ERP Implementation
The Blindspot in Software Engineering

Hopcroft Maureen and Goodland Michael
Kingston University, Faculty of Computing, Information Systems &
Mathematics, Penrhyn Road, Kingston upon Thames, KT1 2EE
Received 15/10/2008, Accepted 30/09/2009

ABSTRACT
The primary focus of software engineering (SE) has been on bespoke work, but many organisations now adopt a strategy of using Commercial-off-the-Shelf (COTS) products and Enterprise Resource Planning (ERP) systems to meet their requirements. The availability of extensive pre-existing functionality in these should reduce risk and improve performance and hence SE has not focused on this area. However, the Standish Group1 estimates only 10% of ERP implementations succeed with full functionality. Modern SE processes are comparable to those adopted by ERP business implementers, but do not overtly address ERP specific issues. Hence, they offer insufficient guidance to organisations and practitioners. The key issue is that, given pre-existing functionality, organisations need to make significant strategic and tactical decisions about whether to change the business to fit the system or whether to change the system to fit the business. This paper examines the current level of software process support for ERPs and the consequent implications for research. It analyses these issues from a theoretical and practical perspective, by relating the experience of ERP implementation to current and emerging SE processes.

Keywords: Enterprise Resource Planning (ERP), Commercial-off-the-Shelf (COTS), Requirements Engineering, Software Process, Software Engineering

1. INTRODUCTION

Software engineering (SE) emerged as a discipline concerned with all aspects of software production. The focus has been on bespoke work and given that the code pre-exists within ERPs, SE has provided minimal commentary on them. It can be argued that ERPs fall outside the remit of SE. This view seems consistent with ERP literature, the vast majority of which is targeted at the wider business implementation. However, it is important to note that SE does not simply address technical development, the evolution of SE has continually identified that issues may arise across a broader lifecycle and processes have been developed beyond a purely technical remit to embrace this. Processes cover areas related to feasibility, requirements, prototyping and implementation. These areas are the source of many issues in ERP implementation. Further, ERP implementations increasingly include technical work through enhanced configuration features and the need for extensive integration with other systems. The full SE lifecycle in more recent processes is increasingly comparable to proprietary ERP lifecycles used by vendors and consultants.

These factors suggest it is apposite for SE to examine where it can add value to ERP implementation. The need for research in the area has been brought to the fore by Sommerville[2, 3], initially in 2005 at the JENUI conference and more recently revised as a keynote presentation at ASWEC in 2008, where he 'discusses the changes to the ways in which commercial software is developed by the configuration of COTS and how the research community should respond to this.'

This paper provides some response to this challenge. The core of the paper is an analysis of two ERP implementations which use differing SE processes. This is used to inform the derivation of requirements for an ERP process and to analyse the degree of fit of existing SE processes to those requirements.

It is structured as follows: section 2 clarifies definitions for COTS and ERPs; section 3 analyses current COTS research, its applicability to ERPs and outlines the theoretical proposition; section 4 analyses two ERP implementations using differing SE processes to identify degree of fit and root cause of issues; section 5 uses this analysis and existing research to derive ERP requirements for an SE process against a generic comparison evaluation framework and to analyse existing processes against these requirements; section 6 draws conclusions and highlights areas for future research.

2. DEFINITIONS

An ERP system as defined by Bingi et al[4] is an integrated software solution that spans the range of business processes that enables companies to gain a holistic view of the business enterprise. It promises one database, one application, and a unified interface across the enterprise.

ERP implementation is not explicitly covered in SE literature. ERPs have similarities with COTS products on which there is emerging research.

The characteristics of COTS products as defined by Albert and Brownsword[5] can be summarised as products that are offered by a vendor for profit via selling, leasing or licensing to the general public. These products are supported and evolved by the vendor who retains all intellectual rights. They are available as multiple, identical copies and are used without any modifications to the internals of the system.

3. CURRENT RESEARCH

The premise is that SE needs to provide increased guidance to ERP implementations. ERPs have similarities with COTS products on which there is emerging research. This raises two initial questions: what is the state of COTS research and how applicable is this to ERP implementation?

