



## REEF INTERNETWARE®

*A New Architecture for Meeting the  
Challenges of Modern E-business*

A TECHNICAL WHITE PAPER  
JULY 2000

# CONTENTS

Introduction	3
Technology Challenge	3
Business Requirements	3
Evolution of Traditional Technologies	4
Internet Application Architecture Requirements	5
Reef Architectural Elements	7
The Reef Engine	13
Three Levels of Customization	15
Licensing and System Requirements	17
Conclusion	18

---

## INTRODUCTION

This technical white paper discusses the new and emerging online challenges of modern e-business and Reef's software-based approach to meeting these challenges.

This paper is intended to provide IT managers and CIOs with an understanding of how Reef Internetware® works and why it represents the best software architecture for the development and deployment of flexible Web sites and online applications.

---

## TECHNOLOGY CHALLENGE

While the network and presentation technologies upon which the Internet was founded have proven to be robust and extensible enough to meet today's Internet requirements, the same cannot be said of the applications and tools which emerged to combine these technologies for content delivery and application development.

Hypertext Mark-up Language (HTML), developed originally for hyperlinked access to static content, was quickly augmented by server-side scripting solutions such as Common Gateway Interface (CGI) for dynamic content management and database access. As powerful and effective as these technologies remain, the rapid rate of change in the Internet community demands solutions that combine these technologies and others to enable dynamic content management and services delivery by ordinary business users.

Applications and tools that evolved to support the relatively specific applications of yesterday's Web no longer scale to support the ubiquity of today's and tomorrow's online services, nor do they support the effective provision of such services by non-technical knowledge workers.

---

## BUSINESS REQUIREMENTS

### *Managing dynamic, decentralized, media-rich content*

The problem with earlier generation technologies is that they support an outdated understanding of how online content and services should be provisioned. Such technologies have not kept pace as online content has changed in the following ways:

- From static to dynamic presentation-Whereas online content used to be static, today it is highly dynamic: subject to rapid, often continuous change.
- From one to multiple sources-Content comes from many different places today, in many cases syndicated from remote information services, multiple databases, multiple directories, and multiple business applications.
- From centralized to decentralized delivery-Mirroring the business organizations themselves, content delivery is being flattened today, with information being published directly by its originators, wherever they happen to be located in the organization or the world.
- From text-only to media-rich formats-Online content is no longer simply formatted text, but increasingly takes the form of rich audio, visual, and textual multimedia.
- From a little content to a lot-There is more content to manage now than ever before, and there will be even more in the future.

Businesses require technologies that support today's-and perhaps more importantly, *tomorrow's*-online demands. Technologies that empower business users to manage online content and applications on their own-without having to funnel every request through an IT intermediary. And technologies that can scale to meet future needs, both in terms of capacity and capabilities.

### *Deploying online applications and e-services-fast*

Moreover, Internet-based business is less and less about solely content and more and more about services, processes, transactions, and other commercial activity. To deliver these capabilities, traditional computing applications of all types are being ported to the Internet, to intranets, and to extranets, with completely new, often highly specialized online applications continuing to appear as well.

Businesses are under competitive pressure to deploy all of these applications rapidly, and expand and update them as business needs evolve. Yet, effective Internet-based business management precludes the luxury of traditional multi-month development cycles.

Businesses also face the pressure to protect existing investments by maintaining strong connections to legacy systems, applications, databases, and other business-critical infrastructure components.

### *Developing and fostering online communities*

Finally, many organizations are realizing that the Internet is as much a platform for the controlled or spontaneous development of online communities as it is for basic information technology. Applications and content can be personalized to serve the unique needs of non-geographic communities, constituencies, and individuals.

In fact, the Internet provides newfound opportunities for highly effective one-to-one marketing and highly satisfying customer service experiences through customized content, personalized access to specific information and applications, and tailored interactions with other members of online communities. To take advantage of such emerging forms of community, Internet development environments must support a rich concept of user management and access control.

---

## EVOLUTION OF TRADITIONAL TECHNOLOGIES

To fully appreciate the challenges of implementing traditional Web technologies today-and the unique advantage Reef offers-it is helpful to understand how such technologies came to be and how they work.

### *CGI*

Common Gateway Interface (CGI) provides a framework for running scripts and other programs in a host machine's operating environment, which can be used to serve content from a file system or database, or to process specific business logic. One of the early attractions of CGI was its freeform flexibility and program language independence, which made it easy for individuals and small teams to build CGI-based applications very rapidly.

