A Method for Assessing Legacy Systems for Evolution

Jane Ransom, Ian Sommerville, and Ian Warren
Computing Dept., Lancaster University, LANCASTER LA1 4YR, UK
Email: bjr, is, iw@comp.lancs.ac.uk

Abstract

Legacy systems are usually critical to the business in which they operate, but the costs of running them are often not justifiable. Determining whether such systems are worth keeping requires an overall assessment of the system. We present an assessment method that examines a legacy system from its technical, business and organisational perspectives. The method guides users through assessment of these perspectives by selecting assessment characteristics and assigning values to them. The method provides further guidance on interpreting the results obtained from assessment. Our assessment method can be tailored to the needs of particular evolution projects and organisations. It is not prescriptive of particular tools and techniques, and can be instanciated to offer a cost/risk trade-off. Quick estimates can be derived from performing the method at a high level. The risk of producing an inaccurate assessment can be reduced by further iterations of the method, performed at more detailed levels.

1. Introduction

A legacy system is a system which was developed sometime in the past and which is critical to the business in which the system operates. Typically, legacy systems were developed before the widespread use of modern software engineering methods and have been maintained to accommodate changing requirements. These two factors result in systems which are often difficult to understand and expensive to maintain. Many legacy systems thus present a dilemma - such systems are indeed critical to the business process, but maintaining them incurs unjustifiable expense [1].

A legacy system may evolve in a number of ways, depending on factors such as its technical condition, its business value, and the characteristics of organisations involved in maintaining and operating the system. Continued maintenance, some form of reengineering, or replacement are general evolution strategies, of which one or a combination may be an appropriate way of evolving a legacy system. Reaching a decision about how best to evolve a legacy system cannot be made spontaneously; rather it should be based on an *assessment* of all relevant system attributes.

System assessment is used to gain an understanding of a legacy system, which is fundamental to any system evolution exercise [2, 3]. System assessment should be an initial activity for evolution projects. Our literature survey has exposed a lack of practical advice for assessing systems, and for evolution project management in general. Sneed's work [2] is a noticeable exception.

The assessment method presented in this paper essentially involves measuring the system's technical quality, its business value, and its associated organisational characteristics. The method provides guidance on deriving and interpreting these measurements. The result can then be used as the basis for selecting a particular evolution strategy.

Our assessment method is a product of the Esprit project RENAISSANCE. The main objective of RENAISSANCE is to develop a systematic method for system evolution and reengineering which is geared to the requirements of commercial systems. The RENAISSANCE method [4] offers guidance for system modelling, techniques for migrating legacy systems to distributed client/server technology, and advice for evolution planning. RENAISSANCE is built on two key ideas:

1. *Reengineering must be company and project specific*. The method is designed so that it can be instanciated according to particular company and project requirements.
2. *Both the reengineered process and the reengineered system must be continuously refined*. The RENAISSANCE method can evolve to accommodate new technology, techniques and tools. The principal product of following the method is a system which has been transformed from a legacy state to an evolutionary system. Evolutionary systems [1] accommodate change by incorporating evolution as a core activity of the software life cycle, and not as an extension to it.

The remainder of this paper is focused on the assessment method, which is realised as an evolution

planning activity within the more comprehensive RENAISSANCE method. Section 2 introduces the assessment method, which is followed by sections 3 to 6 which respectively describe particular method activities: business value assessment, external environment assessment, application assessment and interpretation of results. Section 7 concludes by summarising the contribution made by this work.

2. Assessment method

The system assessment method is composed of a number of activities, illustrated in Figure 1. The method is designed to be iterative and may be performed at a number of levels of detail to offer a cost/risk trade-off. In the interests of applicability to a range of organisations, the method is not prescriptive of techniques and tools used to perform method activities. Rather, we offer practical guidance through the assessment process.