The acquisition and implementation of COTS products brings new problems to the SE discipline. It is no longer feasible to assume that traditional software processes, along with an understanding of business requirements, will enable delivery of a usable system. The success of COTS products is based on faster application development combined with lower total cost of ownership (TCO).The Standish Group's 2000[1] survey shows an industry-wide 54% usage of COTS based applications (CBAs). Primary research to support these is either the value based processes for CBAs[6] or based on the major functions around CBAs as in SEI EPIC process[5].

The SEI EPIC process provides a framework within which to manage CBAs with 'what' but not explicitly 'how' or 'when'. Yang et al[6] state that although SEI EPIC provides insights on important COTS considerations, '…it lacks intermediate milestones and has at least three major problems: it lacks guidance on which steps to perform next, generates very little status information and increases the likelihood of non-convergence due to its 'study and wait' cycle'. The value based processes for CBAs proposes skipping requirements, but an earlier study by Morisio et al[7] suggests that requirements definition omission is dependant on whether the domain of an application is stable. The discussion lends credence to Boehm's[8] '…some initial CBA development processes are emerging', but these are not yet standardised or proven.

The definition of a COTS product broadly covers a product offered by an ERP vendor. However there are important differences around the functional breadth of ERPs, which may cover a very broad range of an organisation's processes, the capability for configuration and the extent of integration with other systems. These differences create a step-change between COTS and ERPs. A COTS product is defined as being used without any modifications to the internals of the system, this is more often not the case for ERP implementations. Additionally, Albert and Brownsword's[5] SEI EPIC propose the scope of a COTS solution '…can be initially fielded in a period of six to twelve months'. ERP implementations typically take significantly longer than this. In his talk on 'Construction by Configuration', Sommerville[2, 3] mentions both COTS and ERP products inferring distinction.

Research to date has provided insight into the specific issues around implementing COTS based products. However, the research is in it infancy and the focus is on a wide range of COTS products. There is no specific reference to issues around enterprise level COTS products, ERPs. The lifecycle of ERPs is similar to, but more complex than, the average COTS lifecycle and presents additional challenges.

ERP specific research is hence required, but should be informed by continuing COTS research. This paper addresses this by considering ERP implementations within the context of three software processes to derive whether these are fit for use for ERP implementation. Two case studies are used, one using waterfall, one using an agile approach loosely based on Dynamic Systems Development Method (DSDM). The Rational Unified Process, (RUP) is then used as the basis for theoretical comparison. A generic comparison methodology, the Normative Information Model-based Systems Analysis and Design, (NIMSAD) framework proposed by Jayaratna[9] was used. Jayaratna's proposed set of questions was summarised by Avison and Fitzgerald[10]. The summarised questions have been used to evaluate the processes against the framework, identifying areas of strong fit, key gaps and areas of rudimentary support where the process could be tailored to fit an ERP implementation. This yields a 'fitness for purpose' of each software process to ERP implementation.

4. ERP IMPLEMENTATIONS

The ERP implementation case study was for two phases of an ERP implementation for a mid range public sector organisation. Conventional software processes, firstly waterfall, then quasi DSDM, were broadly followed. The analysis was based on:

- Review of key project documents
- Structured interviews at director and senior management level
- In-depth analysis of calls logged on a call logger and issues lists to identify root cause of issues
- User evaluation questionnaire targeting hands-on ERP end users

4.1 Waterfall Model

The first implementation broadly adhered to a waterfall model. It was tightly managed by a global management consultancy using a proprietary methodology with distinct phases and sign offs and was delivered to time and budget. There was a detailed requirements document, but no detailed specifications were produced as the system was pre-existing. As the requirements had been developed prior to the implementation, business processes were implemented 'as-is'. The system was configured with a minimum of customisations, manual work arounds were used to balance system functionality and as-is process. Data migration and interfacing started late and were under-estimated.

Senior managers felt benefits had not been realised and over 50% of users felt that the ERP needed major customisations.

Available issue lists and call logger issues were analysed. Issues were categorised as: functionality, either around configuration or customisation, (41%), training, where additional training was required (36%) and reporting (3%) as shown in Figure 1.

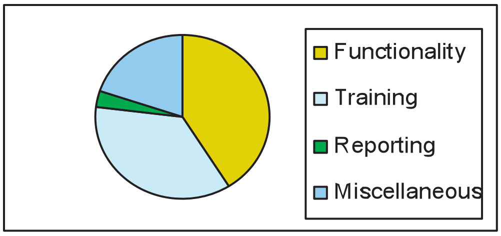

Figure 1. Waterfall Model Implementation Issues.

