From “Make or Buy” to “Make and Buy”: Tailoring Information Systems Through Integration Engineering

Karl Kurbel and Claus Rautenstrauch
University of Muenster

Bernhard Opitz and Rolf Scheuch
TRIGON Gesellschaft fuer Organisationsberatung und Projektabwicklung mbH

Integrating information systems has become a major issue in IS development and information management. As organizational change is a permanent process, new information systems have to be developed continually to meet new requirements, but old ones also continue to be in use. This means that new components have to be integrated into existing environments. To make the process of permanent extension and modification manageable, a methodology is needed that supports easy integration of modules into existing environments. We call this methodology Integration Engineering. The paper outlines principles and methods of Integration Engineering with respect to developing client-server based business information systems. The methodology was successfully applied in a large project. The goal of the project was to develop an enterprise-wide information system for a plant-building company. The case study is described in the second part of paper.

IS Development in Changing Organizations

For many companies today, integrating their information systems (IS) is of strategic importance. Integration has become the major issue in IS development and information management. In a recent survey, both IS managers and academic staff from American universities rated integration on top of their respective priority scales (Trauth, et al., 1993). One of the reasons for this is the growing number of information systems already in use. They all deal with particular aspects of business problems, but many of them are stand-alone systems or, at best, loosely interconnected—with all the problems due to redundancies and inconsistencies.

Organizational changes such as re-orientation towards business processes require development of new information systems and restructuring old ones. For example, existing IS functions have to be examined and reengineered if they are to be incorporated into new process-oriented information systems (Kurbel, 1994). In today’s companies, existing IS environments have to be taken into account whenever developing new systems is under consideration. IS development hardly ever starts “from scratch”. New software systems must comply with existing functionality and data structures.

Integrated information systems (IIS) have a number of advantages over separate ones. Some of them are (Scheer, 1991):

- Data redundancies and functional redundancies are reduced or avoided.
- Transition times between steps of a business process become significantly shorter.
- One person can perform several steps of a business process, instead of different people performing different steps. Additional effort to get acquainted with each special case is avoided.
- The cost of coordinating people is reduced when less people are involved in a process and when they get
support from an integrated information system. 

At first glance, the easiest way to realize an IIS appears to be buying an integrated standard package of business software. Such packages are available today. In Germany, for example, SAP’s R/2 and R/3 systems have become something like a “standard” for large and medium-size companies (Ricciuti and Semich, 1993). There are installations in the US, too. R/2 and R/3 are enterprise-wide information systems, covering most business areas. 

However, the vote for “buy” also has some major drawbacks: Universal standard software systems are big and not so easy to handle. They contain many functions and features that may be irrelevant to the particular company or that need to be adjusted. Hundreds of parameter settings may have to be fixed (and maintained) in order to adapt oversize standard software to enterprise-specific requirements (Wedel, 1990). For the user, it is often impossible to judge effects of parameter constellations in advance. Some settings may even be counterproductive. For these reasons, installing and maintaining parametrized systems is time-consuming and expensive. But even if parameters could be adjusted in an optimum way, there are always some unfilled requirements left—regarding the particular industry branch or idiosyncrasies of the individual enterprise. The missing parts are not necessarily large but often of utmost importance to the users. They have to be developed by individual programming, unless branch-specific software components are available. 

The extreme opposite of standard packages is individual software, but the number of companies developing their IIS completely by individual programming is decreasing. Although the decision for “make” has the advantage that all, and only individual requirements will be met without overhead, experience from 30 years of software engineering has shown that developing complex software systems is expensive, error-prone, and full of economical risks. 

Taking the pros and cons of individual and standard software into account, the appropriate strategy to come to an integrated information system is in many cases the middle way between “make” and “buy”: For business areas well covered by standard software, the respective parts of a package can be used. Only if requirements are not met well, individual IS development is considered. A combination of standard and individual modules suits most companies best. 

However, integrating standard software and individual programs can be very difficult. A prerequisite for integration is well-defined interfaces; i.e. standard systems have to be open systems. The way the term “open” is mostly used today, it means either that 

1. a software system has a client-server architecture, with the server holding the database and the clients most of IS functionality, or 
2. a programming interface is available which may be used by software developers to extend systems by individual functions. 

Being open in either sense is important, but it is not sufficient for integration. Even though a client-server architecture supports data integration, and software interfaces are helpful for customization, more is needed. We call the desired property software compatibility, and we use the term in a similar way as in object-oriented programming (Meyer, 1988). 

Compatibility with regard to a piece of software means that both its functionality and its data can be combined with other software, i.e. the respective software components can be integrated into a new logical unit. Compatibility is an additional characteristic of software quality, such as maintainability, reliability, and user-friendliness. It is an important property of both standard and individual software, ensuring both that they can be integrated into existing software environments, and that today’s IS can be integrated with IS to be developed in the future. 

In this paper, we describe an approach of developing integrated information systems. It is database-centered and uses a methodology enabling software developers to plan, design, and implement compatible software systems. The methodology is called Integration Engineering. 

Integration Engineering 

Integration Engineering enhances software engineering by principles, methods, and tools supporting development and maintenance of compatible information systems. Integration Engineering has some ideas in common with Information Engineering (Martin, 1989), but there are also major differences. Both approaches take an enterprise-wide perspective, or at least they consider more than one information system at a time. However, the basic idea of Information Engineering is top-down modelling and developing new IS, i.e. it more or less
Related Content

INDUSTRY AND PRACTICE: A Metadata Management System to Support Data Interoperability, Reuse and Sharing
Stephanie Cammarata, Iris Kameny, Judy Lender and Corrinne Replogle (1994). Journal of Database Management (pp. 30-42).
www.igi-global.com/article/industry-practice-metadata-management-system/51134?camid=4v1a

Evaluation of MDE Tools from a Metamodeling Perspective
www.igi-global.com/article/evaluation-mde-tools-metamodeling-perspective/3393?camid=4v1a

Big Data in Massive Parallel Processing: A Multi-Core Processors Perspective
www.igi-global.com/chapter/big-data-in-massive-parallel-processing/198767?camid=4v1a

On the Efficiency of Querying and Storing RDF Documents
Maria-Esther Vidal, Amadís Martínez, Edna Ruckhaus, Tomas Lampo and Javier Sierra (2012). Graph Data Management: Techniques and Applications  (pp. 354-385).
www.igi-global.com/chapter/efficiency-querying-storing-rdf-documents/58619?camid=4v1a