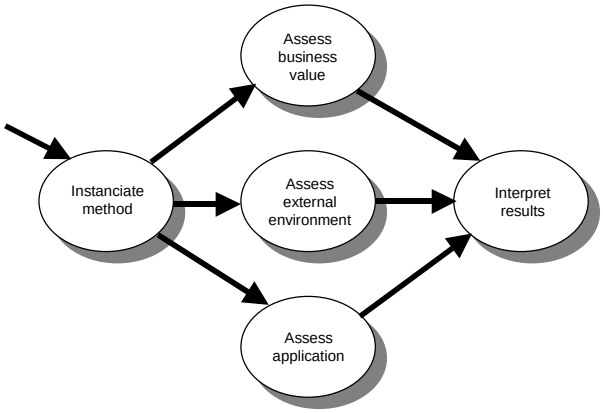

Figure 1 Assessment method activities

2.1 Aims

The product of the assessment method is to gain a sufficient depth of understanding of the legacy system. Understanding the system from technical, business, and organisational perspectives provides a firm foundation from which a decision can be made for an appropriate evolution strategy. The following questions are typical of what can be answered on completion of the assessment exercise.

- *Is the system critical to the organisation in which the system operates?* Assessment might reveal that a system is not essential to the continued operation of the business. If this is the case, it is not necessary to consider the system further for evolution.
- *What are the organisation's business goals?* From a business perspective, assessors must understand the business goals of the organisation that uses the legacy system. Business goals generate evolution requirements.
- *What are the evolution requirements?* Evolution requirements are derived from business goals and assessment activities. Requirements need to be anticipated to determine whether the existing system can satisfy the requirements.
- *What is the anticipated lifetime of the system?* A system's lifetime is dictated by factors such as serviceability of software and hardware. When hardware or support software becomes obsolete, the useful life of the system is limited.
- *What is the required life of the system?* If it becomes apparent that the system is not required beyond a short period, it is pointless to incur the expense of transforming it to an evolutionary system. Conversely, if it is expected that the system will support key elements of the business process for some considerable time, then it may be wise to take some action.
- *What is the technical state of the system?* Where application software, for example, is in a poor technical state, it will be difficult to understand and expensive to maintain. For a system with a long required life, effort to make the system more evolveable would be sensible.
- *Is the organisation that operates the system amenable to change?* The organisation's attitude towards change is a critical success factor for evolution projects.
- *Does the organisation responsible for evolving the system have sufficient resources?* Factors such as the organisation's technical maturity, skills base and quality of employees, and level of tool support have an impact on evolution projects.

2.2 Instanciation

An instanciation activity is necessary to tailor the assessment method to the needs of the organisation and project. Instanciation involves defining two parameters:

1. *Assessment technique*. Either expert opinion or quantitative metrics may be used. This decision is based on availability of individuals to play expert roles, and maturity of the organisation performing the assessment.
2. *Assessment level*. The level of detail at which to carry out the assessment. Many of the properties of legacy systems can be examined at varying degrees of granularity. This decision is principally based on the depth of understanding required.

To illustrate method instanciation, it is often useful for large organisations to prepare a portfolio of applications that are candidates for evolution [2]. To do this, each

application should be assessed quickly - the risk of inadequate understanding is not significant at this stage. A method instanciation based on high level assessment using expert opinion would be sensible in this case. Further method iterations can subsequently be made at a lower level of detail for each of the evolution candidates to gain a better understanding.

Instanciating the method also involves identifying which activities are relevant to the current assessment exercise. The assessment activities shown in Figure 1 may be performed in parallel, and in some cases, particular activities can be omitted. This flexibility is necessary to provide a user-driven method. For example, in the case where a decision has been made to migrate to client/server platforms, there is no need to assess the legacy system's hardware (part of Assess external environment), as it is a foregone conclusion that the legacy hardware will be replaced.