Limitations and disadvantages of the CGI scripts or programs approach began emerging, however, as large and mission-critical applications began to be constructed with it. For example, use of the native server operating environment by CGI scripts or programs for processing invites harrowing security liabilities. The unstructured programming flexibility, while beneficial for the rapid deployment of smaller applications, renders larger-scaled development and especially maintenance costly, complex, and time-consuming. Also, the native file systems often used by CGI scripts or programs can rapidly become unwieldy and unsupported in dynamic, fast-changing environments.

## *ASP and CFML*

In an effort to address some of the limitations of CGI scripts or programs, commercial software developers created several useful though proprietary technologies. These included Active Server Pages (ASP), developed by Microsoft Corp. for its Windows NT® platform, and ColdFusion Mark-up Language (CFML), based on a popular Web application scripting engine from Allaire Corp. While this was an improvement over CGI scripts or programs, these solutions failed to effectively address the issues of advanced user administration, file system management, and database design and maintenance. In addition, such commercial developments typically yielded proprietary, monolithic, and vertically integrated servers requiring extensive and highly specialized technical expertise to set up and maintain.

*Considering this progression, what is lacking becomes clear: a solution that addresses, with a single, horizontal platform, the broader requirements of the current and future online environment-which is precisely where Reef Internetware comes in.*

---

## INTERNET APPLICATION ARCHITECTURE REQUIREMENTS

Reef's approach to both current and emerging e-business requirements was to start from scratch and take a fresh look at the many challenges organizations competing in the Internet Age business environment face-then examine the best ways to meet those challenges. Stepping back, Reef identified, among others, the following Internet application architecture requirements that drove the development of the Reef Internetware solution: a true suite of modular and customizable software applications built on a shared, horizontal engine that provides common services and functionality to all applications.

### *Standard foundation technology*

First, the new solution must be based on a secure, widely accepted, standards-oriented, and future-proof programming technology that supports rapid, object-oriented development of a platform operating exclusively in a networked environment.

For this, Reef chose the Java suite of programming languages, tools, and protocols which, when coupled with a highly disciplined object definition and modeling development methodology, yields unprecedented advantages in software quality and time-to-market. All Reef developments use Java and Java-based technologies. Extensible Mark-up Language (XML)-based protocols and mechanisms have also been integrated into the Reef Internetware suite.

### *Shared software application backbone*

Next, it was clear that a single monolithic software product would never be able to satisfy the diverse requirements of the new and increasingly complex universe of Internet applications. On the other hand, e-business applications, whatever their nature, must be able to share data and functionality, this indicated the need for a software bus upon which applications could be built. Services made available on the software bus would be transversal, or available to all application modules.

From this notion arose the Reef Engine, a scalable software platform upon which network applications can be integrated or extended, and which provides shared access to data and functionality.

### *Flexible data management*

To eliminate specific database knowledge as a required skill set for the development of Reef-based applications, it was determined that a relational database must be integral to the Reef solution. At the same time, this relational database must retain the flexibility and modularity of an object-oriented implementation. Administrators, for example, could not be expected to reconfigure a database whenever a change to a Web site or an application was required.

Reef met this challenge with a revolutionary approach to database container definitions based on simple HTML forms. This unique approach enables business users within an organization to define data structures "on the fly" through a simple, familiar browser interface.

### *User and access management*

User management and access control must be part of the architecture as well. These capabilities must be flexible enough to meet real-world user profile administrative requirements and support integration with current methods for user, employee, or customer management.

Reef satisfied this requirement with a hierarchical, yet elastic user management and access control scheme that supports multi-level administrative interfaces as well as the personalization of user experiences and data access. These user management capabilities are defined for both administrators and maintainers of the Reef site—as well as for clients of the site, thereby enabling robust cross-functional and single-point administration of all user management needs.

### *Multimedia management*

To support the increasingly media-rich nature of modern online content, a mechanism for the streamlined storage, management, and retrieval of multimedia data objects was a fundamental requirement. To meet it, Reef developed a searchable online repository, or "mediastore", to be offered as a shared service to all applications built on the Reef architecture.

### *Platform independence*

A solution that could be installed on a variety of server hardware and operating platforms, as well as support numerous Web servers, databases, and other software technologies already in use at customers' sites, were both critical requirements for success.

Reef's use of Java and the ubiquitous Java Virtual Machine (JVM) satisfies the former requirement, while careful design of the interfaces to local and remote modules satisfies the latter.

### *Browser-based administration*

For maximum solution flexibility, Reef determined that administrator and developer access to the solution must take advantage of thin client (i.e., browser) technology, providing the twin advantages of client platform independence and universal administrative access from anywhere on the Internet.