Further detailed analysis identified the root cause of the issues. For example, a functionality issue may represent problems in requirements definition or configuration, or processes that were not working as expected. All 41% of the functionality issues could be traced back to requirements, as missing, poorly defined, poorly designed and implemented or deferred requirements as shown in Figure 2.

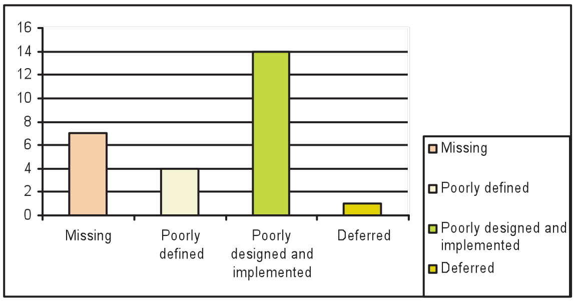

Figure 2. Waterfall Model Requirements Issues.

4.1.1. Waterfall Model - Key Issues

The use of the waterfall model can in some aspects be seen as broadly successful. It produced good quality software with few ‘bugs’ and fulfilled most of the departmental processes. However, the system was not widely accepted by users or senior management and there were missing and problematic requirements. The system was being used at a sub-optimal level, since there was inadequate synchronisation between the business processes and the ERP. Several of these issues have their roots in how the SE process was used with an ERP implementation.

The processes have been implemented as-is with insufficient attempt to implement new processes or tailor processes to fit the ERP. The requirements were developed prior to the selection of the ERP and pre-suppose a given set of business processes. Within the waterfall process there was no stage to iterate processes, requirements and pre-existing functionality. This raises several issues for SE. Firstly, when and to what level requirements should be developed. Historically, business processes were developed prior to the SE lifecycle, and then requirements were developed. The inter-linkage between business process and requirements is now much tighter and they must be cognisant of each other. Second, there needs to be guidance on how to balance tailoring of the ERP to fit requirements or requirements to fit the ERP. This is a fundamental trade-off for user requirements, project cost and future upgrade path.

There were a large number of missing, or poorly defined/implemented requirements. The root cause was insufficient iteration between user and designer. In a bespoke software project, the designer will analyse requirements to design a solution. They will challenge the user as to the underlying nature of the requirement. This highlights missing or poorly defined requirements earlier in the process. With a pre-existing system, much of this activity is by-passed. This was exacerbated by the fact that no functional specifications were produced. This was seen as unnecessary given the system existed. However, it has left users without a clear overall view of the system functionality. The issue for SE is that with existing software there is need for a clearer process and artefacts to reflect activities at the design stage.

Data migration and interfacing started late. The root cause is that the ERP lifecycle focuses on the ERP and configuration or customisation. However, integration and data are critical to the overall solution and need to be addressed early.

Lack of user 'buy-in' and sub-optimal usage. The root cause was inadequate change management, support and training. There was no departmental champion or super-user who understood the system, acted as a champion and could derive the best from it. These are requirements for any successful implementation. However, they are more important in an ERP, where the users will have had less flexibility in defining how the system will behave. Conversely, an ERP will almost certainly offer better in-flight user configuration than a bespoke system, allowing minor improvements to be made on an ad hoc basis. If managed correctly this can be a significant positive aspect of an ERP.

4.2. Agile Approach

The second implementation broadly followed a loose agile approach, a quasi DSDM model. Requirements scoping workshops comprising up to three ERP functional consultants and eight to ten users were conducted to gain high level requirements. The workshops facilitated discussion focussing on the users' requirements against the functionality offered by the ERP. The workshop notes were documented. Prototyping was then used to clarify these requirements but there was no definitive requirements document. A gap analysis was done to identify the deficiencies of the ERP against business requirements together with business importance and impact. The outcome of the prototyping and gap analysis provided the basis for necessary customisations. This led to a substantial amount of customisation, at least sixty programs that were specified as both functional and then technical specifications.

The functional configuration was not fully documented, which made it difficult to link business processes to program objects and then to testing scripts to ensure that all programs required to run the process had been configured or built. This was followed by incomplete user acceptance testing, with a lack of end to end business activity scripts resulting in a lack of end to end process testing.