2.3 Experts, metrics and information sources

Expert opinion is a technique that can be used at any level of detail. The use of expert opinion is of course constrained by the availability of individuals to fill expert roles. System assessment requires experts not found in forward engineering projects:

- *Application business expert*. One or more individuals are required to fill this role. The role requires a deep understanding of business processes, and an ability to determine whether the process complies with current and anticipated business requirements. Senior members of the organisation in which the legacy system operates are good candidates for this role.
- *Legacy functional expert*. Individuals are needed who understand how the system supports its users. Candidates for this role include system operators and middle managers who use the system.
- *Legacy implementation expert*. Personnel who have experience with developing and maintaining the legacy system, and who have acquired a deep understanding of the system are needed for this role.

There is usually no difficulty in finding individuals to fill the first two expert categories. Legacy implementation experts are however more transient, and it is not always possible to find such individuals. In this case, alternate sources of information must be relied on such as source code, documentation, the operational system itself, and historical information, such as change requests.

Metrics offer the most objective form of assessment, but are constrained to detailed assessment where components are identified with attributes which can be quantifiably measured. This is generally not true of high level assessment. For example, attempting to calculate a quantifiable value for the complexity of a system's application software without assessing in detail, the components that comprise the system is meaningless. Where metrics are used, caution must be exercised. In particular, practicing organisations must be sufficiently mature to run a metrics program, and must be able to interpret the meaning of metrics within their local environments. Naïve use of metrics is likely to result in a gravely misunderstood system.

3. Business value assessment

The goal of business value assessment is to determine the importance of the system to the organisation. In many cases, changes to the underlying business process mean that old systems are now of only peripheral value, and there is little effort in expending time and effort in modifying such systems. In other cases, the systems are business-critical and must be maintained in operation. Business value assessment can be carried out at two levels:

1. *High level*. A high level assessment using expert opinion is recommended in all cases to eliminate systems which are clearly of little value. A high level assessment will also expose candidates for more detailed analysis.
2. *Detailed level*. Business value analysis involves systematic ranking of the application against business criteria. This approach is more time consuming and Sneed [2] suggests that it is best performed by business analysts.

High level assessment involves:

- *Identifying appropriate experts for consultation*. A number of individuals should be consulted, including end-users and managers responsible for the business process where the system is used.
- *Constructing assessment questionnaires*. Questionnaires should identify where the system is used in the process, the relationships between the system and other systems, the costs and changes required if the system was no longer available, and defects and problems with the existing system. Precise questions are dependent on the system being assessed.
- *Performing the assessment*. In addition to questioning experts, effort should be allocated to watching how the system is used. This often reveals details about its value, which do not emerge from questionnaires, for example business process support deficiencies.

4. External environment assessment

The external technical environment of a system is the union of hardware, support software, and critically, the infrastructure of the organisation.

4.1 Hardware

The hardware of a system includes many components that require regular maintenance. The hardware maybe centralised at one site, or it may be dispersed over many sites and linked by networking technology. The quality of hardware is determined by the total maintenance costs and whether the hardware is still supported. Typical hardware components found in legacy systems include mainframe and mini computers, disk drives, tape units, terminals, printers and networking hardware.

Vendor/supplier rating
Maintenance costs
Failure rate
Age
Ability to perform function
Performance

Table 1 Hardware characteristics

Similarly to business value assessment, hardware can be assessed at either a high or detailed level. High level assessment provides a quality estimate for the hardware of a legacy system as a whole, and must be performed by expert opinion. Detailed assessment involves identifying particular hardware components of the system. In both cases, a set of characteristics must be identified and used as the basis of assessment. Selection of characteristics is dependent on the system being assessed, but some general characteristics are presented in Table 1.

Assessment progresses by assigning each component (detailed level) or the system (high level) a value for each of N characteristics. We suggest a simple technique based on assigning each characteristic a value in the range 1 to 4, where a low value indicates concern. The rating is simply the sum of the scores. A score less than N * 4 / 2, where N * 4 is the maximum score possible, generally indicates that some action needs to be taken.

