Toward an Integrated Framework for Modeling Enterprise Processes

Enterprise process modeling is the most important element in the design of next-generation ERP systems.

Imagine that a large company has spent hundreds of millions of dollars and three years implementing an enterprisewide information system. But when the system goes live, the company discovers the system is incapable of supporting the volume and price structure of its distribution business. The project fails, and the company suffers monumental losses and is ultimately driven into bankruptcy. The company sues the vendor of the software package, blaming it for its losses. A hypothetical situation? Hardly. This disaster story is as true as it is regrettable and avoidable.

Since the mid-1990s, many large- and mid-size enterprises have implemented off-the-shelf enterprise software packages (also called enterprise resource planning, or ERP, systems) to integrate their business activities, including human resource management, sales, marketing, distribution/logistics, manufacturing, and accounting. Enterprise systems promise not only information integration but the benefits of reengineered and radically improved business processes as well. “The business world’s embrace of enterprise systems,” according to [4], “may in fact be the most important development in the corporate use of information technology in the 1990s,” an assessment that’s just as valid today. However, despite a few dramatic successes, many companies still reportedly fail to realize these benefits while incurring huge cost and schedule overruns.
Given these mixed results, what should companies do? Rejecting enterprise software as an enterprise-integration solution altogether would be foolhardy, given the multimillion-dollar investments many companies have made in the software and the partial success many of them have realized. Moreover, next-generation enterprise software from such vendors as SAP, Oracle, and PeopleSoft is evolving rapidly, promising to improve flexibility, implementation, and support for the extended enterprise through modules for customer relationship management, advanced planning systems, supply chain management, and collaborative commerce in a Web-based environment. Consulting firm Gartner Group estimates that by 2005, this next generation of ERP it calls ERP II [11] will replace current ERP systems, thus requiring companies to upgrade. Enterprise process modeling is crucial to the design of ERP II systems.

A common source of difficulty implementing enterprise software involves management’s understanding of its own business processes [6]. A business process, which is different from a traditional business function, is typically cross-functional and involves the reciprocal or simultaneous flow of information between two or more functional areas, as well as among the functions within these areas. For example, the order-fulfillment process involves inputs from sales, logistics, manufacturing, and finance, as it progresses from sales order entry, to delivery of the product, to the final step of collecting cash payment from customers. Business processes, including order-fulfillment, procurement, and product development, hold the key to the financial success of an enterprise. In theory, an enterprise system is ready to support business processes because it encapsulates best business practices, or the tried and successful approaches to implementing business processes, and is hence the ideal vehicle for delivering the benefits of an integrated cross-functional approach.

However, “As many companies get ready to implement standard software, they encounter the problem of how to simplify and model the enormous complexity of their business processes” [6]. The result is that companies often face the dilemma of whether to adapt to the software and radically change their business practices or modify the software to suit their specific needs.

Even if they decide to modify the software, they still face maintenance and integration issues. Many enterprise systems today are notoriously inflexible with respect to process specification and implementation [4]. Moreover, the packages are difficult to change and extend due to their complex proprietary application program interfaces and database schemata—a far cry from proposed open standards of e-commerce [8]. Even if a company were to overcome this barrier, modify its software, and painstakingly build complex interfaces with other information systems, its maintenance and integration issues would still not be completely resolved. The modification trauma is reexperienced every time the enterprise software vendor issues a new release of its software. To be sure, leading ERP vendors are working to resolve these issues, though much remains to be done to realize the ERP II vision.

A holistic solution approach garnering considerable researcher attention calls for renewed focus on enterprise process models instead of on technologies alone. It envisions enterprises having the flexibility to redesign enterprise processes—regardless of whether the new processes are derived from clean-sheet process reengineering unhindered by technological considerations or whether industry-standard best practices are incorporated into the software. From this perspective, process models that are easily created, modified, and analyzed greatly aid process-reengineering efforts to realize the promised benefits of enterprise systems. Hence, researchers and managers are increasingly interested in techniques, existing and new, for business-process modeling, specification, implementation, maintenance, and performance improvement [9].

