and *PctMod* is .676. This is not surprising since if *Package* is 0, then *PctMod* must also be 0. The fit with standards variables, *PctPL*, *PctDBMS*, *PctOS* are also correlated. We ran models with each pair of variables and each variable individually, but this did not affect the overall results, so we left all three variables in the analysis.

Table 2 Correlations among explanatory variables

	Num Area	Num Type	Num Pack-Mod	Pct Mod	Team Both	Num DBMS	Num DBMS2	Num OS	Num PL	Num DBMS	Num DBMS2	Num OS	Num PL	Pct DBMS	Pct OS	Finance	Strategic	Co1	Co2	Co3	Co4	Sponsor	Installed	Squared
NumArea	1.00	-0.10	0.02	0.06	0.14	-0.04	-0.05	-0.02	-0.01	-0.02	-0.01	-0.02	-0.01	-0.02	-0.18	0.09	0.14	-0.16	0.30	0.02	-0.08	0.07	-0.09	-0.09
NumType	-0.10	1.00	0.25	0.16	-0.04	0.01	-0.05	-0.07	-0.14	0.18	0.09	0.01	-0.15	0.00	0.16	-0.36	0.30	0.16	-0.36	0.30	0.18	0.06	0.11	0.11
Package	0.02	0.25	1.00	0.68	0.17	0.03	0.01	0.05	-0.03	0.30	0.21	0.05	-0.07	0.18	-0.01	-0.09	0.31	0.39	0.00	0.01	0.01	0.01	0.01	0.01
PctMod	0.06	0.16	0.68	1.00	0.19	0.08	0.02	0.23	0.07	0.20	0.07	-0.01	-0.07	0.28	-0.01	0.03	0.17	0.33	-0.01	0.00	0.00	0.00	0.00	0.00
TeamBoth	0.14	-0.04	0.17	0.19	1.00	0.05	0.02	0.17	0.13	0.03	-0.06	-0.12	0.23	0.25	-0.15	0.41	0.23	0.00	0.16	0.14	0.14	0.14	0.14	0.14
Num DBMS	-0.04	0.01	0.03	0.08	0.05	1.00	0.92	-0.01	0.02	0.30	0.22	0.28	-0.05	0.23	0.25	0.01	-0.24	0.24	-0.02	-0.11	-0.11	-0.11	-0.11	-0.11
Num DBMS2	-0.05	-0.05	0.01	0.02	0.02	0.92	1.00	-0.03	-0.01	0.19	0.13	0.19	-0.06	0.14	0.13	-0.01	-0.19	0.18	0.02	-0.20	-0.20	-0.20	-0.20	-0.20
Num OS	-0.02	-0.07	0.05	0.23	0.17	-0.01	-0.03	1.00	0.37	-0.04	-0.09	-0.07	-0.03	0.15	-0.07	0.34	0.07	-0.06	0.17	0.02	0.02	0.02	0.02	0.02
Num PL	-0.01	-0.14	-0.03	0.07	0.13	0.02	-0.01	0.37	1.00	-0.03	-0.09	0.02	0.27	0.23	-0.10	0.44	-0.13	-0.07	-0.05	0.20	0.20	0.20	0.20	0.20
Pct PL	-0.02	0.18	0.30	0.20	0.03	0.30	0.19	-0.04	-0.03	1.00	0.75	0.56	0.03	0.36	0.61	-0.02	-0.08	0.54	-0.27	0.10	0.10	0.10	0.10	0.10
Pct DBMS	-0.12	0.09	0.21	0.07	-0.06	0.22	0.13	-0.09	-0.09	0.75	1.00	0.60	0.00	0.28	0.60	-0.15	-0.03	0.35	-0.35	0.17	0.17	0.17	0.17	0.17
Pct OS	-0.18	0.01	0.05	-0.01	-0.12	0.28	0.19	-0.07	0.02	0.56	0.60	1.00	-0.09	0.09	0.49	-0.15	-0.34	0.25	-0.19	0.10	0.10	0.10	0.10	0.10
Finance	0.09	-0.15	-0.07	-0.07	0.23	-0.05	-0.06	-0.03	0.27	0.03	0.00	-0.09	1.00	0.32	0.03	0.46	0.05	-0.15	0.06	0.23	0.23	0.23	0.23	0.23
Strategic	0.14	0.00	0.18	0.28	0.25	0.23	0.14	0.15	0.23	0.36	0.28	0.09	0.32	1.00	0.10	0.42	0.00	0.30	-0.05	0.22	0.22	0.22	0.22	0.22
Co1	-0.16	0.16	-0.01	-0.01	-0.15	0.25	0.13	-0.07	-0.10	0.61	0.60	0.49	0.03	0.10	1.00	-0.17	-0.10	-0.09	-0.25	0.21	0.21	0.21	0.21	0.21
Co2	0.30	-0.36	-0.09	0.03	0.41	0.01	-0.01	0.34	0.44	-0.02	-0.15	-0.15	0.46	0.42	-0.17	1.00	-0.16	-0.13	0.06	0.18	0.18	0.18	0.18	0.18
Co3	0.02	0.30	0.31	0.17	0.23	-0.24	-0.19	0.07	-0.13	-0.08	-0.03	-0.34	0.05	0.00	-0.10	-0.16	1.00	-0.08	0.09	-0.03	-0.03	-0.03	-0.03	-0.03
Co4	-0.08	0.18	0.39	0.33	0.00	0.24	0.18	-0.06	-0.07	0.54	0.35	0.25	-0.15	0.30	-0.09	-0.13	-0.08	1.00	-0.15	-0.19	-0.19	-0.19	-0.19	-0.19
Sponsor	0.07	0.06	0.00	-0.01	0.16	-0.02	0.02	0.17	-0.05	-0.27	-0.35	-0.19	0.06	-0.05	-0.25	0.06	0.09	-0.15	1.00	-0.25	-0.25	-0.25	-0.25	-0.25
Installed	-0.09	0.11	0.01	0.00	0.14	-0.11	-0.20	0.02	0.20	0.10	0.17	0.10	0.23	0.22	0.21	0.18	-0.03	-0.19	-0.25	1.00	1.00	1.00	1.00	1.00
InstalledSquared	-0.09	0.11	0.01	0.00	0.14	-0.11	-0.20	0.02	0.20	0.10	0.17	0.10	0.23	0.22	0.21	0.18	-0.03	-0.19	-0.25	1.00	1.00	1.00	1.00	1.00