As a result, Reef products rely entirely on standard browsers for all administrative and user access, with no need for specialized or proprietary client components.

### *Open architecture and API*

Focused on delivering a comprehensive, integrated foundation for Internet-based business, Reef rolled all of the above-mentioned capabilities into a single Reef Engine architecture with a standards-based API for customization and provision for connections to external networked services, including back-office systems. Because interfaces between new Web applications and legacy systems are based on standardized protocols, Reef provides an environment for application interconnection.

## Rapid deployment

To help organizations meet increasingly demanding time-to-market requirements, the Reef solution includes a mechanism to rapidly build online applications and services supporting the core Internet application arenas of content, commerce, community, and coordination—using generally available skill sets. These applications meet specific needs and offer distinct benefits within their respective categories, while taking common advantage of the underlying Reef Engine for horizontal capabilities such as global searching.

Designed from the ground up to meet the rapidly emerging requirements of e-business, Reef's resulting combination of technologies comprises an unprecedented environment for the rapid development and deployment of Internet-enabled applications.

---

## REEF ARCHITECTURAL ELEMENTS

The following discussion describes in detail the core architectural elements that make up the Reef Internetware solution. This begins with a logical view of the architecture, followed by a discussion of key functional elements and how they work together in a Java-enabled environment, and ending with an overview of security and access control mechanisms.

This discussion provides a foundation for the remainder of this paper, which includes an examination of the Reef Engine at a product level, an overview of key capabilities, and a discussion of how customized solutions are supported through multiple application programming interface (API) levels.

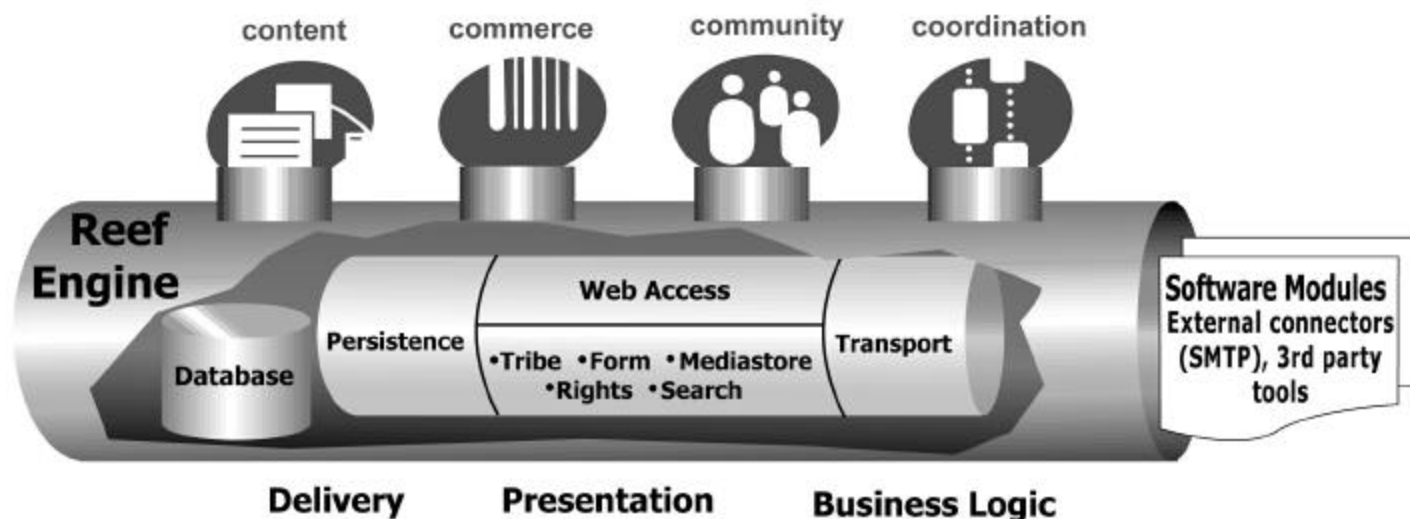
---

### Logical architecture

Before examining the functional elements of the Reef architecture, it is useful to understand the logical relationships between the various system components comprising the Reef Engine and applications. To this end, the Reef architecture is divided into the following logical elements:

- Foundation
- Application
- Web access
- Transport

### Reef Applications



## FOUNDATION

Foundation modules are "services" whose functionality and data are shared among Reef applications. These include modules such as user management and access rights, multimedia storage, indexing, and management or search on common data. The object orientation of the Foundation allows Reef to provide reusable code for all Reef products, enriching Reef product feature sets and accelerating product releases.