We've developed an enterprise process-modeling framework that can serve as the foundation for next-generation enterprise systems. Here, we outline some limitations of existing enterprise modeling techniques and architectures, describe a prototype implementation of the framework, and conclude with the significance of this work.

Enterprise Process Modeling Techniques

Many techniques for modeling enterprise processes, including Data Flow Diagrams (DFDs), Integration Definition for Function Modeling (IDEF0), and activity diagrams in the Unified Modeling Language, have their roots in process modeling for software development. In 1992, [3] reported “Process modeling work is still young, and the span of the research agenda is still being formulated. Nevertheless, work to date holds promise for benefits in management, process-driven environments, and process reengineering.” However, this promise has been only partially realized with the evolution of a number of techniques, architectures, and frameworks focusing on modeling the enterprise in general and business processes in particular. Such approaches focus on modeling the enterprise in order to reengineer or redesign business processes with the help of information technology. In a broader context, the technology itself is but a part of a greater whole, including the enterprise, the supply chain, and entire groups of related industries. Techniques include the Computer Integrated Manufacturing Open System...
Architecture business process modeling approach, the Integrated Enterprise Modeling approach, the Purdue Enterprise Reference Architecture, the predicate-logic-based Toronto Virtual Enterprise method, Baan’s Dynamic Enterprise Modeling method, and SAP’s adaptation of the Event-driven Process Chain method (part of the Architecture of Integrated Information Systems). Each provides a basic set of constructs to model enterprise functionality [5] but also involve four major gaps:

Need for a theory base. Existing process models are descriptive but lack prescriptive capabilities. That is, they do not provide business modelers or system architects a formal theoretical base from which business processes can be analyzed in a rigorous, quantitative manner. This gap is serious; formal analysis is essential for learning the effect of changes in process logic and parameters on business performance measures in the interests of making better business decisions. Moreover, an underlying formalism would help the enterprise system architect generate multiple, whole-process views of the enterprise at various levels of abstraction. Such a capability is essential for managing the enterprise.

Need for modeling and implementing distributed computing. Many existing process modeling techniques do not explicitly incorporate the distributed computing paradigm of the Internet and lack the syntax and semantics necessary for modeling the distributed enterprise and for designing and implementing the process model in an Internet-based environment. Still needed are modeling techniques compatible with the distributed infrastructure of the Internet. Companies using mid- and large-scale ERP systems are dispersed geographically, making it imperative that their users be able to collaborate in creating, modifying, and analyzing process models from any location at any time. The need for distributed access to process models is even greater for next-generation extended enterprises and virtual organizations. The process models in the enterprise software toolkit have not kept pace with other developments in the computing paradigm.

Need for new process redesign semantics. Many process modeling techniques, especially those originally designed for software development, including DFDs, are general-purpose by design. As a result, they lack explicit semantics for enterprise-oriented concepts like cost and time. While enterprise modeling architectures and workflow software for process redesign allow for these concepts, they are not generally tied to enterprise software. Moreover, information on cost drivers and process performance measures, including time, quality, and efficiency, are not readily captured in existing enterprise modeling systems. The challenge for the architect is to create a simple and usable process-modeling technique that also represents enterprise-oriented semantics.

Need to link business and engineering processes. Conventional wisdom points to the fact that business results are tied to physical processes, whereby resources are converted to products satisfying market demand. However, contemporary process modeling approaches do not adequately reflect the interrelationships between business and production engineering. Engineering approaches generally focus on physical conversion at one end of the process spectrum, while business approaches focus on market and financial strategies at the other end. To be effective, process design, control, and improvement demand the use of modeling methods with scalable and dynamic properties providing seamless links between business and technical process issues.

Holistic Management
Together these four needs support the case for an integrated process modeling framework. Our framework thus takes an interdisciplinary approach, with inputs from information systems, accounting, computer science, industrial engineering, and business disciplines (see Figure 1).