Table 3 Empirical results

Relevant topic	Hypothesis	Parameter	Estimate	Standard error	P-value
		<i>Intercept</i>	3582.20	1697.92	0.03
Scope	H1	<i>NumArea</i>	0.05	0.06	0.46
	H2	<i>NumType</i>	-0.07	0.05	0.14
Approach	H3	<i>Package</i>	0.21	0.11	0.05
	H4	<i>PctMod</i>	-0.57	0.27	0.04
	H5	<i>TeamBoth</i>	0.27	0.11	0.02
Technology	H6	<i>NumDBMS</i>	0.23	0.13	0.07
	H6	<i>NumDBMS2</i>	-0.20	0.08	0.01
	H6	<i>NumPL</i>	0.02	0.06	0.79
	H6	<i>NumOS</i>	0.04	0.12	0.73
	H7	<i>PctPL</i>	-0.59	0.30	0.05
	H7	<i>PctDBMS</i>	0.02	0.13	0.91
	H7	<i>PctOS</i>	0.28	0.12	0.02
Environmental control factors	Application	<i>Strategic</i>	0.20	0.09	0.02
		<i>Finance</i>	-0.06	0.06	0.30
	Organization	<i>Sponsor</i>	-0.16	0.06	0.01
		<i>Co1</i>	0.27	0.22	0.22
		<i>Co2</i>	0.03	0.11	0.78
		<i>Co3</i>	0.03	0.10	0.75
		<i>Co4</i>	0.07	0.24	0.77
	Time	<i>Installed</i>	-689.12	15.83	0.04
		<i>InstalledSquared</i>	33.16	327.86	0.04
		LogNormal Scale Parameter (σ)	0.1710	0.0240	
		<i>Log Likelihood</i>	-18.872		

The Maximum Likelihood results above pertain to the model depicted by Eqs. (6) and (7) in the text.