From a project management perspective, there was far less rigorous implementation control. The project lacked a robust method and plan leading to ill defined implementation phases and issues on sign-off and go live decisions. The data migration was poorly specified and the complexity underestimated resulting in an extended cut over period. User training was conducted before the system was completely built and configured.

Senior management felt that the second phase of the ERP implementation was too 'hot on the heels' of the first phase and that the organisation would have benefited from a 'bedding down' period giving first phase users time to 'buy in' to the ERP and time for the business to address second phase detailed requirements.

An overwhelming majority of users (over 70%) felt that the ERP had not improved operations in their department. A similar percentage believed that the ERP did not provide the functionality needed to do their jobs and that the system was not easy to use and was more time consuming. They did not feel that more training or support would facilitate more effective use of the ERP.

Available issue lists and call logger issues were analysed. These were categorised as: functionality, either around configuration or customisation, (40%), training (19%), reporting (17%), enhancements (24%), request for functionality that was not delivered at go-live or the request for new functionality, as shown in Figure 3.

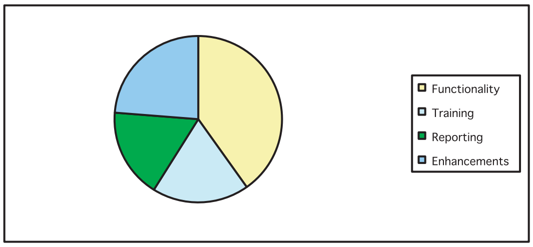

Figure 3. Agile Quasi DSDM Implementation Issues.

Root cause analysis identified that 68% of the issues could be traced back to requirements as shown in Figure 4, the majority in the software category. There were few core software problems but many problems with customisations.

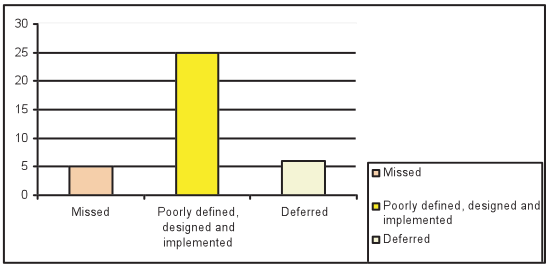

Figure 4. Agile Quasi DSDM Requirements Issues.

The second phase implementation had project management issues that were independent of the fact that an ERP was being implemented and issues need to be carefully weighed to take account of this. However, it is still possible to draw conclusions on the ERP implementation and the use of quasi DSDM to support that. The key issues, their root cause and the issues for SE to address are summarised below.

Poor fit between the ERP module and high level business requirements, leading to poor requirements definition, large scale customisation and problems with future product upgrade path. The root cause is that the decision to implement the module of the ERP was taken on the basis of a desire for an ERP strategy without an assessment of the degree of fit to the core business requirements. A more detailed assessment may have led to a different strategy, for example using a best of breed COTS system as a plug-in to the ERP, or, if the ERP was the right strategic option, a different software process could have been used that was more appropriate to heavy customisation. The issue for SE is that the software process needs to start earlier in the entire project lifecycle. With ERPs, there is a blurring of business process design, IS strategy and the SE process. The SE process needs to assist in determining when the degree of fit is adequate to implement and customise, or when an alternative best of breed or bespoke plug-in to an ERP is required.

There is a large number of missing requirements and significant requirements that emerged very late in the implementation. There were also many poorly defined or poorly implemented requirements. The root cause is there was no baseline requirements document. If a high level requirements specification had been produced, this could have been used to drive a gap analysis to identify the degree or lack of fit between requirements and the ERP and potentially adopt a different strategy or software process. The SE issue is again the role of requirements where software pre-exists. There is some debate in SE as to the extent to which requirements are needed in a COTS environment[6, 7]. However, in this instance the absence of requirements caused major issues.

Further, quasi DSDM was not effective in driving out requirements and design to meet them. While part of this was due to inadequate project management and SE disciplines, there was no SE guidance on how to conduct the activity in an ERP environment. There needs to be a clearer software process and artefacts to reflect the key activities in the design stage where software pre-exists.

5. ERP REQUIREMENTS AND SE PROCESS SUPPORT

In order to determine an SE process to support ERP implementation it is necessary to establish a framework for the process, to derive key ERP requirements against that framework and to identify key differences with existing SE processes. The NIMSAD evaluation framework is used to structure the ERP requirements and as the basis for evaluation. Table 1 below evaluates and assesses the software processes, waterfall and quasi DSDM from the case study and RUP, for theoretical comparison, against NIMSAD and ERP derived requirements.

