

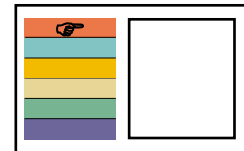
COMPONENT-BASED FRAMEWORKS FOR E-COMMERCE

Forward-thinking companies have come to realize that e-commerce is neither just a buy-side nor sell-side packaged application. They have learned that mission-critical business opportunities abound and they must implement many e-commerce initiatives along the way to building the virtual corporation that thrives in the Digital Economy. To them, e-commerce is an infrastructure for a whole new way of doing business. They have learned if they extend their business processes across company boundaries and integrate them with their suppliers' and customers' business processes something totally new starts to happen.

E-commerce applications can be categorized into the four major groupings shown in Figure 1. On the sell side of most companies, I-Market applications include online catalog management, order management, trading communities, marketing and advertising, while Customer Care applications involve customer self-service, customer relationship management and business intelligence support. On the buy side, Vendor Management Systems automate the procurement of indirect

operating resources including sourcing, bid/ask buying and custom supplies catalogs. Extended Supply Chain Management includes collaborative forecasting and planning, scheduling and logistics. These are general categories, and some companies such as GE and Home Depot already have identified more than 70 e-commerce opportunities for competitive advantage and business concept innovation.

To achieve coherence and manage the complexity and change inherent in multiple e-commerce applications, an overarching structure is needed—an application architecture. Joined electronically, companies operating in the Digital Economy must share a common foundation for integrating their unique business processes and embrace the software component paradigm as the way forward. Their core business processes are embedded in legacy, enterprise resource planning (ERP) and client/server systems. In order to retarget these internal systems outward, common inter-enterprise application functions are needed. As shown in the inner oval of Figure 1, information boundaries, workflow/process management, trading services, searching



*The software
paradigms of
component-based
frameworks for
e-commerce
promise to
provide companies
with the speed
and agility they
need to compete
in Internet time.*

∞
**PETER
FINGAR**
∞

and information filtering, data/process integration, and event notification form the foundation for refocusing information systems from the inside out. These are the elements or components common to all e-commerce applications, and are essential to integrating a multitude of individual e-commerce applications. Without integration of e-commerce applications, chaos will reign. These new breed applications must not only be integrated with each other, but also with existing ERP and legacy systems.

Component-based E-commerce Architecture

Distributed object computing is now recognized as the way forward in building enterprise information architectures that can operate in advanced client/server, intranet, and Internet environments. In essence, using objects to build information systems is like building a simulation with business objects representing the people, places, things, and events that are found in the business setting or domain. Business objects reflect the real world and thus greatly enhance understanding and communication among systems developers and business people. And business objects reduce complexity because programmers do not need to know how an object works internally. They only need to know what the object is and the services it provides. Object technology, however, does have some downsides including a steep learning curve. Business objects, though they represent things in the real world, become unwieldy when combined and recombined in large-scale commercial applications. What is needed are ensembles of business objects that provide major chunks of application functionality (for example, preprogrammed workflow, transaction processing, and user event notification) that can be snapped together to create complete e-commerce applications.

Figure 1. E-commerce common application functionality.