5. Empirical results

To assess fit and determine the relationships of the decision and environmental control variables with system lifetime we estimate three models. A naive model includes only an intercept term. A second model includes only the environmental control variables. The full model includes these control variables as well as the managerial decision variables outlined above that are related to our seven hypotheses. Both the environmental control variables outside of the IT manager's control and the decision variables under their control have strong impacts on fit. A likelihood ratio test of the environmental control variable model relative to the naive model is significant at the 1% level ($\chi^2(9) = 33.11$). The likelihood ratio test of the full model relative to the environmental control variable model also is significant at the 1% level ($\chi^2(12) = 27.38$). Table 3 reports the parameter estimates for the full model.

As is typical in empirical work, not all the explanatory variables are significantly related to the dependent variable—system lifetime measured in days.

To assess the model as a predictive tool, after estimating the model, the parameter estimates were used to calculate the probability that each system would continue to be used t years after installation. For the replaced systems, we then identified the age in years when the model predicted that the probability of survival would fall below 75%. This age was compared to the actual age of the system when it was replaced. For roughly half of these systems (15 of 33), this predicted age was either the actual age when replaced or one year longer (i.e., if a system was actually replaced after five years, the predicted probability that the system would still be in use fell below 75% in year 5 or year 6). For the systems still in use as of our 1994 cutoff date, the model predicts that 25—less than 17%—had less than a 75% probability of surviving past

the cutoff date. While there is room for improvement, these numbers are in line with the accuracy of software cost estimating models based on industry data [5].

The parameter estimates represent semi-elasticities. Therefore, to assess the relative impact of each variable, holding the impact of all other variables constant, we defined a “base case” level for all the variables in Table 3. In this base case *Package*, *PctMod*, *TeamBoth*, *Strategic*, *Sponsor*, *NumDBMS* (and, hence, *NumDBMS2*), *Finance* and the Company variables are all set to zero. *NumArea*, *NumType*, *NumPL* and *NumOS* are set to 1. The fit with standards variables *PctOS*, *PctDBMS* and *PctPL* are all set to their average percentage values (.58, .45 and .28, respectively). Finally, an installation date of May 24, 1987 is assumed, since this is the average installation date. The level of each significant variable (or variable combination) is then increased while holding the rest of the variables at their base case levels and the impact of this change on the probability that the system survives *t* years or

longer calculated. Figure 3 presents a graph of these probability curves. In particular, the binary variables are increased from 0 to 1; *PctOS* and *PctPL* are both increased by .22 to .80 and .50, respectively; *NumDBMS* and *NumDBMS2* are first increased from 0 to 1 (referred to as *NumDBMS+1*), and then from 1 to 2 and 4, respectively (referred to as *NumDBMS + 2*); *PctMod* is increased from 0 to 25% (referred to as *PctMod25%*, this also includes an increase in *Package* from 0 to 1). Finally, the installation date variables are both increased by one year (referred to as *Start + 1Yr*).

5.1. System decision factors: Scope

No support is found for either scope-related notion that systems supporting more functional areas [H1] or decision types [H2] are replaced more quickly. That is, *NumArea* and *NumType* are statistically insignificant. This lack of impact contradicts hypotheses H1 and H2, but is consistent with Swanson and Dans’ [28] finding that business functional complexity has no impact

