

A COMMON FRAMEWORK FOR COMPARING DIFFERENT EAI APPROACHES

Adra Al-Mosawi, College of Information Technology, Department of Information Systems,
University of Bahrain, Bahrain, amosawi@itc.uob.bh

Sofiane Sahraoui, Bahrain Institute of Public Administration, Bahrain,
s.sahraoui@bipa.gov.bh

Abstract

This paper attempts to review different EAI approaches and integrate them into a unified framework enabling their comparison. Approaches that have been developed in the literature are first reviewed and organized into three categories based on integration level; architectural approach; and technological artifacts involved. A common framework is subsequently developed by synthesizing and combining the three categories into a unique classification system that allows for each EAI approach to be catalogued. The proposed framework outlines enterprise application integration through a three-layer architecture (Business Architecture Layer, Application Architecture Layer and, Technology Architecture Layer) with different types of integration within each layer. The framework provides an analytical capability to organize different EAI approaches and assess their suitability to different integration requirements.

Keywords: EAI, Integration, Framework, Approach

1 INTRODUCTION

In the mid 1990s, specialized software tools termed Enterprise Application Integration (EAI) solutions were developed to support the integration of an enterprise's applications. The EAI solution addresses integration problems more effectively by developing a central integration infrastructure. The integration infrastructure manages routing, mapping, and tracking of messages between applications (Linthicum, 1999; Ruh *et al.*, 2000). This architecture reduces the complexity of application integration from $n*(n-1)/2$ to a simpler linear problem. As a result the creation, maintenance and changing of integration logic can be managed in a more flexible and efficient way since changes are limited and interconnections are fewer in number. Moreover, EAI solutions have a non-invasive nature, requiring no or limited changes to applications because exchanges between applications are routed through the central integration infrastructure which translates and reformats them such that they are recognised by the target application. The EAI solutions also provide the infrastructure and mechanisms for extending and integrating both old and new application systems (Zahavi, 1999). EAI further result in the reduction of overall integration cost due to a decrease of both integration time and maintenance costs (Puschmann and Alt, 2001).

Yet, EAI solutions vary in their underlying approaches and adopt different terminology and concepts. A plethora of integration approaches are presented in the literature to describe application integration, thus making the selection of an EAI solution difficult.

2 ENTERPRISE APPLICATION INTEGRATION APPROACHES

Analysts have proposed various approaches to describe the area of application integration (Duke *et al.* 1999; Ring & Ward-Dutton, 1999; Ruh *et al.*, 2000) resulting in a plethora of integration architectures, models, approaches, services, levels and layers in the existing literature. These approaches can be grouped into three categories. The first regroups integration approaches based on the level of integration they address; the second category is based on an architectural approach to integration; and the third includes approaches that deal with integration as a technological artifact. The different categories with supporting references are briefly reviewed below.

2.1 Approaches based on integration level:

This category of approaches is best illustrated by Linthicum (1999) who proposed four integration levels:

- *Data Level EAI* is for moving data between data stores. This level extracts information from one database and updates it in another database.
- *Application Interface Level EAI* is for leveraging of interfaces exposed by custom or packaged applications to gain access to application services.
- *Method Level EAI* is for sharing application logic or methods. This level is accomplished either by defining methods that can be shared and hosting them on a central server, or by rebuilding each application using a method sharing mechanism such as distributed object technologies.
- *User Interface Level EAI* is for integrating applications by using their user interfaces as a common point of integration. This approach is more suitable for custom application integration.

Samtani *et al.* (2002) use similar concepts as Linthicum to report four EAI levels; *User Interface Integration*; *Data Integration*; *Function or Method Integration*; and *Business Method Integration* which occurs at the business process level. Others have suggested similar approaches to Linthicum and Samtani *et al.* (Ruh *et al.*, 2000; Vernadat, 1996; Themistocleous & Irani, 2006; Puschmann and Alt's, 2001; Ring & Ward-Dutton, 1999). The difference between the various approaches lies in the denomination of the different levels and the number of levels considered. For instance, Puschmann and Alt's and Ring and Ward-Dutton each included an object level integration that Linthicum and Samtani *et al.* did not include.