Table 1. ERP Derived Requirements and SE Process

NIMSAD Criteria	ERP Derived Requirements	Waterfall	DSDM	RUP
Problem Situation	Management and organisation buy-in Organisation wide process/data model Market awareness	Outside process None None	Limited support None None	Some support None None
Problem Solving Formulation	Understanding of system context, boundaries and organisation impact Business process as-is and to-be models, gap analysis, high level requirements	No existing functionality support Modelling tools exist	Prototyping could support existing functionality	Prototyping could support existing functionality
Problem Solving Solution Design	Iterative approach to trade-off between requirement/customisation	No rules to manage trade-off	No rules to manage trade-off	No rules to manage trade-off

Based on practical experience from the ERP implementation and on emerging academic work on COTS based systems[5, 6, 7, 11, 12], there are two broad aspects:

- ERPs require *broader* interaction with the wider business strategy and project lifecycle;
- ERPs require *deeper* work in some areas of the SE process or lifecycle. The most significant is the need to a seek trade-off between business process and systems modification.

The three software processes are primarily used to develop systems from scratch. The criteria above against the NIMSAD framework show that the requirements for implementing a system that is already built are considerably different to those required for building a system. The key areas for concern are:

- *The Problem Situation*
- *Problem Solving - Formulation*
- *Problem Solving – Solution Design*

In the *problem situation phase* none of assessed software processes look at the market either at the inception or at the end of the lifecycle. This is fundamental to the success of an ERP implementation and hence a key weakness of the SE processes. SE processes have not needed to address this as their remit has been to produce systems or code from scratch. However, for an ERP implementation this has an absolutely profound affect on the system. It is key to the initial selection, the extent to which all ERP modules are implemented or plug-in best of breed solutions are used as complimentary and to the future ERP release adoption strategy.

The other factor in the *problem situation* is organisation wide buy in, an ERP may affect the entire organisation. This can partially be addressed as part of RUP in the business or domain modelling and could be partially addressed by DSDM in the feasibility study. However, neither process takes the holistic picture necessary to understand the ERP footprint across the organisation.

The key issue in *problem solving formulation* is the timing and role of requirements. If these are developed too early and rigidly it will be difficult to take advantage of best-practice business processes and functionality trade-off inherent in the ERP. If they are developed too late, or not developed, then there is a high risk of major gaps, missing and poorly defined or implemented requirements.

Problem solving solution design, the key issues here are the need for an iterative process to determine when to customise the ERP and when to customise the business process. This area needs to be addressed with specific problem solving techniques that are easily understandable by stakeholders and that can be modelled as part of the conceptual design.

6. CONCLUSION

ERPs may be implemented broadly in line with the waterfall model, DSDM or RUP. However, none of these software processes overtly address ERP specific issues. The key difference between an ERP and bespoke implementation is that the bulk of the code pre-exists and arguably represents 'best-practice' business processes. Stakeholders need to make strategic and tactical decisions about *whether to change the business to fit the system or the system to fit the business*. From an SE perspective the key issues this raises are:

Business Strategic**:** a strategic decision to implement an ERP may constrain parts of the business to use prescribed business processes and functionality and will constrict users to a specific look and feel. There hence needs to be strong market awareness of product functionality and direction and wide organisational buy-in to the solution. Historically, market awareness has not been a part of software processes this needs to change as the reliance on COTS products increases. SE needs to contribute to this debate, bringing the disciplines of business process, information systems (IS) strategy and SE closer.

IS/SE Strategic**:** Historically business processes were designed and used to drive requirements. Since business process and functionality exist in an ERP this can no longer be sequential. This raises key SE issues around the timing, role and definition of requirements: how far should stakeholders go in defining requirements since the system already exists? If no requirements are produced there is no traceability or proof the system will meet user needs. If a requirements specification is produced too early and too detailed, this may constrain use of the ERP. There is work evolving in this area, but no current consensus in SE research that addresses, for example, whether detailed requirements are necessary in COTS implementation[5, 6, 13]. Yang et al[6] have suggested that there should be no requirements in a COTS environment. However, this was a key root cause of the problems with the implementations cited. A clear requirements definition and gap analysis would have identified the high level of missing functionality. A clear SE process is required.