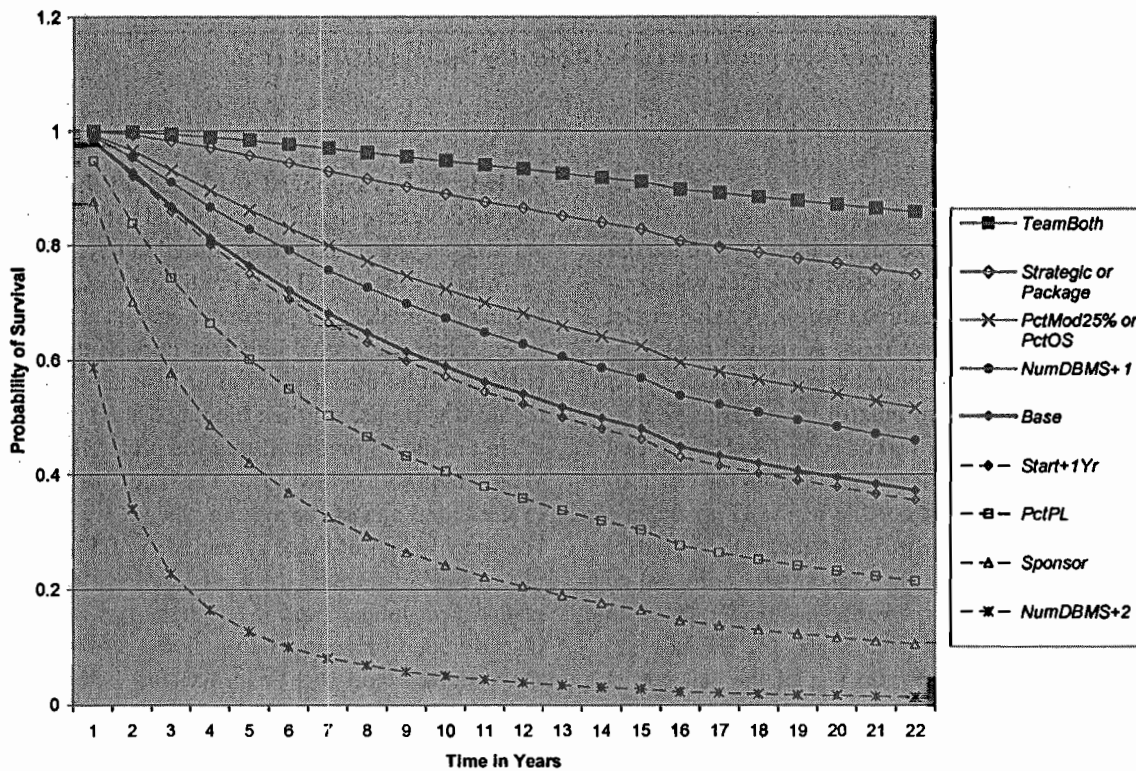


Fig. 3 Explanatory variables’ relationship with the probability of system survival

on remaining system life. While the requirements of a system that supports multiple areas or decision types may be more likely to change since it encompasses a greater breadth of functionalities, this complexity is not significantly related to the length of the system life.

5.2. System decision factors: Approach

The custom-package approach decision [H3] is significantly associated with a system's expected life span. In particular, as hypothesized, packaged software has a longer life (i.e., the Table 2 variable is *Package*, its parameter estimate equals 0.2107, its *p* value equals 5%). This jibes with Barry et al. [2] who find that packaged software is less volatile than custom developed software.

As expected, the more packaged software is modified [H4], the shorter is the expected life span (*PctMod*, -0.5668 , 4%). Note that for the *PctMod* variable to be greater than 0, the *Package* variable must equal 1. Therefore, *PctMod* is implicitly an interaction term and implies that a company uses its packaged software longer if it modifies its processes to fit the software system rather than modifying the software to fit its processes. More specifically, the estimated parameter value indicates that, all else equal, software that is modified more than 37% will lose all of the extended life associated with using packaged software (i.e., $.2107 - (.5668 * .37) = 0$). In our sample, 21% of the packaged applications were modified more than 37%.