## APPLICATION

Application elements include the application code which, in turn, depends upon the Foundation for shared services. One important aspect of the Reef design is that application code is completely independent of the Web environment, enabling business objects to be triggered from anywhere.

The link between the Web and the application code is accomplished through the Web access layer. This allows application code to remain independent of access methods and provide upgrade capabilities to thin clients beyond Web browsers (e.g., palmtop computers, mobile telephones). Presentation logic is separated from business logic, providing a high degree of site design flexibility as well as facilitating rapid module development.

## WEB ACCESS

Web access is where all code related to linking product modules and the Web server is stored. This is implemented through the use of an access servlet, for authentication and environment settings, as well as a set of Java beans that allow the applications to be accessed via server-side JavaServer Pages™ (JSPs). These JSPs are used instead of HTML files and are filtered by the Web server through an application server for execution.

## TRANSPORT

Flexibility in the selection and use of transport mechanisms is very important and a key aspect of the Reef architecture. The obvious transport scheme for Java modules is Remote Method Invocation (RMI)- and indeed, this is a component of the Reef Engine design.

---

### *Java-enabled architecture*

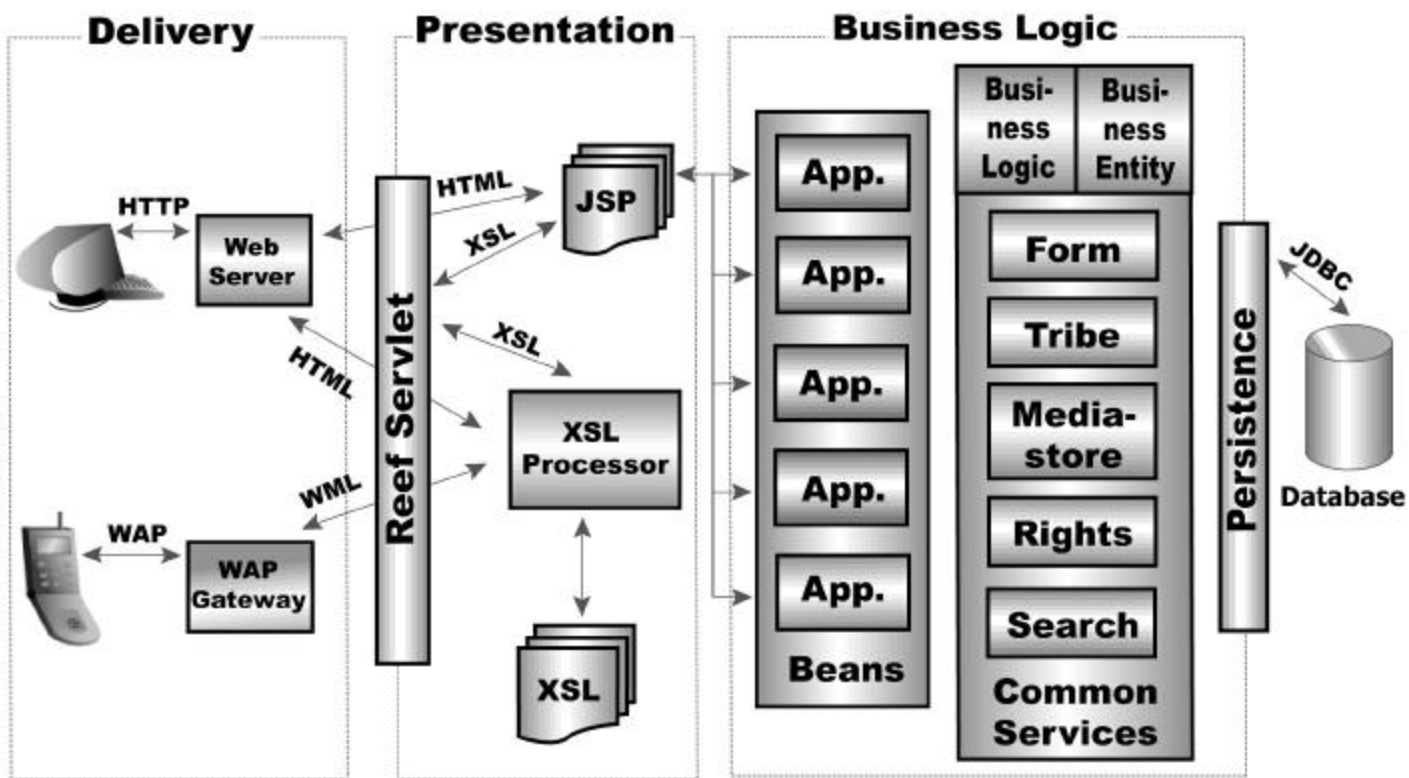
The Reef architecture takes full advantage of the openness and flexibility of the Java development environment for Web-enabled applications. Java-a set of object-oriented programming technologies originally developed at Sun Microsystems and now being extended and adapted through the open Java Community ProcessSM (JCP) mechanism-provides a number of significant benefits to Reef, Reef customers, Reef development partners, and Reef channel and distribution partners.