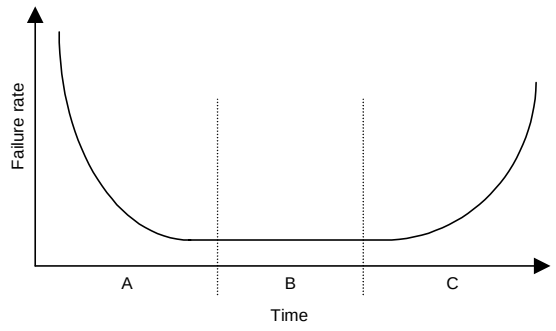

Figure 2 A Bathtub curve

Using our scheme, consider vendor/supplier rating for example. Where the vendor is no longer in existence, the characteristic would be assigned 1. An improved situation, warranting a value of 2 is where the vendor no longer exists, but a third party supports the hardware component. A vendor who is still in existence, but whose future appears uncertain attracts a value of 3, and a top value of 4 represents the best case where the vendor exists and its future seems assured. Other characteristics should be treated similarly.

The lifetime of hardware components typically follow a Bathtub curve, illustrated in Figure 2. Failures in the first period (A) are relatively high dropping down to a low rate for a relatively long period (B). As the component begins to wear, failures increase (C) to the point where the component must be replaced. A low score for a hardware component generally means that the component is in area C of the curve.

An overall rating for the hardware can be obtained by a weighted average of all assessed components. To conclude hardware assessment, the impact of hardware on software should be assessed. This can measured on a scale of 0 to 1, where 0 indicates that a strong dependency exists - changing hardware will demand software changes. This is particularly true of obsolete hardware. A value of 1 represents the case where the software is not dependent on the hardware.

4.2 Support software

The support software environment of a system comprises many components, which require regular maintenance in the form of upgrades. Typically, there are many interdependencies between support software components. Moreover, support software is often dependent on particular hardware. Application software, in turn, is often dependent on system software. Such dependencies must be taken into account during the assessment exercise. Examples of support software components include operating systems, databases, transaction processing monitors, compilers, 4GLs and networking software.

License costs
Frequency of fixes/patches
Quality of support personnel

Table 2 Additional characteristics

The assessment process for support software is the same as for hardware. The hardware characteristics from Table 1 are all applicable to support software. In addition to these, and any other specific characteristics introduced by method practitioners, we suggest the characteristics in Table 2.

If the support software is generally satisfactory, it may be possible, in the interests of minimal disruption, to reengineer or replace only components of poor quality. However, it must be stressed that individuals performing the assessment exercise should understand the relationships between software components that might be replaced and their interdependencies with other system software components and hardware.

Finally, a factor to be used to adjust the quality of legacy application software developed using particular support software should be determined by deciding the importance of it to the legacy system. A value in the range 0 to 1 should be assigned, where 0 indicates that the support software is obsolete, and where 1 shows the support software to be used.

4.3 Organisational infrastructure

The organisational infrastructure includes the organisation responsible for developing and maintaining the system, and the organisation that operates the system. In some cases these organisations will be the same. The organisational infrastructure is likely to have an impact on overall system quality. Organisational factors can be difficult to assess, but may be critical to the success or otherwise of an evolution project. We suggest the following factors should be considered:

- *Type of organisation and system users*. The organisation that operates the system may have staff dedicated to software development, or alternately, all development and application management may be outsourced. System users may perform repetitive clerical jobs, or may be involved with more skilled work.
- *Technical maturity of the organisation*. A measure for the development organisation is influenced by factors such as whether modern software engineering methods are used, whether the organisation conforms to a certified standard, and whether process improvement programs are operative.
- *Training procedures in the organisation*. Evolution is likely to be more successful if the organisation is committed to staff training. This is true for both organisations.
- *Skill levels of system support*. If the skill and experience level of the system support staff in the development organisation is poor, making major changes to the system may be unwise.
- *Organisational attitude to change*. Some organisations (often smaller organisations) are very responsive to change. Other organisations (often public bureaucracies or large companies) may have a more negative view of change, and by default, resist changes which are imposed on them.