The approach team hypothesis [H5] also is confirmed. A blended team is associated with a longer system life (*TeamBoth*, 0.2699, 2%). A blended team offers the advantages of internal business knowledge and external expertise. If the goal is to acquire software that is used longer, then the insource/outsource debate misses an obvious means to gain at least some of the advantages offered by each approach. This raises a future research question concerning the optimal team mix.

Since all three of these approach variables have relatively large parameter values, the approach decision is strongly related to the appropriate planning horizon. The probability of survival curves for *Package* and *TeamBoth* in Fig. 3 reflect this in that using either packaged software or a blended team has a stronger positive relationship with system life than does any other factor. Thus, a system that differs from the base case only in that a blended team or vanilla package was

used has a 95% probability of lasting ten years as compared to only a 59% probability for the base case. A switch to package software that is modified 25% (*PctMod* 25%) reduces this gain but still leaves this probability at 72%. This implies that when determining the software acquisition approach, a firm should not just compare internally-developed, custom software with packaged software. Rather it should leverage the expertise of internal and external constituencies. Our results imply that if longer lived systems are desired, minimizing customization and incorporating consultants to assist with implementation are warranted.

5.3. System decision factors: Technical complexity and fit with standards

Both the technical complexity of the system and its fit with technology standards also are related to system life. One aspect of technical complexity [H6] is significantly related to system life—the use of database management systems. Using a single database system instead of a file system is associated with longer life (*NumDBMS*, 0.2296, 7%). In addition, as expected, increasing technical complexity related to using more than one DBMS is associated with shorter system life (*NumDBMS2*, -0.1955 , 1%). The number of programming languages (*NumPL*) or operating systems (*NumOS*) used has no impact.

To put the magnitudes of the two DBMS parameters in perspective, two curves are presented in Fig. 3. A slight increase is seen in the survival probability when a single DBMS is used (*NumDBMS* + 1, where *NumDBMS* and *NumDBMS2* both equal 1). However, a quite large reduction is realized if greater than one DBMS is used. In particular, *NumDBMS* + 1, where *NumDBMS* = 2 and *NumDBMS* = 4, shows that the probability of surviving 10 years is brought down to less than 10% (roughly 1/10th of the base case probability).

Fit with standards [H7] also is strongly tied to system life, although the magnitude and direction depends on the technology. Operating system fit (*PctOS*, 0.2819, 2%) is significantly related to longer system life spans. On the other hand, adherence to the firm's standard programming language has a larger but negative association with system lifetime (*PctPL*, $-.5924$, 5%). Thus, a system that differs from the base case only in that the firm adheres to its operating system standard 80% (rather than 58%) of the time is 22% more likely

(i.e., $(72 - 59\%)/59\%$) to last 10 years. Alternatively, systems at a firm that adheres to their programming language standard 50% of the time are 31% less likely ($(59\% - 40\%)/59\%$) to last 10 years than are systems at firms that are average (28%) in their adherence to their programming language standard. This implies that to achieve a longer planning horizon, a standard operating infrastructure should be coupled with an advanced development environment. The more standard operating platform provides a stable environment for software maintenance which allows the firm to focus on business driven changes and not technical integration.

5.4. Environmental factors: Application, organization and time

At least one variable corresponding to each of the three environmental control factors is statistically significant. With respect to application characteristics, as expected, systems considered to be strategic are longer lived (*Strategic*, 0.2034, 2%). No evidence, however, is found that financial and accounting systems (*Finance*) are any longer or shorter lived, all else equal, than are systems supporting other functional areas. With respect to organizational factors, executive sponsorship, somewhat surprisingly, is associated with shorter lived systems (*Sponsor*, $-.1570$, 1%). This result implies that how the firm measures system success need not equate with how long the system is used. Similarly, as an anonymous referee pointed out, this result may be driven by the executives' ability to afford more ideas and get more projects of varying life spans implemented. It is also a bit surprising that there are no idiosyncratic company differences (none of the four company variables are significant). Hence, no firm exhibits a predisposition to replace systems more quickly or slowly than any other.