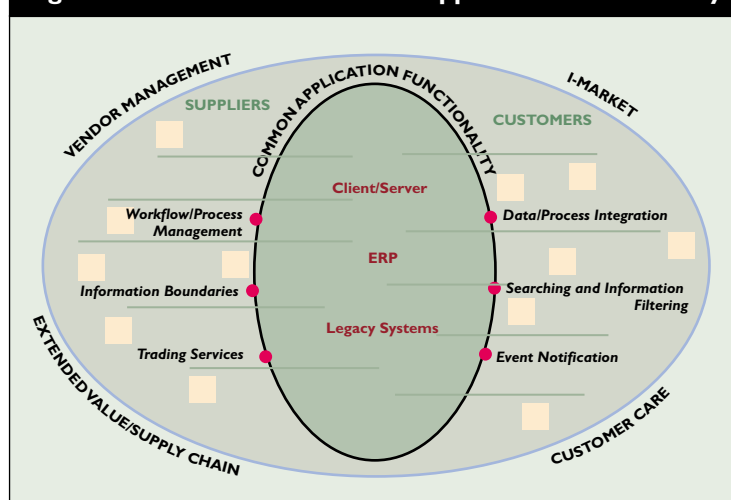
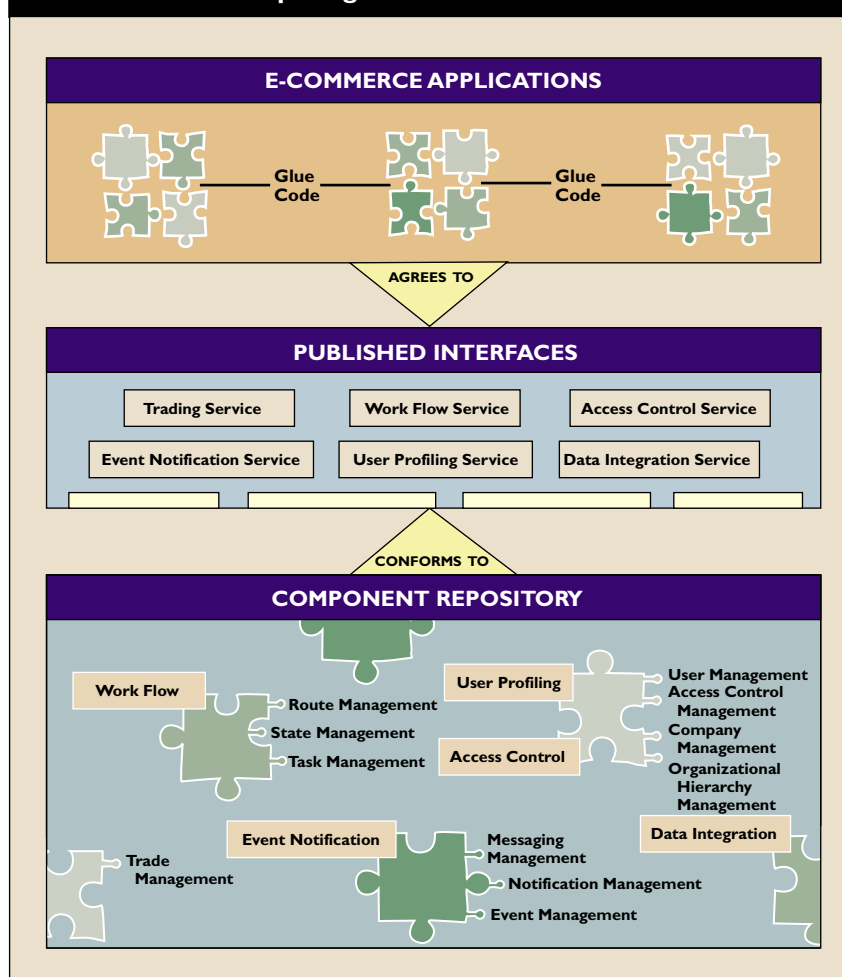
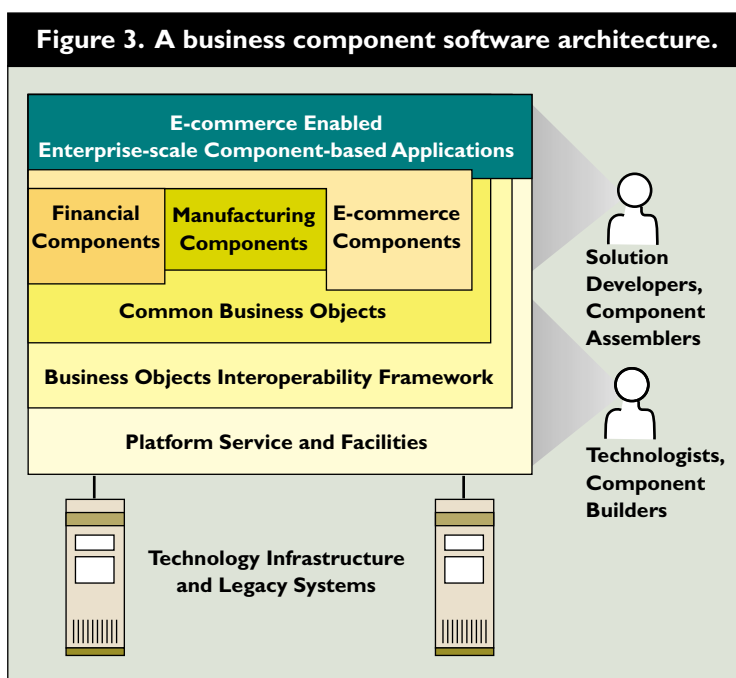


