The Missing Link: Systems Integration in the Curriculum

Robert L. Rariden
Illinois State University
Normal, Il 61790

Abstract

This paper considers the need to include components of the body of knowledge required to do systems integration into the Information Systems curriculum. The fact that systems integration, often known as 'Enterprise Application Integration' (EAI), is often connected to the now common occurrence of bringing an Enterprise Resource Planning (ERP) package into an organization is addressed. Based on this fact, and the mandate to support legacy applications, the incorporation of various components into the Information Systems curriculum is advocated. Various system integration components which might be included in such a curriculum are identified and described.

Keywords: Systems Integration, Enterprise Application Integration, EAI, Systems Analysis, Systems Design

The Missing Link: Systems Integration in the Curriculum

1. Introduction

This paper addresses the issue of incorporating elements from the body of knowledge used by systems integrators into the Information Systems / Information Technology (IS/IT) curriculum. Program (and system) maintenance and application integration are two of the pressing issues facing IS/IT today. IS/IT departments have had legacy problems for many years and so maintenance, while a serious problem, is not a new issue. Now, however, organizations buy "best of breed" systems and applications with the intent of integrating them with the balance of the organization's information technology portfolio. "[T]he integration and interoperation of dissimilar computing systems and application software for the purpose of information sharing and software reuse has become an essential and critical task." (Li, 2001) Further, "[t]hese projects often represent the single largest investment in an information systems project in the history of these companies, and in many cases the largest single investment in any corporate-wide project." (Li, 2001)

2. Background

Systems Integration is the integration of multiple, diverse, systems and applications into a larger system which transparently provides functionality to users as if the suite of integrated components were one large system.

Systems integration is often connected to bringing an ERP package into the organization. The effort can result in a host of questions relative to corporate risk as well as technical issues related to the integration. (Sumner, 2000) The effort can be so demanding that it may require re-engineering "business processes to "fit" the package, rather than trying to modify the software to "fit" the organization's current business processes." (Sumner, 2000)

A specific focus of systems integration may be involved with 'Enterprise Application Integration' (EAI). Systems integration as EAI is fundamentally a response to the history of creating large monolithic single-purpose applications. And while there are good historical reasons for such development, for years there have been pressing business reasons to leverage this diverse hodgepodge of platforms and development approaches. No business person wishes to simply discard a capital asset. Thus after years of building "islands of automation" users and business managers are demanding that seamless bridges be built in order to link these islands together. The effect of this pressure is to essentially merge the disparate, stand-alone, applications into what functions as a single, unified, corporate-wide application. Systems integration identifies the skills, tools and techniques for this task. In these cases EAI is the application of the systems integration body of knowledge within the enterprise context. If done fully and correctly, EAI allows the virtually unrestricted sharing of

data and business processes between applications and data sources both within the enterprise and between business partners.

The systems integrator must be knowledgeable enough to make the required, potentially sweeping, changes to application interfaces and shared data stores as necessary. The possible impacts of error in this area are evident. Thus the integrator must be able to analyze and understand both business process and data. They must have a vision of transitions from the current state to a future state and confidence in their ability to bring about the required transformation. The systems integrator, of course, does not set the goals here. The business, either through its subject matter experts (SME), business managers, or both, must identify the areas within which suitable processes and data required for the integration are to be found. As a result, the actual activities of the systems integrator usually take on several dimensions, including a data dimension, a methods dimension, an application interface dimension, a user interface dimension, as well as others.

3. Need for Systems Integration Concepts in the Curriculum

There are, then, two drivers mandating that the organization acquire system integration capabilities within its personnel. The first is the issue of legacy applications. As an organization develops and uses applications over time certain applications become indispensible to its operations. These are not likely to be discarded easily. Thus, if they are to work with newer applications they must be integrated together with them. The second is the practice of buying "best of breed" applications to get high performance functionality within a defined scope. Almost by definition it is impossible for an organization to purchase a suite of applications which will support its entire scope of operations without modification. Thus the need to integrate whatever the total set of systems needed to support the entire operation of the enterprise.

When IS/IT students are hired by an organization they are most often first put onto maintenance tasks. What we can expect for the future is that a high volume of systems integration work will require them to work on systems integration projects as well. In general, programming and systems development is of the same cloth as program and systems maintenance. The knowledge required to fix code is essentially the same as that required to generate the system in the first place. But the devil, as usual, is in the details. Maintenance problems impose far more constraints on what can be done than original development does. The latter is done in a context of a relatively clean slate while the former is not. The skills and techniques used to identify, represent, and work with these constraints differ from those used in new development. Interfaces need to be identified in common formats, existing functionality needs to be utilized according to its given design, the allocation of functionality is defined by the units to be integrated, and so on. The challenge includes mastering new techniques as well as adapting them to new situations. (Kobryn, 1999)

The true implications of design from an organizational perspective, including maintenance, are not seen in the academic curriculum for the most part. Obviously academicians are focused on introducing new material, building on a firm foundation, and showing the relationships between the concepts introduced throughout the educational experience. But what about the everyday challenge of fixing what wasn't broken, but now is? Or that of making multiple systems, each developed with no knowledge of the other, work together? In some ways systems are integrated in order to resolve large issues which, from one perspective are maintenance, but are so big for any of the existing systems that it is just simpler to add massive new functionality. Another facet of systems integration is to get systems which were not intended to talk to each other to be able to do so. Exposure to these concepts is mandatory in today's IS/IT world.