Any modeling approach must keep the user in the loop. The strengths of popular process modeling techniques, including DFDs and IDEF0, reflect their simplicity; even novice end users readily understand the associated graphical symbols and terminology. The framework emphasizes business users and specialized modelers who create, modify, analyze, and use enterprise process models.

These models have at least two layers: front-end graphical and back-end formal. A theoretical base is established by well-defined mappings between the user-oriented graphical model at the front end and its corresponding formal representation at the back end. The mapping is two-way; a formal representation can be generated from a user’s graphical model and vice versa. Note that the mappings are more than translations. Analysis performed at the back end may provide inputs
to modify the front-end graphical model and vice versa. The framework entails a process modeling language incorporating various process improvement methods, as shown in Figure 1.

Implementation. We demonstrated the framework’s feasibility with a proof-of-concept implementation at the Center of Computer Integrated Manufacturing at Oklahoma State University with four components: a graphical process modeling technique; Petri net theory providing a formal theory base; XML for mapping the front-end graphical layer to the formal Petri net layer; and a Web-based software prototype.

The front-end modeling layer is based on our newly developed graphical process modeling language—the Enterprise Process Modeling Language, or EPML [1]—which builds on such existing process modeling techniques as DFDs, IDEF techniques, and SAP’s Event-driven Process Chain technique. Figure 2 outlines a model created in EPML involving approval of purchase requisitions. A requisition is first assigned to an approver; if no approver is available, the requestor is prompted to take appropriate action. If an approver is available, the requestor is informed, and a record is maintained of the status of the requisition. Once the approver processes the requisition, the requestor is notified of the rejection or approval outcome.

The graphical models created through EPML are then formalized as Petri net models. Although Petri net theory is used here, the framework is general-purpose, allowing for the coexistence of other formalisms as well. Petri nets provide a strong mathematical foundation for modeling and analyzing concurrency, choice, asynchronous completion [7], state transitions, and other aspects of business processes [2, 9]. Petri net-based analysis results in such quantitative summary measures as throughput and response time. Petri nets have also been used for modeling workflows and verifying the correctness of control flow in business processes [2, 9, 10].

The translation between the front-end EPML model and the back-end Petri net representation is achieved through an XML-based markup language; Figure 3 outlines this approach using a subset of a larger model we created for a representative next-generation enterprise involved in the direct selling of computers to end customers.
culation needed to create the final product or service. Once correctness of control flow is established, it is necessary to ensure the correctness of the resource assignment policies and input-output specifications. This second step involves enrichment of the TSD with the related areas must collaborate to develop effective tools, techniques, and methods for ERP II. Architectural issues underscored by this work include: building holistic process models that link business and technical parameters; integrating—semantically, logically, and physically—process submodels created by distributed users; linking descriptive models to underlying formal analytic models; and linking process models with the overall logic of enterprise systems.

### References

8. Radding, A. ERP, Componentization, and E-commerce, 1999; see irmagazine.earthweb.com/erp/print/0,,11981_616621,00.html.

Nikunj P. Dalal (nik@okstate.edu) is an associate professor of information systems in the Department of Management Science and Information Systems in the College of Business at Oklahoma State University in Stillwater.

Manjunath Kamath (mkamath@okstate.edu) is a professor in the School of Industrial Engineering and Management and director of the Center for Computer Integrated Manufacturing at Oklahoma State University in Stillwater.

William J. Kolarik (kolarik@okstate.edu) is a professor and head of the School of Industrial Engineering and Management at Oklahoma State University in Stillwater.

Eswar Sivaraman (esivaram@jmu.edu) is a visiting assistant professor in the Department of Systems Engineering and Operations Research at George Mason University in Fairfax, VA.

This research was supported by a grant from the National Science Foundation (Award No. 0075580) under the Scalable Enterprise Systems Initiative, the research team included M. Kamath, N. Dalal, W. Kolarik, and A. Lau.

© 2004 ACM 0002-0782/04/0300 $5.00