Figure 2. Large-grained e-commerce components exposing standard interfaces.



This approach is embodied in the next step in the evolution beyond objects: software components. Components are self-contained packages of functionality that have clearly defined, open interfaces



that offer “plug-and-play” high-level application services. Components can be distributed dynamically for reuse across multiple applications and heterogeneous computing platforms. The later characteristic is why Java (“write once, run anywhere”) portability has had such a dramatic impact on enterprise computing and component development, and why the Extensible Markup Language (XML) is essential for developing a shared Internet file system.

When implemented in software as components, e-commerce functions appear to application developers as a set of services. These services are part of a layered software architecture. Within each component, services are implemented by a collection of business objects. A service is simply a request protocol for a logical unit of work—for example, update employee’s address, move the task to next person in queue, send the purchase order to the supplier via EDI and so on. Services can be invoked without the requester needing to know the implementation details of the software that delivers them. The process is very much like driving a car without needing to know how the engine works. It is extremely important, however, that component services are provided through a standard, published interface to ensure interoperability, ease of use, and loose coupling. To summarize, an application component is a collection of code that provides one or more services based upon a clearly-defined, standard interface as illustrated in Figure 2.

These brief paragraphs reveal a breakthrough sought since the beginning of software development: true software reuse! The common e-commerce com-

ponents will be used throughout the growing portfolio of e-commerce applications. The architecture is flexible and meant to evolve. Initially, of course, a company has no components. As it proceeds to build its next generation systems it will make or, more likely, buy the components that will populate a growing repository. Over time the architecture will accommodate robust and adaptive information systems that embrace rather than cringe at business change. Its overarching structure will allow companies to grow the software they need to compete for the future.

E-service

E-commerce application components will not be delivered to corporations as a big pile of parts and pieces. Instead the components will be preassembled into industry specific application frameworks as illus-

trated in Figure 3. These frameworks represent applications that are almost, say 60%–80%, complete.

Software component architectures are service-based: end-user services, business process services, and data services. Application components rely on distributed computing infrastructure services, freeing solution developers from the complexity and intricacies of the underlying technologies. Component builders are technologists who use component-based software engineering disciplines to produce components of extreme quality. Solution developers consume these prefabricated components during business process modeling and rapid application assembly.