2.2 Architectural approaches to integration

In contrast to the level-based approaches, Duke *et al.* (1999) partition the integration problem into different architectural layers:

- *Business architecture layer* defines the organizational structure and the workflow.
- *Application architecture layer* defines the actual implementation of business concepts.
- *Technological architecture layer* defines the communication infrastructure.

Losavio *et al.* (2002) refer to the same architectural layers as Duke *et al.* using the terms *Process layer*, *Service layer* and *Mechanism layer*. Brown (2000) also refers to the same architectural layer as Duke *et al.*, but splits their business architectural layer into business process and information architectural layers. Brown indicates the need for information architecture to define the way the organization's data is structured for accessibility, navigability and comprehension. Hasselbring (2000) expands the basic architecture by adding one horizontal layer for each vertical. Hasselbring's

expanded architecture has three additional layers: *Inter-organizational Process layer*, *Enterprise Application Integration layer*, and *Middleware Integration layer*.

2.3 Integration as a Technological Artifact

Themistocleous and Irani (2003), Erasala *et al.* (2003), Duke *et al.* (1999), Klasell and Dudgeon (1998), Puschmann and Alt (2001) and Edwards and Newing (2000) tackle application integration as a technological artifact. Themistocleous and Irani (2003) propose the following technological layers of integration:

- *Connectivity layer*, responsible for connecting or linking applications.
- *Transportation layer*, responsible for transferring information from source applications to the integration infrastructure and from the integration infrastructure to target applications.
- *Transformation layer*, responsible for translating and reformatting source application formats into valid formats for the target application.
- *Process automation layer*, integrates business processes and controls the integration mechanism.

Likewise, Duke *et al.* (1999) use the same approach to integration using five layers instead of four. They suggest that a solution based on EAI should involve, in addition to transportation and transformation, timing and sequencing rules that govern when transportation and transformation take place and should also include the integrity constraints that determine the success or failure of the integration.

A raw listing of the different EAI approaches across categories and with supporting referenced in the literature is shown in Table 2-1.

Integration Approaches		Author
Data Integration	Data Level EAI	Linthicum, 1999
	Data Level Integration	Ring & Ward-Dutton, 1999
	Data Integration Model	Ruh <i>et al.</i> , 2000
	Data Integration	Samtani <i>et al.</i> , 2002; Puschmann and Alt, 2001
	Data Warehouse Integration Approach	Grimson <i>et al.</i> , 2000
Function Integration	Method Level EAI	Linthicum, 1999
	Functional Integration Model	Ruh <i>et al.</i> , 2000
	Function or Method Integration	Samtani <i>et al.</i> , 2002
Object Integration	Object Integration	Puschmann and Alt, 2001
	Object Level Integration	Ring & Ward-Dutton, 1999
User Interface Integration	User Interface Level EAI	Linthicum, 1999
	Presentation Integration Model	Ruh <i>et al.</i> , 2000
	User Interface Integration	Samtani <i>et al.</i> , 2002
Application Interface Integration	Application Interface Level EAI	Linthicum, 1999
Process Integration	Process Integration	Puschmann and Alt, 2001
	Internal Process Level	Ring & Ward-Dutton, 1999
	Business Method Integration	Samtani <i>et al.</i> , 2002
	Functional Integration Model	Ruh <i>et al.</i> , 2000
	Cross-enterprise Process Level	Ring & Ward-Dutton, 1999
Business Layer	Business Architecture Layer	Duke <i>et al.</i> , 1999; Hasselbring, 2000
	Business Process Architecture Layer	Brown, 2000
	Information Architecture Layer	Brown, 2000
	Inter organizational Process Layer	Hasselbring, 2000