Figure 3 reflects that both of these significant environmental variables have a powerful relationship with system life. In particular, a change from the base case to a strategic system is associated with the same large increase in system survival probability as is the switch to packaged software (i.e., the *Strategic* survival curve coincides with the *Package* curve). On the other hand, a system sponsored by an executive has much smaller probability of lasting 10 years than does the base case (24% vs 59%).

The two variables measuring time—the installation date and its square—also are statistically significant (*Installed*, -689.123 4%; *InstalledSquared*, 33.165, 4%). However, although these parameter values are large, when taken together with the intercept term, their resulting impact is very modest (i.e., the combined effect of the three changes little over time). This modest impact of technology and business process change is depicted by the *Start + 1Yr* curve in Fig. 3 which lies only marginally below the base case curve. Clearly, taken together the installation date variables imply a slight trend towards longer lived systems. Since the pace of business change is increasing, technological advances focused on making systems more adaptable must be compensating for this change. It is for further study to determine whether this trend will continue.

6. Study limitations and future research

As with all empirical research, this study has a number of limitations and future extensions. Primary limitations are the age of the data and the number and breadth of observations. The end date for the data in this study is June, 1994. Any technological advances, organizational innovations or environmental changes since then, including the rise of the internet, new programming languages, the spread of object-oriented methodologies and the use of off-shoring, are, therefore, not modeled. It is a question for future research as to whether these changes alter the relative importance of the decision and environmental variables with respect to system life. We believe that the primary managerial drivers of system life continue to be how the system is acquired and its complexity.

The data set consists of 181 systems from five companies. Hence, our results may not generalize to systems at other companies. However, given the breadth in our data concerning company size, industries represented and types of systems implemented, we expect that our results will carry forward. That said, a larger scale analysis or multiple small scale tests over a range of companies would provide greater evidence of generalizability. If conducted, these studies might wish to include additional variables, such as system size. Size was omitted from this study because the participating firms did not have good measures for it, and because our focus was on estimating system life at the feasibility analysis stage where these measures are not

Table 4 Summary of empirical results

Decision factor	Hypothesis	Expected impact on appropriate planning horizon	Empirical evidence
Scope	H1: Number of functional areas	Decrease	No Support
	H2: Number of decision types	Decrease	No Support
Approach	H3: Packaged software	Increase	Support
	H4: Packaged software: Level of modification	Decrease	Support
	H5: Blended team	Increase	Support
Technology	H6: Technological complexity	Decrease	Support
	H7: Fit to standards	Increase	Support and contradiction

well known. Size is believed, however, to have a significant impact on system life and, therefore, could be included in future work.

A potentially valuable future research topic is the integration of the various literatures that are related to system management. In addition to this work looking at system life, system planning, system volatility and system quality all have been analyzed, but an integrated model and approach continues to be elusive.

7. Summary

This paper presents the first empirical analysis focused on estimating the life of software under consideration at the feasibility analysis stage. This estimate defines the appropriate planning horizon for the software acquisition decision. An empirical analysis of how long actual systems are used shows that management decisions made at the business case stage concerning system approach and technology, but not scope, affect a system’s life span. For example, based on our results, an internally developed, custom, non-strategic system sponsored by someone other than an executive that has average values for the other significant variables will have a 75% chance of being used for at least 5 years. However, a system that differs only in that it is acquired as packaged software has a 75% chance of lasting 22 years or more.

This empirical work is meant to be illustrative. As with software cost estimation models like COCOMO and QSM’s SLIM, a particular firm is likely to estimate system life more accurately utilizing measures similar to those outlined in this paper gleaned from their own software systems and, perhaps, those of like firms. The model discussed in the paper, however, does provide a

starting point for estimating a proposed system’s life and for either adjusting these estimates based on key management decisions and environmental characteristics or basing management decisions on the desired system life span.