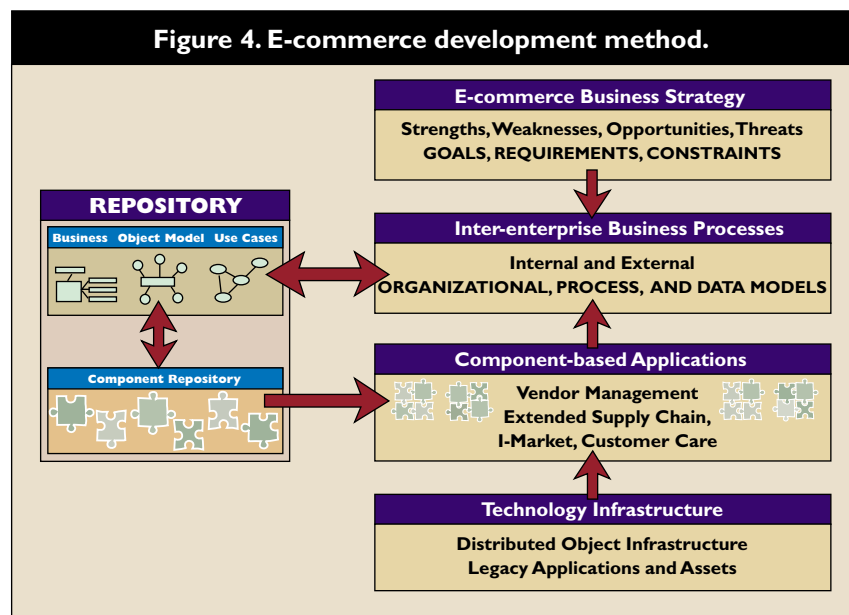
Component architectures divide software into construction and consumption: once built in compliance with standards, newly constructed software components can register the services they provide, while other components can subscribe to and consume these services. Components do not act alone, they plug into component frameworks that connect participating components and enforce rules of component interaction. Component frameworks mediate and regulate component interaction. Component frameworks are arranged in a tiered architecture. Figure 3 illustrates components and how they plug into an interoperability framework that in turn calls on the services and facilities of a distributed computing platform.

The e-commerce components provide essential inter-enterprise functionality for e-commerce (negotiation, mediation, inter-enterprise user access management, inter-enterprise workflow management and event notification). The task of solutions devel-

opers is to customize and extend the frameworks to incorporate the unique business rules and processes of a company. Thus, solution developers are insulated from the technology plumbing and can concentrate on the unique character and knowledge of the company that provides its competitive advantage.

The notion of business-centricity is formalized in Peter Herzum and Oliver Sims' business component architecture, a hierarchy of software components that are isophomoric with business concepts—the software is architected with tight mappings to the ways business people speak and define their world of business organizations, policies, and processes [1]. From elementary “distributed components” to “federations” of business component systems, each higher-level construct encapsulates the lower-level components and business objects. A distributed component (DC) is a design pattern for an autonomous software artifact that can be deployed as a pluggable binary component into a runtime environment with a network accessible interface, always passed by reference. Moving up the hierarchy, a business component (BC) represents and implements an autonomous business concept such as customer by encapsulating all its software artifacts, in effect it provides an interface to a whole collection of DCs necessary to realize a single business concept. Next, a Business Component System (BCS) combines multiple BCs and provides a common interface that renders complete business processes such as invoice management, while Federated Component Systems (FCS) provide the interfaces for BCSs to interconnect with other BCSs across the Internet and perhaps belonging to different organizations—the essence of e-commerce applications.

With e-commerce frameworks built from a hierarchy of business components, business people and software developers are freed to work with high-level business constructs that can be reified in software. The Herzum and Sims' approach strives to ensure that business notions and semantics remain the basic mental models of the entire software development process. Advanced e-commerce applications are composed of business services (popularly called e-services) of suppliers and trading partners. They do not run on a single computer, instead, the network



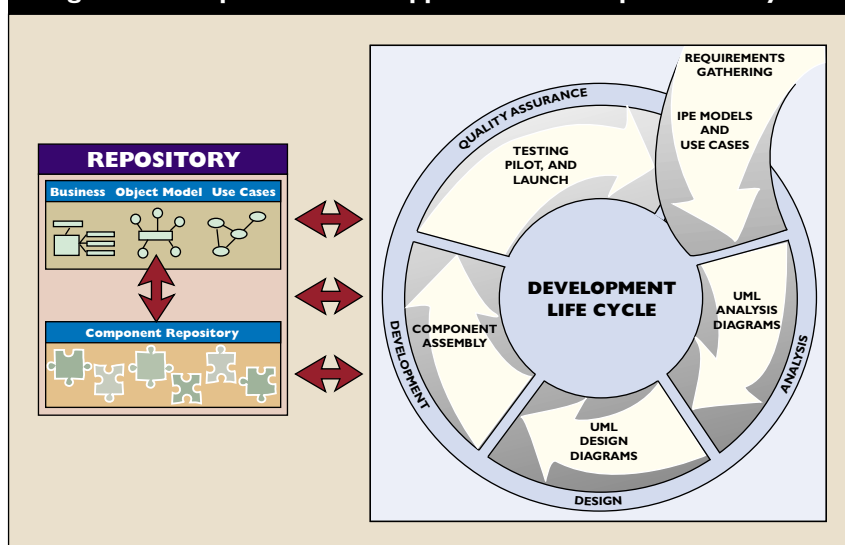
becomes the computer shared by all participants. Herzum and Sims' federations are a natural for designing and implementing external e-services. Finer-grained components suffer the same limitations as the original notion of business objects while federated business component systems can be used to build the e-commerce frameworks and a software factory capable of implementing change at Internet speed. What's more, as companies discover that all their information systems should be e-commerce systems, these frameworks can provide the coherent architecture needed to migrate to such a future.