Application Layer	Application Architecture Layer	Duke <i>et al.</i> , 1999; Hasselbring, 2000; Brown, 2000
	Enterprise Application Integration Layer	Hasselbring, 2000
Technology Layer	Technology Architecture Layer	Duke <i>et al.</i> , 1999; Hasselbring, 2000; Brown, 2000
	Middleware Integration Layer	Hasselbring, 2000
Process Automation Layer	Process Management Service	Puschmann and Alt, 2001; Erasala <i>et al.</i> , 2003
	Process Automation	Themistocleous and Irani, 2003
	Process Automation Layer	Klasell and Dudgeon, 1998
	Business Logic Layer	Edwards and Newing, 2000
	Time and Process Layer	Duke <i>et al.</i> , 1999
Transformation Layer	Transformation Service	Puschmann and Alt, 2001
	Translation Layer	Themistocleous and Irani, 2003
	Transformation Layer	Duke <i>et al.</i> , 1999
	Message Brokering and Translation Layer	Klasell and Dudgeon, 1998
	Integration Service Layer	Edwards and Newing, 2000
	Transformation Services and Distribution Services	Erasala, <i>et al.</i> , 2003
Connectivity Layer	Connectivity Services	Puschmann and Alt, 2001
	Connectivity Layer	Themistocleous and Irani, 2003
Transportation Layer	Transport Layer	Duke <i>et al.</i> , 1999; Klasell and Dudgeon, 1998
	Message and Transportation Layer	Edwards and Newing, 2000
	Communication Services	Erasala, <i>et al.</i> , 2003

Table 1. Integration Approaches

To provide a unified way of comprehending the different EAI approaches, the next section proposes a common framework for comparing different EAI approaches.

3 A COMMON FRAMEWORK FOR COMPARING DIFFERENT EAI APPROACHES

The drivers behind integration include the business need to improve enterprise processes and the subsequent need to integrate technically different applications (Al Mosawi *et al.*, 2006). This means that EAI is a complex task involving both business and technical challenges. From the business viewpoint, EAI involves integration of disparate business processes and functions. From the technical viewpoint, EAI involves integration of incompatible and heterogeneous technologies. In order to manage the complexity of EAI, an architectural approach is necessary. Many researchers (Duke *et al.*, 1999; Hasselbring, 2000; Brown, 2000) believe that an architectural approach is the solution to the integration and interoperability challenges that organizations are facing today. An architectural approach stresses the planning and integration of all information systems with organizational structures and processes, and can be used as an umbrella for guiding and supporting integration activities (Ring & Ward-Dutton, 1999). As such, this paper proposes a framework that considers three architectural layers for EAI: Business Process Architecture Layer, (b) Application Architecture Layer and, (c) Technology Architecture Layer (see Figure 1). The separation of the integration aspects into these three layers is crucial to solve integration problems because EAI is both a business and a technical problem (Al Mosawi *et al.*, 2006). The three architectural layers are discussed below.

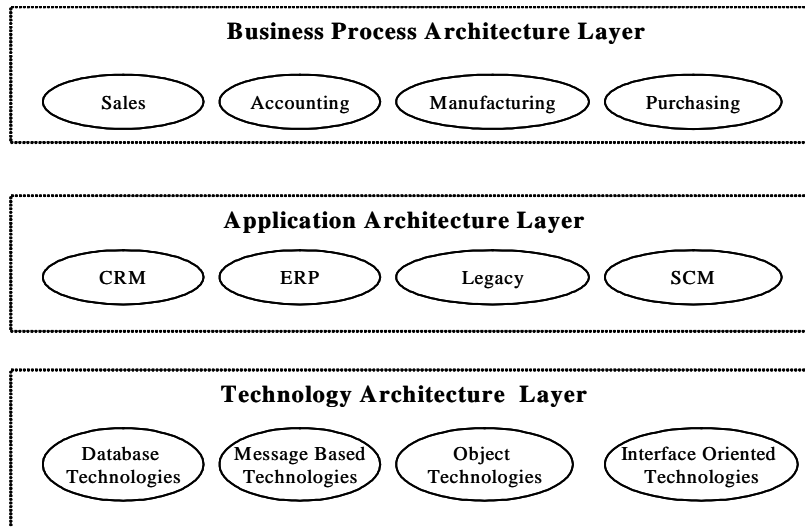


Figure 1. EAI architectural layers

Business Process Architecture Layer: this layer defines the business models underlying enterprise processes which are necessary to achieve the business objectives. It defines the process tasks that must be carried out to conduct business in a particular domain. In Figure 3-1, domain units such as sales, purchasing, manufacturing and accounting are structured in the top layer. Each domain has its own process tasks, rules and constraints.