Tactical Level: During prototyping and/or specification, users constantly need to make a trade-off between changing the business or changing the system. There is no defined SE process for this, or defined artefacts or rules on how to make the trade-off. There is generally no specification of what the users have agreed, since the system exists, and hence no requirements traceability.

The key questions that need answering are: what is the optimum artefact to denote requirements to the users; what evaluation mechanisms should stakeholders use to make this trade-off?

This paper serves as a precursor for further research. Future work will include more diverse case studies to identify and analyse root causes of ERP implementation issues, examination of existing SE processes their relationship with emerging COTS processes and how these could be brought together to provide an SE process to support ERP implementation.

AMR Research[14] estimates a $US30 billion market for ERPs. The current focus of ERP vendors is scaled down versions for the mid market, for example, SAP's Business One and Oracle's Special edition. These will expose a much wider ERP user base who cannot necessarily afford the expensive consultancy support. SE needs to provide explicit guidelines that will improve the success rate of ERP implementation in organisations.

As Boehm[8] states '… there are still major challenges for the future …processes for enterprise-level COTS and system-of-system COTS'. The need for this research is urgent.

REFERENCES

[1] Standish Group, The Cost of ERP http://www.standishgroup.com/chaos/beacon_243.php, 2002

[2] Sommerville, I., "Construction By Configuration: A New Challenge For Software Engineering Education.", 2005 http://www.comp.lancs.ac.uk/computing/resources/IanS/Ian/Lectures/ConsByConfigSEEducation.pdf

[3] Sommerville, I., "Construction By Configuration: An Opportunity For SE Research", 2008 http://www.cs.st-andrews.ac.uk/%7Eifs/Talks/ASWEC2008-Keynote-Sommerville.pdf

[4] Bingi, P., Sharma M.K., Godla, J.K., Critical issues affecting an ERP implementation, *Information Systems Management*, Vol 16, No. 3, 1999, pp. 7–14

[5] Albert, C., Brownsword, L., "Evolutionary Process for Integrating COTS-Based Systems (EPIC): An Overview, Technical Report, CMU/SEI-2002-TR-009, Software Engineering Institute, Carnegie Mellon University, 2002, http://www.sei.cmu.edu/publications/documents/02.reports/02tr009.html

[6] Yang, Y., Bhuta, J., Port, D., Boehm, B., "Value-Based Processes for COTS-Based Applications," *IEEE Software*, Vol 22, Issue 4, July 2005, pp. 54-62

[7] Morisio, M., Seaman C., Parra, A., Basili, V. R., Condon, S., and Kraft, S., "Investigating and Improving a COTS-Based Software Development Process", *Proceedings of The 22nd International Conference on Software Engineering* (ICSE 2000), Limerick, Ireland, June 2000
http://www.cs.umd.edu/projects/SoftEng/ESEG/papers/83.83.pdf

[8] Boehm, B., "The Future of Software and Systems Engineering Processes." , 2005 http://sunset.usc.edu/publications/TECHRPTS/2005/usccse2005-507/usccse2005-507.pdf

[9] Jayaratna, N., *Understanding and Evaluating Methodolgies NIMSAD: A Systemic Framework*, McGraw-Hill, 1994.

[10] Avison, D., Fitzgerald, G., *Information Systems Development: Methodologies, Techniques and Tools*, Fourth Edition, McGraw-Hill, 2006.

[11] Morisio M., Seaman C., Basili V. R., Parra A., Raft S., and Condon S., "COTS-based Software Development: Processes and Open Issues," *Journal of Systems and Software*, Vol. 61, no. 3, April 2002, pp. 189-199.

[12] Yang, Y., Bhuta, J., Boehm, B., Port, D., Abts, C., "Composable Process Elements For Developing COTS-Based Applications," *2003 International Symposium on Empirical Software Engineering* (ISESE '03), 2003, vol. 00, no., p. 8.

[13] Torchiano, M.,Morisio, M., "Overlooked Aspects of COTS-Based Development," *IEEE Software*, vol. 21, March/April 2004, no. 2, pp. 88-93.

[14] AMR Research (2005) Reilly, K., "AMR Research Releases Report Showing Overall European Market for ERP Vendors to Grow 7% Annually Through 2009," http://www.amrresearch.com/