- Rapid development-First and foremost, when Java development is preceded by detailed object modeling, coding and debugging of new modules is very rapid compared to more traditional approaches, reducing time-to-market for new products and features. Reef takes full advantage of these benefits by working to the Unified Modeling Language (UML) modeling standard and development methodology.
- Robust networking-While Java can be employed in non-networked environments, it was the first complete programming language to be devised, and which has evolved, with networking applications and requirements as fundamental to its design.

- Modularity and integration-Finally, Java is an open standard and widely supported, with a built-in modularity that encourages the integration of software components from multiple vendors. The Reef architecture includes and relies on standard components that interact in well-defined ways, supporting a general Reef design goal of architectural independence enabling the integration of best of breed server modules and technologies.

Key aspects of the Java-enabled architecture include:

- Delivery
- Presentation
- Business logic
- Common services
- Database



## DELIVERY

Traditional Web server. The job of the Web server, a software service or daemon which runs in the background on a networked host, is to respond to Hypertext Transfer Protocol (HTTP) requests by serving up HTML pages to remote browsers. Traditionally, Web servers interact with local file systems to find the requested static content, or employ scripting languages to present dynamic content or to access databases.

Reef Web server. In the Reef architecture, the Web server is a discrete, third-party process, that interacts with the servlet engine (running inside the local Java Virtual Machine or JVM) in standard ways to assemble and present dynamic content that is stored in a database. Because virtually all Web server software supports standard connections to the servlet engine, Reef customers are free to choose the Web server that best meets their needs.

## PRESENTATION

HTML. HTML is a set of tag-based text mark-up rules that is so easy to learn and use that millions of people have employed it to build their own Web pages. As a result, HTML coding skills are widespread and easy to secure. However, since HTML was designed primarily for the presentation of static content, a standard way to support dynamic content models in a Java environment is required.

JSP. JavaServer Pages technology-a Java-oriented extension to the HTML tag language-separates the user interface from content generation, enabling designers to change the overall Web page layout without altering the underlying dynamic content.

JSP is used with Reef products in site definition and customization. The presentation and structure of the online applications are coded in JSP by the application integrator or custom developer (typically a Web development services firm or the in-house IT staff of a large organization). When these pages are later called by the Web site visitor or user, HTML content is dynamically generated by the Reef Internetware application and returned to the client browser.

JSP technology enjoys a performance advantage over other HTML alternatives in that JSPs are compiled into Java classes on the server rather than being laboriously parsed every time they are called.

Reef employs straightforward HTML for the development of reusable presentation items such as Reef Publisher templates, and employs JSP technology for the definition of dynamic content.

Reef servlet. Java classes that are called directly by the Web server are known as servlets. The Reef servlet addresses requests from the Web server for dynamic content and dispatches them to the respective JSPs, providing them with a common services context for processing. The Reef servlet currently operates in the Allaire JRun environment.

XSL and XML. Reef is also a pioneer in the use, with some products, of XSL, the style sheet description language of Extensible Mark-up Language (XML), which itself is widely hailed as the eventual replacement of the current HTML standard for Internet presentation. XSL provides a more structured separation of presentation and logic than does JSP.

Some Reef products are configured so that the JSP logic returns XML, which is easily formatted using XSL before being converted to HTML for real-time presentation to the user. XSL allows HTML programmers to quickly define page presentation templates, providing a simple and flexible environment for site customization. Moreover, the transform mechanism supporting XSL can be enhanced to support other XML-based page description protocols such as Wireless Application Protocol (WAP), or even legacy technologies such as ASP and CFML.

## BUSINESS LOGIC

Java beans and JSP. In Java parlance, a "bean" is a reusable software component-a Java class (or subroutine) with special properties allowing it to be implemented in a self-contained and modular fashion. Java beans are called from JSP tags in the same way that subroutines and functions are called from the main flow of software programs. In the Reef product architecture, the Reef JSP API is the set of available beans that can be called from JSPs.

