## *Enterprise Application Requirements*

Like any modern application, an enterprise application must be reliable, perform well, provide an intuitive and efficient user interface, and so on. But beyond these common qualities, it can be characterized by three specific attributes.

An enterprise application is:

**Large**

A multi-user, multi-developer, multi-machine, multi-component application that can manipulate massive data and utilize extensive parallel processing, network distributed resources, and complex logic. It can be deployed across multiple platforms and inter-operate with many other applications, and it is long lived.

**Business Oriented**

Its purpose is to meet specific business requirements. It encodes business policies, processes, rules, and entities, is developed in a business organization, and is deployed in a manner responsive to business needs.

**Mission Critical**

An enterprise application must be robust enough to sustain continuous operation. It must be extremely flexible for scalability and deployment, and allow for efficient maintenance, monitoring, and administration.

These qualities clearly make the task of enterprise development extraordinarily challenging, and the trend is toward rapidly increasing demands. The rapid improvement of computer hardware and software, combined with global economic competition — and opportunities — have created an environment in which business systems must respond quickly and deliver unparalleled levels of performance. As these demands continue, developers must automate even more of their businesses, build their software even faster, serve more and more users, and process a rapidly growing mass of data.

Aside from these challenges, the power, complexity, and rate of change of the technology used in building these corporate solutions makes efficient development ever more difficult.

To design an enterprise application you must consider and balance an enormous array of application requirements, such as:

* Its business goals.
* How soon it must be delivered.
* Its budget.
* How many people will develop, test, and maintain it.
* How many concurrent users it must support.
* The importance of performance and ease of use.
* The hardware it must run on.
* Where it will be deployed.
* What security is required.
* How long you expect to use it.

Without a systematic way to understand the relationships among these complex and often conflicting requirements, it’s hard to know where to begin. A simpler model can help reduce this complexity, and provide an organized way to design and build applications that chart an optimum course among the many requirements. This model is introduced in the section, Enterprise Application Model.

**2.Challenges of Enterprise Application Development**

Timing has always been a critical factor when organizations adopt new technologies, and the accelerated pace of the information-driven business model puts greater emphasis on response times. Organizations need to be able to project enterprise systems into various client channels, and to do so in a way that’s reliable, productive, and capable of sustaining frequent updates to both information and services. The principal issue is how to keep up with today’s business challenges—whatever they may be—while maintaining and leveraging the value of existing information assets. In this environment, timeliness, productivity, security, and predictability are all absolutely critical to building and maintaining momentum. A number of factors can enhance or impede an organization’s ability to deliver custom enterprise applications quickly and to maximize their value over their lifetime.

Programming ProductivityThe ability to develop and deploy applications is key to success in the information  
economy. Applications must go quickly from prototype to production and must continue to evolve even after they are deployed.  
Productivity is thus vital to responsive application development. Providing  
application development teams with standard means to access the services  
required by multitier applications and standard ways to support a variety of clients  
can contribute to both responsiveness and flexibility.  
The current divergence of technologies and programming models is a destabilizing factor in Internet and other distributed computing applications. Traditional  
Web technologies such as HTML and Common Gateway Interface (CGI) have  
provided a mechanism for distributing dynamic content, while back-end systems  
such as transaction processors and database management systems have provided  
controlled access to the data to be presented and manipulated. These technologies  
present a diversity of programming models: some based on well-defined standards; others on more ad-hoc standards; and others still on proprietary architectures.  
With no single application model, it can be difficult for teams to communicate  
application requirements effectively and productively. As a result, architecting  
applications becomes more complex. What’s more, the skill sets required to integrate these technologies aren’t well organized for effective division of labor. For  
example, CGI development requires coders to define both content and layout of a  
dynamic Web page.  
Another complicating factor in application development time is the choice of  
clients. While many applications can be distributed to Web browser clients  
through static or dynamically generated HTML, others may need to support a specific type of client or to support several types of clients simultaneously. The programming model needs to support a variety of client configurations, with  
minimum effect on basic application architecture and on the application’s core  
business logic.

Integration with Existing SystemsMuch of the data of value to organizations has been collected over the years by  
existing enterprise information systems. Much of the programming investment  
resides in applications on those same systems. The challenge for developers of  
enterprise applications is how to reuse and commoditize these existing information  
assets.  
To achieve this goal, application developers need standard ways to access  
middle-tier and back-end services such as database management systems and  
transaction monitors. They also need systems that provide these services consistently, so that new programming models or styles aren’t required as integration  
expands to encompass various systems within an enterprise.

Freedom of ChoiceApplication development responsiveness requires the ability to mix and match solutions to come up with the optimum configuration for the task at hand. Freedom of  
choice in enterprise application development should extend from servers to tools to  
components. The wide range of J2EE compatible solutions available today and in  
the future ensures the maximum freedom of choice.  
The availability of choices among server products gives an organization the  
ability to select configurations tailored to their application requirements. It also  
provides the ability to move quickly and easily from one configuration to another  
as internal and external demand requires.  
Access to the appropriate tools for the job is another important choice. Development teams should be able to adopt new tools as new needs arise, including  
tools from server vendors and third-party tool developers. What’s more, each  
member of a development team should have access to tools that are most appropriate to their skill set and contribution.  
Finally, developers should be able to choose from a ready market of off-theshelf application components to take advantage of external expertise and to  
enhance development productivity

Response to DemandWhen designing large-scale distributed applications, both availability and scalability  
are key considerations. The more easily and automatically that an application can  
handle changes in use patterns and system configurations, the better. Systems that

require any redesign, recoding, or redeployment to achieve either availability or  
scalability will limit flexibility and diminish expected performance.  
To scale effectively, systems need to be designed to handle multiple client  
interactions with ease. They need mechanisms for efficient management of system  
resources and services such as database connections and transactions. For highest  
availability, they need access to features such as automatic load balancing and  
failover, without any effort on the part of the application developer. Applications  
should be able to run on any server configuration appropriate to anticipated client  
volumes and to easily switch configurations when the need arises. Support for  
clustered application deployment environments contributes to achieving many of  
these goals

Maintaining SecurityMore than ever, information systems security is on the minds of IT managers and  
system architects. That’s because protecting information assets to maximize their  
value can jeopardize that very value. Traditionally, IT departments have been able to  
maintain a relatively high level of control over the environment of both servers and  
clients. When information assets are exposed in less-protected environments, it  
becomes increasingly important to maintain tight security over the most sensitive  
assets, while allowing seemingly unencumbered access to others.  
One of the difficulties in integrating disparate systems is providing a unified  
security model. Single sign on across internal application and asset boundaries is  
important to creating a positive user experience with the applications. Security  
needs to be compatible with existing mechanisms. In cases where customers need  
to access secure information, the mechanisms need to maintain high security (and  
user confidence) while remaining as unobtrusive and transparent as possible.