Application Architecture Layer: this layer defines the application services needed to deliver the required automation of business processes. These services support the management of functionalities and the operations of business processes. The services are presented in terms of enterprise applications such as CRM, ERP, legacy systems, and SCM.

Technology Architecture Layer: this layer defines the technical infrastructure and specifies the integration technologies, namely what middleware products the organization will use. The integration technologies include database oriented technologies, message based technologies, interface oriented technologies and distributed object technologies.

EAI consists of multiple integrations within the three architectural layers, hence a number of different configurations across the layers. Each unique configuration represents an EAI approach. Hence, an EAI approach is a specific combination of different integrations within one or more of the architectural layers.

Figure 1 provides an overview of the different integration layers for EAI. However, it is important to recognize that Figure 1 does not show how integration is achieved within each layer, nor does it reflect which elements are involved in the integration (data, object, or process). As per table 1, the integration of enterprise applications can involve data, object, or process, depending on the integration requirements (Linthicum, 1999; Ruh *et al.*, 2000; Samtani *et al.*, 2002; Ring & Ward-Dutton, 1999). This amounts to a number of integration types depending on which elements are combined. Different integration types have their advantages and could be justified for providing the best solution to an integration requirement. A mapping of the different integration types onto each architectural layer is thought to yield different configurations of alternative EAI approaches.

Figure 2 introduces an expanded EAI framework. The framework builds upon the common EAI architecture (Figure 1) by explicitly specifying the possible integration types. The three architectural

layers, the integration types within each layer and the relationships between the layers are discussed below.