These activities are supported by knowledge taught in systems analysis and design courses, to be sure, but need to be supplemented with additional methodologies, architectures, and technologies. This is due to the large number of concepts which are currently not core to systems analysis and design as considered from the perspective of new system development.

4. The Challenge to Curriculum Design

To some extent all IS/IT programs are constrained by credit limits. In some cases these constraints are extremely severe. Where the IS program is a component of another degree the overall curriculum may allow for an introductory course followed by a two course programming sequence, then a database course, and one course in systems development concepts. At best this only leaves room for an elective or two.

Programs which are oriented towards a stand-alone IS major often include a second systems development course which tries to go full life-cycle with a project. The intent is to take a project from concept/proposal to implementation. The idea includes incorporating issues of project management, organization fit and feasibility in the case of live projects, application of design to requirements specification, etc. What this latter type of course can hope to accomplish is to give the student a positive experience, working with a group, in which most, if not all, of the

important systems concepts can come to life. The prerequisite for this even being possible, given the lack of experience which a student team will possess, is a clean slate. The project needs to be stand-alone and isolated in the sense that the standard "show-stopper" variables of a real organizational project can be controlled and/or mitigated. In essence it is important to avoid most of the real systems integration issues. Having guided close to a hundred of these projects the author can report that this is a standard cause for re-scoping student projects during an academic term. However this leaves the realities of the daily challenges faced by the IT professional unaddressed in the IS/IT curriculum.

The list of issues facing a systems integrator is matched by a set of technologies presently used to deal with the problems presented by integration. An example is the use of middleware to link disparate applications. The purpose of middleware is to provide a "glue" between applications. It is a communications area providing common functionality to otherwise disparate applications. There are multiple views regarding the strategy which should be adopted to provide this glue. Each of these strategies identifies standards which stipulate that certain functionality be provided in any implementation of the strategy. Message-Oriented Middleware (MOM), for example, can assure that every message sent will be delivered (as long as the node has not been permanently terminated). This is not a new technology, but it is not well covered in the standard IS/IT curriculum.

Students need to enter the organization with a firm understanding of the solution types which they will be tasked to provide. This knowledge requires strengths in any or all of the concepts introduced in the section below.

One approach to dealing with the curricular issue is to offer a distinct course addressing these topics. In essence a special topics course with a title such as "System Integration", "Enterprise Application Integration", etc. The author's current institution has addressed this problem in exactly this fashion This option, however, is only available to those curricula not completely constrained by accreditation standards. For those programs so constrained, the real solution may well be to revise and re-orient the curriculum which currently exists.

5. Knowledge Components for Systems Integration

A curriculum which prepares a student to engage in systems integration could introduce the following components. The components identified below were developed by the Object Management Group and are more fully explained in their literature as cited. The level of expertise which the student should attain is necessarily limited given the enormous amount of information referenced by these topics. It may be possible to acquire a working knowledge of certain of the topics in existing other courses. UML, for instance, could be introduced in the Systems Analysis and Design course(s). In other cases this may be more difficult. In those cases it may be possible to bundle multiple topics into a stand-alone course.

Standards and specifications for modeling (OMG, 2002)

Unified Modeling Language (UML)
A specification defining a graphical language for visualizing, specifying, constructing, and documenting the artifacts of distributed object systems.

Meta-Object Facility (MOF)
Standard interfaces that can be used to define and manipulate a set of nteroperable metamodels and their corresponding models.

Common Warehouse Metamodel (CWM)
Standard interfaces that can be used to enable easy interchange of warehouse and business intelligence metadata between warehouse tools, warehouse platforms and warehouse metadata repositories in distributed heterogeneous environments.

XML Metadata Interchage (XMI)
A specification enabling easy interchange of metadata between modeling tools (based on the OMG-UML) and metadata repositories (OMG-MOF based) in distributed heterogeneous environments.

XMI Production of XML Schema
An XML schema provides a means by which an XML processor can validate the syntax and some of the semantics of an XML document. This specification provides rules by which a schema can be generated for any valid XMI-transmissible MOF-based metamodel.

Standards and specifications for identifiying and mapping interoperability (OMG, 2002)

Common Object Request Broker Architecture (CORBA/IIOP)
Specification of an architecture for middleware technology called an Object Request Broker that provides interopreability among clients and

servers distributed over a heterogeneous environment.