By relying on Java beans at this level of development, Reef leverages the broad advantages of the Java language itself. Furthermore, since Java beans are well understood and defined, Reef is in an excellent position to extend its open APIs below the JSP level. By doing so, Reef enables its customers and partners to write custom beans to meet specific requirements such as performing custom calculations on a dataset before presenting the result to the Web site visitor. Java beans are also written to facilitate connections to external data sources or services, such as payment processing bureaus.

## COMMON SERVICES

Business logic and persistence layer. The common services logic processes the requests from the Reef application servlets, providing application-level and low-level business logic as well as a persistent interface to the database. The persistence layer also includes the object mapping onto the native relational database.

Vendor independence. There are several challenges related to the mapping of object architectures to relational databases, particularly in the area of preserving inheritance of object properties. While a number of application server implementations make strides in this area, a design goal of the Reef architecture was vendor independence at this level, so that Reef customers were not locked into using particular application server suppliers. For this reason, the persistence layer is built into the interface between the business objects and the data store.

Open integration. Reef's use of open standards such as Java RMI for server connectivity and communications holds open the promise of future integration with third-party software products at the Web server level.

## DATABASE

Content storage. Content and site schema is stored in an embedded object relational database. The Reef architecture interacts with the database through a Java Database Connectivity (JDBC) connection. The unique recursive properties of the Reef database design allow applications to scale up and grow organically with little or no significant changes to the underlying database. Moreover, changes to data storage objects such as online forms are facilitated through the browser-level interface, with no specific database knowledge required of users or administrators. Customers hosting Reef Internetware need only the usual database maintenance skills.

Physical separation. The Reef Engine and the database can be hosted on different servers, ensuring greater flexibility in your network design and more performance scalability.

---

## *Security and access control*

Security is, of course, of primary concern in any networked application environment. Reef approaches security from multiple angles.

## SYSTEM-LEVEL SECURITY

By its nature, the Java Virtual Machine (JVM) architecture offers superior system security over traditional shell-based CGI Web applications. Using the Java APIs, there is no need to execute user-supplied commands in the host system shell. The Reef Engine has been developed with system security in mind, with no Web access to the host file system, for example.

User-level system access and security is controlled through the Reef Tribe Manager administration application, which can be used to define and edit user profiles and access rights for all Reef applications. These profiles allow administrators to define user permissions with a high degree of control and granularity, over both specific applications as well as specific functionality within these applications. User profiles, including passwords, are securely maintained in a database—not in the host file system. That means that with the Reef architecture, username/password combinations are never transmitted between hosts.

## USER AUTHENTICATION

Reef Internetware implements secure authentication through the intuitive use of usernames and passwords. Once authenticated, the user's security profile determines access settings for functionality within the Reef Internetware suite. Future enhancements will include additional authentication security policy settings such as password length and composition limitations (e.g., requiring passwords of a certain minimum length and including a mix of numbers and letters), password expiration, and force password change.

## USER AUTHORIZATION

Reef employs a two-level user authorization mechanism. First, through the common access controller servlet, the Reef Web access layer verifies whether the page requested by a given user is allowed or not. Second, the Reef application itself uses access control settings (again, configured through the Reef Tribe Manager administration application) to determine whether requested actions are legal. This security can be further augmented through the use of the browser's own built-in security tools.

## ENCRYPTION AND SECURE SESSIONS

Reef Internetware also supports Web servers that make use of Secure Sockets Layer (SSL) in addition to other secure protocol connections such as SHTTP. Using SSL and HTTPS, the client connection to the browser is authenticated and a secure and encrypted link between the two systems is created. No data is passed via "clear text" between the systems and is therefore nearly impervious to any attempt to sniff data, snoop passwords, or spoof identities. Administrators can choose to place either some or all of a given site under this level of security. Sites built using these technologies with Reef Internetware are guaranteed the highest degree of security.

## DIRECTORY SERVICES

Lightweight Directory Access Protocol (LDAP) is playing an increasingly important role in Web site management, primarily in the area of user profile administration. With its eye on this widely used standard, Reef is actively working to embed LDAP access into its Foundation as a component of Reef Tribe management. This will enable Reef Tribe Manager to automatically use any existing LDAP infrastructure (e.g., an employee or customer directory) upon deployment, in effect, enabling Reef Tribe Manager to use any LDAP directory as if it were its own. It will also be possible for LDAP-compliant directory entries to be employed for user authentication.

---

## THE REEF ENGINE

The Reef Engine is a rich core of shared services and systems that provides critical functionality to all Reef applications and eases their integration into existing network infrastructures. By design, the Reef Engine is not an independent application or development environment, but is packaged with one or more Reef applications.