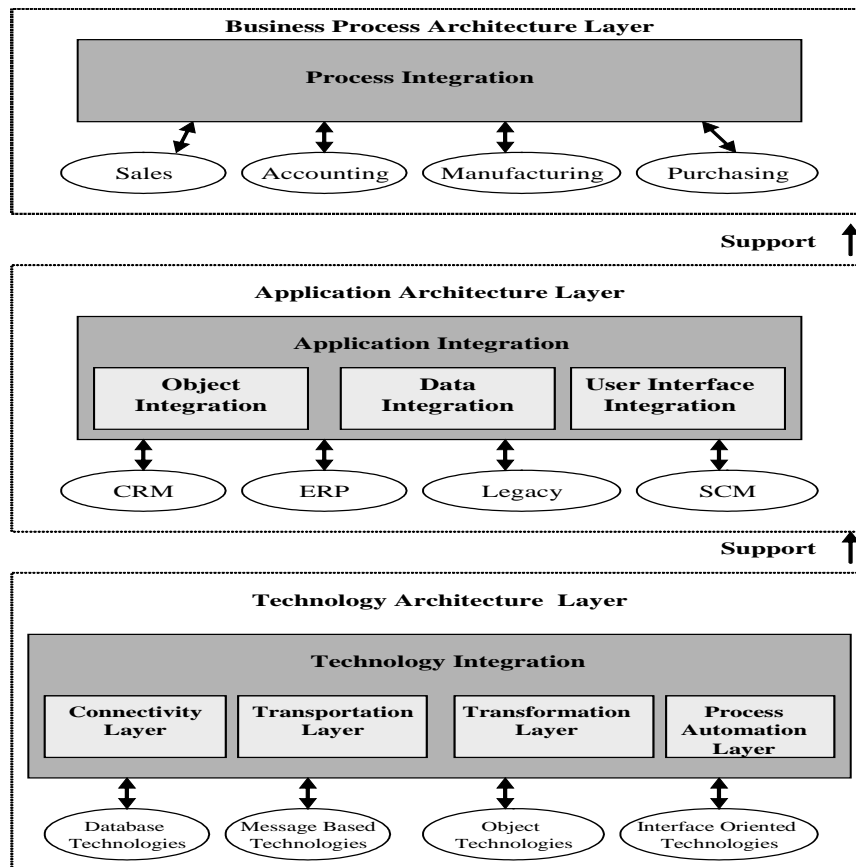


Figure 2. A common framework for comparing different EAI approaches

Business Process Architecture Layer

As stated earlier, the business process architecture encompasses the main processes that the organization wants to operate; the requirements that the processes should fulfil; and the dependencies that exist between the processes. The act of managing these dependencies is called process integration and involves the control structure of the process.

Process Integration: this type of integration defines the hierarchy, sequence, execution logic, events and information movement between different domain units. It also defines the rules that exist within the enterprise which control the flow of information between different domain units. Process integration is, therefore, a strategic solution that leverages business rules to determine how systems should interact (Linthicum, 2001). It aims to achieve integration based on business rules. However, process integration is complex and expensive to implement (Samtani *et al.*, 2002 ; Ring & Ward-Dutton, 1999). It requires a high level of investment and methodological support in which approaches and tools should be applied in a structured way (Ring & Ward-Dutton, 1999). Process integration characteristics are presented in Table 2.

Process Integration	
Strength	Authors
Real-time tracking and analysis of business process	Ring & Ward-Dutton, 1999
More advanced than data and object integration as the logic and reasoning for conducting business are included	Linthicum, 2001
Focus on the actual process and not only on the data	Ring & Ward-Dutton, 1999; Samtani <i>et al.</i> , 2002
Weakness	Authors
Complex to design	Ring & Ward-Dutton, 1999
High initial investment	Ring & Ward-Dutton, 1999; Samtani <i>et al.</i> , 2002
Requires methodological support	Linthicum, 2001; Ring & Ward-Dutton, 1999
Expensive to implement	Samtani <i>et al.</i> , 2002

Table 2. Process integration characteristics

Application Architecture Layer

The application architecture relates to application services and their mutual dependencies in supporting the integration of intra-organizational business processes. The act of managing application dependencies is called application integration and involves the definition of the order of data movement and invocation of application functions. More often, a suite of applications is needed to automate the full set of requirements of the business process. For example a typical business organization would have one procurement process and several applications (exp. order processing, order fulfilment, and invoicing) to fulfil procurement requirements.

Application integration is achieved through integration of object, data, and user interface, or any combination thereof.

- **Object Integration:** this type of integration defines the way information is shared at the function level. This sharing is accomplished by defining the functions that can be shared and the way they are bound or invoked (Ring & Ward-Dutton, 1999). Object integration provides the ability to invoke a function from within one application and have that function executed within another application without having to rewrite each function within the respective application (Linthicum, 2001). It aims to provide the infrastructure for the reuse of application functions. However, this integration is complex and it may be difficult to access the function of some applications because the source code may not be available (Ring & Ward-Dutton, 1999). Further, this integration may require that many enterprise applications be changed or rebuilt to support this integration (Linthicum, 2001). Rebuilding the applications is a very expensive proposition. Object integration characteristics are presented in Table 3.

Object Integration	
Strength	Authors
Integration of business logic and data	Ring & Ward-Dutton, 1999
Reusable	Linthicum, 2001; Ruh <i>et al.</i> , 2000
Weakness	Authors
Difficult if source code is not accessible or no API	Ring & Ward-Dutton, 1999
Extensive changes to existing application as it requires enterprise applications to be changed or rebuilt to support object integration	Linthicum, 2001
Complex	Samtani <i>et al.</i> , 2002

Table 3. Object integration characteristics

- **Data Integration:** this type of integration defines the way data is moved between multiple data sources. This movement is accomplished by defining the data that will be moved from one application to the next and the sequence and frequency of the data movement. Data integration aims to achieve a unified and consistent view of enterprise business data (Samtani *et al.*, 2002). However, this integration assumes bypassing the application logic, therefore limiting real-time transactional capabilities (Samtani *et al.*, 2002). Data integration characteristics are presented in Table 4.