5. Application assessment

This activity is concerned with the application software of a legacy system, and offers scope for the level of depth at which the assessment is performed. There are two basic levels:

1. *System level*. The system is viewed as an atomic entity, and assessment does not consider any of its component parts. Similarly to high level hardware and support software assessment, system quality must be determined using expert opinion.
2. *Component level*. At this level, assessors focus on some or all of the subsystems. It may be that only a few subsystems are selected for further treatment in a large system to provide a representative assessment. Large systems may have hundreds of subsystems, and it is not practical to study each one. Detailed level assessment can further be broken down into two levels of detail, assessed using expert opinion or formal metrics:
    1. *High level*. Subsystem(s) are assessed without considering any of their component parts.
    2. *Low level*. Subsystem(s) are assessed by examining their internal components, which may be subroutines, include files, screen maps, report definitions and data structures. Assessment of a subsystem is based on a weighted average of an assessment of its components.

Complexity
Data
Documentation
External dependencies
Legality
Maintenance record
Size
Security
Test bed

Table 3 Software characteristics

Generally applicable characteristics are presented in Table 3. Characteristic values can be approximated at a system level using expert opinion. Where more effort is deployed in assessing the application software, through more detailed analysis, and perhaps with the use of formal metrics, overall values for system characteristics can be computed as a weighted average of subsystem values. The assessment process for application software is similar to the process described earlier for hardware and support software.

6. Interpretation of results

The assessment activities of the method produce a collection of characteristic-value pairs for hardware, support software and application software. These properties of the legacy system collectively provide a technical perspective of the system. A weighted average assessment value for technical quality can be derived from the technical property values.

The resulting overall measure for technical quality can be compared with the business value (derived from the business value assessment activity), to provide an initial indication of the type of evolution required for the system. Determining a measure of business value falls outside the scope of software engineering, and requires the expertise of a business analyst.

Nolan Norton & Co. [5] provides a practical approach to comparing technical quality with business value, based on plotting the results on a quadrant map (Figure 3). The results for technical quality and business value should be converted to a value in the range 0 to 100. By example, consider a technical quality value of 70 out of a maximum value of 300. The coefficient is obtained by dividing 70 by 300, and multiplying the result by 100.

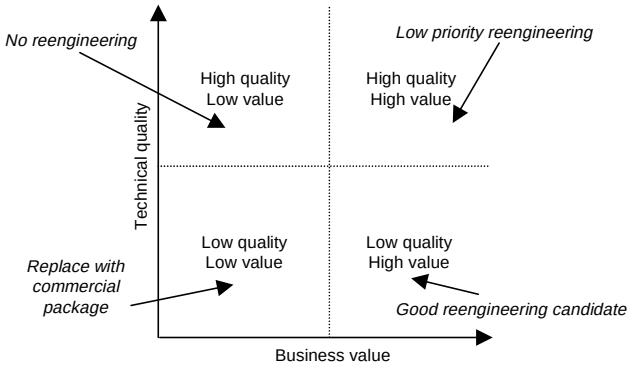

Figure 3 Technical quality and business value

The factors presented in Section 3 can be used to produce a business value coefficient. Alternately, business value may be calculated using the system's cash value. In this case, the coefficient is the annual revenue the system earns, divided by the total annual revenue from all systems in the same business area, and multiplying by 100.

The above approach yields an approximate result, and it is important that further analysis and refinement are carried out.