Standards for Interoperability

The telephone was the first universal, fully interactive information highway. Standards made it possible for any phone company's once-proprietary systems to interoperate with any other phone company's systems. As a result, today anyone can phone anyone anywhere at anytime. Without global standards, however, the telephone would not be universal and would not have its impact on society and the economy.

As we move forward to the Digital Economy, e-commerce transforms from closed, isolated markets to dynamic open markets. Standards are prerequisite to such dynamic markets and market-to-market interoperation. What's new in standards? XML is being touted as the silver bullet for interoperability and indeed it has great potential. But already XML has led to the Tower of Babel problem due to the growing number of XML industry vocabularies—each industry defines a closed standard unique to that industry. Open standards between and among these semantic towers of e-commerce Babel are essential to digital commerce. For example, the def-

Figure 5. Component-based application development life cycle.



initiation of a customer in the real estate industry should be compatible with the definition of customer in the automobile industry—not so in the first wave of industry-specific XML standards. Standards to watch for open e-commerce (tying together the XML Towers of Babel) include CommerceNet's eCo system architecture and the Object Management Group's Electronic Commerce Reference Architecture Model.

Fusing Business and Technology

Competing for the future with e-commerce is not just about technology. It is not just about business. It is inseparably about both, and architecture is the key to fusing the two. In his book *Systems Architecting*, Eberhardt Rechtin paints the picture, "Both architects and business managers live in ill-structured, unbounded worlds where analytic rationality is insufficient and optimum solutions are rare. Both have perspectives that are strategic and top-down. Top managers, like chief architects, must architect strategies that will handle the unforeseeable, avoid disaster and produce results satisfactory to multiple clients—to boards of directors, customers, employees, and the general public. Their common modus operandi is one of fit, balance and compromise in the overall interest of the system and its purposes." [2]

The key components of a business architecture are organization, processes, and technology. The arrangement and connections of these components make up the business architecture. When a business is simple and small, its architecture is implicit. Without explicit business architecture, however, large complex organizations would move rapidly to a state of chaos. The profound and discontinuous change

that is being wrought by the emergence of e-commerce demands a fresh approach to business design.

No longer can a company sit comfortably in its niche in its industry and carry out business planning by extrapolating yesterday's assumptions. And it cannot plan alone since forging new electronic relationships with suppliers, distributors, partners, and customers requires direct participation by these stakeholders. A new approach is needed to design a sustainable, changeable business architecture and to build the essential components that must cross com-

pany and industry boundaries, forming a new business ecosystem.

An effective approach to business architecture must include an integrated and seamless method of business engineering and software development. What is needed are end-to-end, strategy-to-code methods for designing and deploying e-commerce initiatives as illustrated in Figure 4. Business strategy is not strategy-as-usual when it comes to e-commerce. Working directly with the stakeholders inside and outside the business, a shared vision must be developed that allows the players to invent their shared future. The business strategy phase of e-commerce development identifies "what" to do. The business process engineering phase addresses the "how." The task at hand is the mapping and engineering of the inter-enterprise business processes. Work activities, steps and hand-offs are redesigned using the methods and tools of business process reengineering, only this time the mappings are inter-enterprise. Inter-organizational processes may be loosely coupled as in the case of open markets where pass-throughs are made from the I-Market to individual organizations. Or they may be tightly coupled as in the case of a supply chain where process hand-offs are made electronically.