Data Integration	
Strength	Authors
Most common integration approach	Linthicum, 1999
Moves data between multiple data sources and allows data to be exchanged and shared between multiple applications	Ruh <i>et al.</i> , 2000; Samtani <i>et al.</i> , 2002
Provides consistent data	Samtani <i>et al.</i> , 2002
Minimum changes to the source or target applications	Ring & Ward-Dutton, 1999
Weakness	Authors
Does not invoke business logic, limiting real time transactional capabilities	Samtani <i>et al.</i> , 2002
Difficult if database tightly coupled with application logic	Linthicum, 1999

Table 4. Data integration characteristics

- **User Interface Integration:** this type of integration allows the integration of multiple application services through application user interface. It leverages the application's user interface as a point of integration (Linthicum, 1999). User interface integration is particularly useful when there is no well-defined application interface or when original application code cannot be recovered or recompiled (Linthicum, 1999). However, user interface integration cannot be scaled, and thus cannot handle more than a few screen interfaces at a given time (Linthicum, 1999). It only takes place at the presentation and not at the actual inter-connection between applications and data (Linthicum, 1999; Samtani *et al.*, 2002). User interface integration characteristics are presented in Table 5.

User Interface Integration	
Strength	Authors
Easy to implement	Linthicum, 1999
Requires minimum change to existing application	Samtani <i>et al.</i> , 2002
Supports the integration of legacy systems	Robertson, 1997
Useful when there is no well defined application interface or when original application code cannot be recovered or recompiled	Linthicum, 1999
Weakness	Authors
Cannot scale and cannot handle more than a few screen interfaces at a given time	Linthicum, 1999
Faces maintenance problems when screens change	Andrew , 1998
Does not take place at actual interconnection	Linthicum, 1999; Samtani <i>et al.</i> , 2002

Table 5. User Interface integration characteristics

Technology Architecture Layer

Technology architecture relates to the integration technologies and how they work together to create an infrastructure supportive of overall integration. Integration technologies include database oriented technologies, message based technologies, interface oriented technologies and distributed object technologies. They are employed for extracting, translating and transferring application elements (data, object) from a source application to a target application. Integration technologies are thought to operate within different sub-layers of the technology layer (Puschmann and Alt, 2001; Themistocleous and Irani, 2003) depending on the scope and nature of integration sought. These sub-layers are:

- **Connectivity Layer:** this layer is responsible for extracting application elements (data or objects) from the source system. Technologies such as ODBC, JDBC, XML, CORBA, COM/DCOM, EJB, API and Screen Wrapper support this layer (Themistocleous and Irani, 2003).
- **Transportation Layer:** this layer is responsible for transferring the information from source applications to the central integration infrastructure and from the central integration infrastructure to the target application. Technologies such as RPC, MOM, Message Broker, XML, TP Monitors, Application Servers, CORBA, COM/DCOM and EJB support this layer (Themistocleous and Irani, 2003).
- **Transformation Layer:** this layer is responsible for translating and reformatting the information from source application format into a valid format for the target application. Technologies such as Message Broker, Adapters, XML and CORBA support this layer (Themistocleous and Irani, 2003).
- **Process Automation Layer:** this layer is responsible for rule processing, distribution and routing the request or the information to the appropriate application at the appropriate time. It controls the execution of a sequence of transportation and transformation tasks. Technologies such as message broker support this layer (Themistocleous and Irani, 2003).

As noted above, there are several types of integration within each of the three architectural layers. However, there are also integration relationships that span the layers to make up specific EAI approaches. In this three-layered architecture, each lower layer offers a particular type of services to support the layer above. For example, the business process architecture layer constitutes a set of process tasks that are automated or executed by combining or selecting application services from the application architecture layer. The application architecture layer thus describes the enterprise application services required to support the business architecture layer. Likewise, the technology architecture layer describes the integration infrastructure environment required to support the application architecture layer. This integration infrastructure is implemented through several middleware technologies from the technology architecture layer.