- *The organisational infrastructure should be analysed*. The organisational infrastructure is a critical success factor for evolution projects, but is not recognised in Figure 3. Assessment of the development organisation for example, may show that the existing skill base is directed towards COBOL application development on mainframe hardware. A reengineering strategy, which involves migrating to 4GL-based development on client/server platforms, requires a substantial technology shift, which would prove difficult for development personnel to make. This is risk that could be avoided by less ambitious reengineering (restructuring existing programs for example), or mitigated by training staff to use the modern technology.
- *Assessments should be further analysed*. Consider a system with a high business value, but with a low technical quality. Further analysis shows that it is the support software that is primarily responsible for the low technical rating. The combination of high business value, and low technical quality would render the system a good candidate for reengineering. A new release of the support software is expected within 12 months however, with significantly enhanced capabilities. This suggests that the reengineering priority should be downgraded, and a reassessment made when more information about the revised support software is available.
- *Critical decision making should be based on detailed analysis*. Consider the case where a system has a high technical quality, but the risk of mainframe computer support being withdrawn in the near future is significant. Where hardware assessment is performed at a high level, and is found to be of general good quality, the risk and its possible consequences are dwarfed. It is only when assessment is made at a detailed level, decomposing hardware into its component parts, that issues such as these are highlighted.
- *Additional external issues must be considered*. The Year 2000 problem is a good example of a critical success factor for organisations' survivability. Consider the case where application assessment shows the data managed by an application to be of high quality (relevant to the business process and well organised), leading to a naïve conclusion that no data reengineering is necessary. However, the Year 2000 date change issue is a risk with a 100% probability of occurring. The consequences of not preparing data and structures by the year 2000 may be fatal for many organisations.

Analysis beyond performing the assessment activities is thus clearly necessary. A thorough understanding of the elements comprising a legacy system, in addition to external factors is required to drive analysis and subsequently to select systems for reengineering.

7. Conclusions

In this paper, we have presented an overview of a method for assessing legacy systems. The result of the method is a level of understanding, which is related to the effort employed to perform the assessment. The understanding gained is used to determine whether a legacy system is a candidate for evolution. The method is not prescriptive of techniques or tools to perform particular method activities. Moreover, users of the method select which activities are relevant to their circumstances.

The method has been designed to provide for comprehensive assessment of legacy systems. We have demonstrated that a legacy system has technical, business and organisational properties. Each property represents a critical success factor for evolution projects, and so activities are provided to assess all three properties. It must be emphasised that following assessment, a thorough analysis is required in order to effectively interpret the results of assessment.

The method has been developed from collaboration between academia and industry. Consequently, the method is geared to providing practical advice and guidance to industrial practitioners. Practical orientation is complementary to other work, such as [6, 7, 8] which is concerned with more abstract frameworks and process models. The assessment method is currently being evaluated as part of the RENAISSANCE method by industrial organisations involved in the project. We aim to incorporate their feedback in subsequent refinement of the method.

8. References

[1] *The RENAISSANCE Framework*, RENAISSANCE project deliverable, 1997.

[2] Sneed, H. M., *Planning the Reengineering of Legacy Systems*, IEEE Software, January 1995, pp. 24-34.

[3] Battaglia, M., Savoia, G., and Favaro, J., *RENAISSANCE, A Method to Migrate from Legacy to Immortal Software Systems*, 2nd Euromicro Conference on Software Maintenance and Reengineering, 1998.

[4] *The RENAISSANCE Method*, RENAISSANCE project deliverable, 1997.

[5] Sneed, H. M., *Economics of Software Reengineering*, Software Maintenance: Research and Practice, Volume 3, 1991, pp. 163-182.

[6] *Perspectives on Legacy System Reengineering*, Software Engineering Institute, Carnegie Mellon University, 1995.

[7] *Assessing the Evolveability of a Legacy System*, Software Engineering Institute, Carnegie Mellon University, 1996.

[8] *Software Reengineering Assessment Handbook Version 2.0 Volumes I and II*, USA Department of Defense, 1995.

9. Acknowledgments

This work is partially funded by the European commission's ESPRIT initiative (project 22010, RENAISSANCE). Copies of project deliverables, which describe this work in more detail and other aspects of the RENAISSANCE project may be downloaded from http://www.comp.lancs.ac.uk/projects/renaissance/.