While many of the key advantages of the Reef Internetware solution are attributable to the Reef applications suite, these applications would not be nearly as powerful without the transversal, or horizontal services that the Reef Engine provides them with. Five of the major services include:

- Reef forms
- Reef tribes
- Mediastore
- Rights
- Search

### *Reef forms*

Reef forms are database "containers" based on HTML forms. The Reef form API is fundamental to application building with Reef products, and uniquely powerful.

Reef form is a transversal service that allows developers and administrators to define database records and structures using a simple object-oriented Web interface. This is often used to specify the organization of user profiles, e-commerce product records, forum messages, and so on. With Reef forms, administrators can simply specify the data or fields they want to store, without having to know about the underlying database technology or structure, and without having to use any SQL.

In addition to dynamically generating the structure of the data, Reef forms also provides the mechanism for both inputting and outputting the data by serving as a template for the entry and display of the data. More importantly, using the Reef Form Manager application, forms can be changed or updated at any time, by business users and administrators, with changes being reflected instantly in the underlying database structure, and with no manual database tuning or reconfiguration. Aside from the obvious benefits this brings to day-to-day administration, the support of dynamic, "on the fly" changes to Reef forms makes the Reef solution invaluable for maintaining ever-evolving online applications and sites.

Reef Form Manager is itself a highly customizable application providing whatever graphical user interface is necessary to allow administrators of all technical abilities to easily support and manage Web applications.

### *Reef tribes*

Reef tribes are used to manage groups of user profiles. Users are arranged into containers called tribes, which themselves can be hierarchical and can contain sub-tribes. Reef products employ Reef tribes to create, edit, and delete users, tribes, and sub-tribes, and to manage user access rights.

Reef tribes work closely with Reef forms for data entry and retrieval, and are configured through the Web-based Reef Tribe Manager application or GUI. Reef Tribe Manager is the user and Web object access management tool for the Reef environment.

As with Reef forms, Reef tribes provide complete flexibility in defining and maintaining user profiles. User profile structures can be updated at any time, simply by changing the Reef form that defines the profile. Moreover, users can be associated with as many tribes as desired, and new associations can easily be enabled at any time.

As a shared service provided by the Reef Engine, Reef tribes provide centralized administration of users and objects for an entire Web site or application, even when multiple Reef products are employed. And while Reef tribes allow many points of entry for user access or account administration, actual tribe management is centralized. All tribes and user entities can be administered from a single Reef Tribe Manager GUI front end, which, in and of itself, is highly customizable.

### *Mediastore*

The Reef mediastore is a transversal service that provides a repository of any digital data to any Reef application. Graphics, multimedia, text blocks, documents, and other data blobs are supported, and most can be recognized by the mediastore according to Multipurpose Internet Mail Extension (MIME) type. The implementation of the Reef mediastore highlights not only the importance of rich multimedia format support by Web applications, but the need to ensure that the multimedia elements handling facility can be shared across multiple applications in an online environment.

### *Rights*

A rich user rights management capability is built into Reef Internetware. User rights management includes common services and APIs for all Reef applications and the ability for browser-level administrators to

- assign and remove access rights to each tribe and sub-tribe
- assign and remove access rights to each tribe on each object (e.g., publication, forum, form) that the Reef product produces.

Administrators can assign each tribe and sub-tribe specific access rights to the online material that they manage-the right to visit a site but not to update it, for example, or the right to visit one page on a site but not another.

### *Search*

When multiple platforms are used to build a complete online presence-as is often the case in today's rapidly evolving online world-searching through all the differing environments and technologies can become highly problematic. Addressing this challenge, the Reef Engine supports multi-criteria text searches specific to application areas or across all applications in the Reef Internetware suite. In addition to searching for simple text strings, the search engine also supports searching of keywords and data or content attributes and tags. And with the Reef mediastore, it's possible to extend the search capability even further to search within and for multimedia objects.