4 CONCLUSIONS

This paper has developed a common framework for a systematic and unified understanding of different EAI approaches, hence alleviating the confusion surrounding EAI approaches and concepts and sorting through the plethora of integration approaches, layers, levels, models, architectures that exist in the EAI literature.

The proposed framework outlines enterprise application integration through a three-layer architecture (Business Architecture Layer, Application Architecture Layer and, Technology Architecture Layer) with different types of integration within each layer. The framework provides an analytical capability to organize different EAI approaches and assess their suitability to different integration requirements.

Though providing an organizing concept for different EAI approaches, the framework falls short of prescribing normative mechanisms for configuring different types of integrations into coherent EAI approaches. Hence, it remains descriptive at best by delineating the integration types and layers making up an EAI approach. While this could be construed as a weakness in the framework, the development of a 'theory of EAI development' is well beyond the scope of this paper.

References

- Al Mosawi A, Zhao L and Macaulay L. 2006. "Model Driven Architecture for Enterprise Application Integration". Hawaii International Conference on System Sciences, HICSS-39, January 4-7.
- Andrew D. 1998. "Butler Report: Component Based Development". Butler Consulting Group Limited, Yorkshire, UK.
- Brown L. 2000. *Integration Models: Templates for Business Transformation*. SAMS Publishing, USA.
- Duke S, Makey P and Kiras N. 1999. *Application Integration Management Guide: Strategies and Technologies*. Butler Group Limited, Hull, UK.
- Edwards P and Newing R. 2000. *Application Integration for e-Business*. Business Intelligent, London, UK.
- Erasala N, Yen D.C. and Rajkumar T.M. 2003. "Enterprise Application Integration in Electronic Commerce World". *Computer Standards & Interfaces*, 25(2):69-82.
- Grimson J, Grimson W and Hasselbring W. 2000. "The SI Challenge in the Health Care". *Communications of the ACM*, 43 (6): 49-54.
- Hasselbring W. 2000. "Information System Integration". *Communications of the ACM*, 43 (6):33-38.
- Klasell T and Dudgeon S. 1998. *Enterprise Application Integration*. Dain Rauscher Wessels, New York, USA.
- Linthicum D. 1999. *Enterprise Application Integration*. Addison-Wesley, Massachusetts, USA.
- Linthicum D. 2001. *B2B Application Integration*. Addison-Wesley, Massachusetts, USA.
- Losavio F, Ortega D and Perez M. 2002. "Modelling EAI". *Proceedings 22nd International Conference of the Chilean Computer Science Society*, 195-203.
- Puschmann T and Alt R. 2001. "Enterprise Application integration: the case of the Robert Bosch Group". *Proc. 34th Hawaii International Conference on System Sciences (HICSS'01)*, Hawaii, USA.
- Ring K and Ward-Dutton N. 1999. *Enterprise Application Integration: Making the Right Connections*. Ovum Ltd, London, UK.
- Robertson P. 1997. "Integrating Legacy Systems with Modern Corporate Applications". *Communication of the ACM*, 40 (5): 39-46.
- Ruh W, Maginnis F and Brown W. 2000. *Enterprise Application Integration: A Wiley Tech Brie*. John Wiley & Sons Inc., New York, USA.
- Samtani G, Healey M and Samtani S. 2002. *B2B Integration: A Practical Guide to Collaborative E-commerce*. Imperial College Press, London, UK.
- Themistocleous M and Irani Z. 2003. "Towards a Novel Framework for assessment of Enterprise Application Integration Packages". *Proc. 36th Hawaii International Conference on System Sciences (HICSS'03)*, Hawaii, USA.
- Themistocleous M and Irani Z. 2006. "Towards a Methodology for the Development of Integrated IT Infrastructure". *Hawaii International Conference on System Sciences, HICSS-39*.
- Vernadat F.B. 1996. *Enterprise Modelling and Integration: Principles and Applications*. Chapman & Hall, London.
- Zahavi R. 1999. *Enterprise Application Integration with CORBA*, John Wiley and Sons Inc, New York, USA.