Authorization Token Layer Acquisition Service (ATLAS)
Describes the service needed to acquire authorization tokens to access a target system using the CSIv2 protocol. This design defines a single interface with which a client acquires an authorization token. This token may be pushed, using the CSIv2 protocol in order to gain access to a CORBA invocation on the target. This specification solves the problem of acquiring the privileges needed for a client to acquire a set of privileges the target will understand.

CORBA Component Model
Specification of: a Component Implementation Definition Language (CIDL); the semantics of the CORBA Components Model (CCM); a Component Implementation Framework (CIF), which defines the programming model for constructing component implementations; a container programming model describing how an Enterprise JavaBeans (EJB) component can be used by CORBA clients, including CORBA componentsl an architecture of the component container as seen by the container provider; how Component implementations may be packaged and deployed; and definitions of the XML DTDs used by the CORBA Components.

Messaging
This specification provides generalized APIs through which such qualities as the ability to set the required and supported qualities of service with respect to requests are set in clients and servers. In addition, the set of Messaging-related qualities and the rules for reconciling and using these qualities are defined. Finally, the Messaging-specific IOR Profile Component and Service Context are defined for propagation of QoS information.

Object Reference Template
Provides interfaces that support the operations that are implemented by the ORB core, and describes some basic ones, while providing reference to the description of the remaining operations that are described elsewhere.

Enterprise Level Analysis and Design Modeling (OMG, 2002)

UML Profile for Enterprise Application Integration (EAI)
Provides a metadata interchange standard for information about accessing application interfaces. The goal is to simplify application integration by standardizing application metadata for invoking and translating application information.

Business Process and Workflow

There may often be a need to identify and link workflow and software functions within the organization. Reorganization of business processes is often necessary. (Sumner, 2000) This can be accomplished through the use of sophisticated messaging and Web-based technological mechanisms. Tools oriented towards assisting in the challenges system integrators can be used to identify and integrate data. As always, this integration should result in the delivery of system functionality to users by virtue of merging multiple cross-functional, multi-platform interfaces into a coherent suite of functionality. The resulting implementation of this functionality should be transparent to the system users.

6. Conclusion

Most curricula only include a small percentage of the above concepts, scattered over topics and other assorted courses. While it is commendable that the topics are addressed in some way, it is almost imperative that today's students learn to build upon the work of others. We are leaving the era of computing in which each organization could build self-styled infrastructures from scratch. It is still necessary that the organization build its uniqueness by fine-tuning the IT portfolio, but this will be done by purchasing large chunks of that portfolio from IT vendors. These will need to be integrated into the overall organization IT infrastructure. A holistic treatment of the issues involved in this activity warrants a complete and coherent treatment within the IT curriculum of the 21st Century.

7. References

Britton, Chris, 2000, IT Architectures and Middleware: Strategies for Building Large, Integrated Systems, Addison-Wesley Pub Co., Princeton.

Casati, F., Discenza, A., "*Modeling and Managing Interactions among Business Processes"*, Journal of Systems Integration, April 2001, Volume 10, Number 2, pp. 145-168.

Cummins, Fred, 2002, Enterprise Integration: An Architecture for Enterprise Application and Systems Integration, John Wiley & Sons, New York.

Goodyear, Mark (ed.), 1999, Enterprise System Architectures: Building Client Server and Web Based Systems, CRC Press, Boca Raton.

Kobryn, C. "*UML 2001: A Standardization Odyssey*" Communications of the ACM, vol. 42, no. 10, Oct. 1999.

Li, H., and Su, S.Y.W, "*Business Object Modeling, Validation, and Mediation for Integrating Heterogeneous Application Systems*", Journal of Systems Integration, Vol. 10, 307-328, 2001.

Maluf, D.A., Tran, P.B. *,"Articulation management for intelligent integration of information"*, Systems, Man and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, Volume: 31 Issue: 4 , Nov. 2001, pp. 485 -496.

Object Management Group (OMG), 2002, http://www.omg.org/technology/documents/modeling_spec_catalog.htm.

Object Management Group (OMG), 2002, http://www.omg.org/technology/documents/corba_spec_catalog.htm#CORBA_IIOP.

Object Management Group (OMG), 2002, http://www.omg.org/gettingstarted/specintro.htm#AnalDesSpecs.

Object Management Group (OMG), 2002, http://www.omg.org/technology/documents/modeling_spec_catalog.htm#CWM.

Omicini,A., Papadopoulos, G., "*Coordination as a Paradigm for Systems Integration",* Journal of Systems Integration, April 2001, Volume 10, Number 2, pp. 103-105

Sumner, M. "*Risk factors in enterprise wide information management systems projects*", Proceedings of the 2000 ACM SIGCPR Conference on Computer Personnel Research, April 2000