Different systems can have very different life spans. To properly estimate the costs and benefits of a system investment, managers need to accurately estimate system life and, thus, the time frame over which the firm expects to receive benefits and pay operation and maintenance costs. For a particular system investment, going with packaged software or a blended implementation team is associated with a longer system life and, therefore, a longer time span in which to reap the benefits generated by the system. Alternatively, being at the leading edge of technology adoption may provide the firm with new capabilities and allow them to be a technology leader, but if it requires deviating from the firm’s architectural standard for its operating system, a shorter lived system is expected. This would require that the system payback its investment cost more quickly than if the firm had adhered to its operating system standard. It follows that the appropriate estimate of system life can significantly impact not just the Go/No-Go decision but also the IT department’s budget and planning cycles. In such, accurately forecasting system life spans will enable the firm to better estimate the resources required to support the system as well as assess when it will need to start planning its replacement or retirement.

Our analysis extends previous studies that focus on identifying factors affecting system success, reliability, maintenance and remaining system life by evaluating the underlying issues affecting system longevity. It relates management decisions and environmental control variables to how long a system is actually used. This provides a good estimate of what a similar proposed

system's life will be. Table 4 summarizes the empirical findings concerning our seven hypotheses detailing how scope, approach and technology decisions affect a system's life span. Statistically significant results consistent with the approach and technology hypotheses are realized. The scope hypotheses have no support. In addition, interesting findings on the environmental control characteristics provide additional insight into defining an appropriate system planning horizon. In short, holding the levels of the other management decisions and environmental characteristics constant, our key findings are:

- Application scope (defined as breadth of functionality supported) is not related to system life spans.
- The approach taken to acquire a system matters. A blended team combining both internal and external resources is associated with increased system life, as is using packaged software. The useful life of packaged software, however, is reduced the more the software is modified.
- Adhering to technology infrastructure standards is associated with system life, but the direction depends on the technology. In particular, choosing an operating system and sticking with it is associated with increased system life. Alternatively, choosing and sticking with a programming language is associated with reduced system life.
- Environmental variables also strongly influence system life. In particular, while how long a system is used does not vary idiosyncratically across firms even though the firms varied significantly in size and industry, systems sponsored by executives are shorter lived and strategic systems longer lived. Also, somewhat surprisingly, over time as business processes and technology have advanced rapidly, a strong trend toward longer or shorter system life has not evolved. In fact, a slight lengthening in life spans is observed.

Our model offers an easily implemented forecasting procedure for system longevity. Our empirical results reveal how a firm's decisions influence a software system's life. For a particular software solution decision, this allows an improved forecast of its relevant time horizon, and, in such, engenders better cost-benefit comparisons among rival software solutions. Knowing how long a system is expected to last also will help firms determine the effort they want to expend ensuring the system is maintainable. From a strictly descriptive point of view, the empirical analysis of-

fers insights into what types of system characteristics are associated with longer lived systems and which characteristics are associated with shorter lives. For researchers, the framework broadens the scope for examining software success to include its expected life span and provides a stepping-stone to more focused research on particular aspects of the software replacement problem.