Process engineering is a model-based approach to problem-solving. As functional requirements are passed through to the process engineering phase, a repository of previously developed models can be searched for processes that match the requirements. A shared repository of the artifacts produced by business and systems modeling serves as a reference architecture. The repository also contains a standard representation of requirements use cases. Use cases

not only serve as a way of capturing the requirements of a system, they also trigger later development steps from analysis through design, implementation and testing. Each step provides further elaboration of previous steps. The use cases bind the steps together providing traceability and thus permit the management of inevitable change. Use cases are a part of the Unified Modeling Language (UML), which has become the standard for modeling business systems. UML is designed to model components and guide in their construction, assembly and reuse.

New and existing components are held in the component repository for reuse. Figure 5 shows the software development life cycle for an e-commerce application.

In summary, a strategy-to-code method of developing e-commerce systems fuses business architecture with technology architecture. Business strategy involves domain experts who define the initiatives to be pursued based on an analysis of the company's strengths, weaknesses, opportunities and threats. Business strategy determines what problem is to be solved, generating requirements, goals and constraints. Inter-enterprise process engineering defines how the requirements are to be satisfied through new or modified organizations, processes and data shared among the company's partners, suppliers and customers. Component-based applications implement the newly designed business processes, leveraging the technology infrastructure. The entire process is guided by an architectural approach that enables rapid development of business solutions through reuse of all elements in a growing repository of models and software components.

Mission-critical E-commerce

Inter-enterprise process engineering does not result in an end state. It is an on-going process that enables virtual corporations to evolve in a continually changing business ecosystem. Organization form and function must be able to sense and respond to change. Agility is the byword of success—an agile business empowered by agile information systems. No longer can software development be on the critical path for organizational and process change. With change being the constant variable, a new software development paradigm—component-based e-

commerce frameworks—is essential to building agile, virtual corporations.

As e-commerce becomes the preferred way of doing business it becomes mission-critical—by lowering costs or increasing revenues it directly affects the bottom line. Investments in e-commerce require the same ROI analysis, risk assessments and solid management as other strategic investments. Understanding the new critical success factors is essential. New opportunities and threats will become routine

affairs among the players in this brave new world, and thus agility is the critical success factor. When new opportunities are discovered, rapid response is essential. In the world of e-commerce, fast followers can replicate an innovative business model in Internet speed. Thus, the ability to change is now more important than the ability to create e-commerce systems in the first place. Change becomes a first-class design goal and requires business and technology architecture whose components can be added, modified, replaced, and reconfigured. Software solutions must be agile, capable of rapid bundling, unbundling, and rebundling. Component-

based frameworks provide this capability.

Formulating e-commerce strategy should be at the top of management's priorities. Today's successful businesses have developed their unique style, form and ways of doing business. Their legacy assets are the basis of their success and should be leveraged, not obliterated, in the process of developing e-commerce strategy. A firm's competitive advantage is embedded in its unique business processes and its communal knowledge. By leveraging existing mission-critical processes and extending them to its customers, suppliers, and partners, a corporation can build bridges to the Digital Economy. **C**

WITH CHANGE BEING THE
CONSTANT VARIABLE, A
NEW SOFTWARE
DEVELOPMENT PARADIGM—
COMPONENT-BASED
E-COMMERCE
FRAMEWORKS—IS
ESSENTIAL TO BUILDING
AGILE, VIRTUAL
CORPORATIONS.



REFERENCES

1. Herzum, P. and Sims, O. *Business Component Factory: A Comprehensive Overview of Component-Based Development for the Enterprise*. Wiley, NY, 2000.
2. Reichtin, E. *Systems Architecting: Creating and Building Complex Systems*. Prentice Hall, NY, 1991.

PETER FINGAR (pfingar@acm.org) is Senior Vice President of the Noor Group, LLC, based in Washington, DC, and Cairo, Egypt.

© 2000 ACM 0002-0782/00/1000 \$5.00