References

1. R.D. Banker, S.M. Datar, C.F. Kemerer and D. Zweig, Software complexity and maintenance costs, *Communications of the ACM* 36 (1993) 81–94.
2. E. Barry, C. Kemerer and S. Slaughter, How development decisions affect product volatility: a longitudinal study of software change histories. GSIA Working Paper #2003-E66, (2003).
3. A. Barua and T. Mukhopadhyay, A cost analysis of the software dilemma: to maintain or to replace, *Proceedings for the Twenty-Second Hawaii International Conference on Systems Sciences* (1989).
4. V. Basu, E. Hartono, A.L. Lederer and V. Sethi, The impact of organizational commitment, senior management involvement, and team involvement on strategic systems planning, *Information & Management* 39 (2002) 513–524.
5. B. Boehm, *Software Engineering Economics* (Prentice Hall, New Jersey, 1981).
6. R. Bruner, Y2K cost disclosures eyed, *Electronic News* 10/4/1999.
7. T. Chan, S.L. Chung and T.H. Ho, Timing of software replacement, *Proceedings of the 15th International Conference on Information Systems*. Vancouver, Canada (1994) pp. 291–307.
8. L. Erlikh, Leveraging legacy system dollars for e-business, *IEEE IT Pro* (May/June 2000) 17–23.
9. D. Gode, A. Barua and T. Mukopadhyay, On the economics of the software replacement problem, *Proceedings of the Eleventh International Conference on Information Systems*. Copenhagen (1990) pp. 159–170.
10. W.H. Greene, *Econometric Analysis* (Prentice Hall, New Jersey, 2000).
11. M. Hanna, Maintenance burden begging for remedy, *Software Magazine* 13(6) (1993) 53–63.
12. J. Heales, Factors affecting information system volatility, *Proceedings of the Twenty First International Conference on Information Systems* (2000) pp. 70–83.
13. S. Kaplan, Pull the plug on your legacy applications, *CIO* (March 15 2002) 56–65.
14. R.E. Kass, and Adrian E Raftery, Bayes factors, *Journal of the American Statistical Association* 90(430) (1995) 773–795.
15. Kmart Press Release, Kmart to restructure supply chain operations. <http://www.kmartcorp.com/corp/story/pressrelease/news/pr010906.stm>. 9/6/2001.
16. M.M. Lehman, J.F. Ramil, P. D. Wernick, D. E. Perry and W. M. Turski, Metrics and laws of software evolution—the

- nineties view. IEEE-CS Albuquerque, NM. 5–7, November 1997, pp. 20–32. Comp. Sci. Order No. PRO8093.
17. J.E. Lawless, *Statistical Models and Methods for Lifetime Data* (John Wiley and Sons, New York, 1982).
 18. B.C. McNurlin, Replacing old applications, *EDP Analyzer* 21(3) (1983) 1–12.
 19. P. Nelson, W. Richmond and A. Seidmann, Two dimensions of software acquisition, *Communications of the ACM* 39(7) (1996) 29–35.
 20. G. Premkumar and W.R. King, An empirical assessment of information systems planning and the role of information systems in organizations, *Journal of Management Information Systems* 9(2) (1992) 99–126.
 21. G. Premkumar and W.R. King, Organizational characteristics and information systems planning: An empirical study, *Information Systems Research* 5(2) (1994) 75–109.
 22. L.H. Putnam, and W. Meyers, *Measures for Excellence Reliable Software On Time, Within Budget* (Prentice Hall, New Jersey, 1992).
 23. A.E. Raftery, Approximate Bayes factors and accounting for model uncertainty in generalized linear models, *Biometrika* 83(2) (1996) 251–266.
 24. B. Rosser, How to economically justify an IT investment. Research Note TU-14-1816, (2001), Gartner, Inc. <http://www.gartner.com>.
 25. B. Schaller, The origin, nature, and implications of Moore's law: The benchmark of progress in the semiconductor industry. Working Paper, School of Public Policy, George Mason University, (1996), <http://mason.gmu.edu/~rschalle/moorelaw.html>.
 26. Standish Group, The, The Standish Group Report: CHAOS. Technical Report, (1994).
 27. E.B. Swanson, Framing the system replacement decision. Information Systems Working Paper 01-98 (1998) UCLA, Los Angeles, California. http://www.anderson.ucla.edu/acad_unig/info_sys/.
 28. E.B. Swanson and E. Dans, System life expectancy and the maintenance effort: Exploring their equilibration, *MIS Quarterly* 24(2) (2000) 277–297.
 29. A. Taudes, Software growth options, *JMIS* 15(1) (1998) 165–185.
 30. E. Wang, T. Barron and A. Seidmann, Contracting structures for custom software development: The impacts of informational rents and uncertainty on internal development and outsourcing, *Management Science* 43(12) (1997) 1726–1744.
 31. P. Weill and M. Vitale, Assessing the health of an information systems applications portfolio: An example from process manufacturing, *MIS Quarterly* 23(4) (1999) 601–624.