---

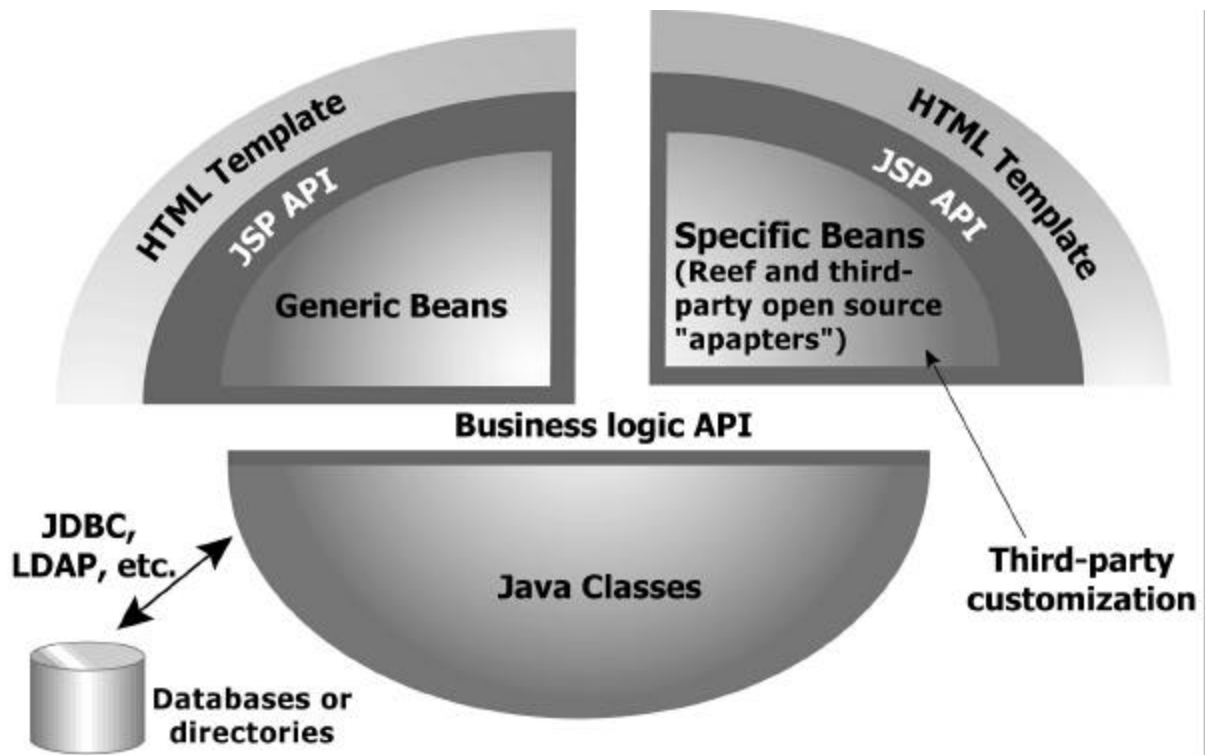
## THREE LEVELS OF CUSTOMIZATION

Modern online networked applications must support rapid customization to service the specific requirements of customers and to enable broad interaction with external products and services such as e-commerce payment processing bureaus or syndicated content providers.

Anticipating and facilitating this situation, Reef Internetware includes multiple customization mechanisms that provide real value for end users, administrators, Application Service Providers (ASPs), and resellers. These mechanisms take into account the fact that customization requirements for Web development are extremely broad, and must meet differing requirements from a variety of customers with a full spectrum of skill levels. At the same time, Reef understands that standard methods of customization are an absolute requirement, because developers and administrators are reluctant to re-train to learn new techniques to support their applications.

Starting at a basic level and moving to a more advanced level, a number of nested development interfaces address access to the Reef environment for application customization, including:

- HTML templates
  - JSP APIs
  - Generic and specific beans
-



## Basic: HTML templates

HTML templates are instrumental in the rapid and consistent deployment of online content by business users. With Reef Publisher, libraries of development templates are considered a high-level API for Reef users who are setting up and managing Web sites, intranets, or extranets. These templates can be used to develop administrative interfaces to Reef Publisher applications, for example, as well as to make other cosmetic enhancements.

In addition to pre-configured templates, Reef customers can easily build their own personalized template libraries using basic HTML coding. Reef templates are based on a simple extension to the HTML standard. The template level, employing straightforward HTML, can be considered the highest level of customization method for users of Reef Publisher.

Looking ahead, Reef products will increasingly take advantage of XSL as a description language for page formatting and templates. XSL allows the use of HTML-like "style sheets" to customize the look and feel of Web applications.

## Intermediate: JSP APIs

Below the template layer lies what is called the "integrator level", which describes a customization layer that is typically managed by product integrators and technical maintainers rather than by end users, and which takes advantage of the JSP mechanism.

JSP is a standard syntax of an extension of HTML, and working in JSP uses many of the same skills as working in HTML. With Reef products, JSP is the primary customization interface. Integrators build applications by developing JSP pages with standard and Reef-specific tags. The Reef tags call generic or specific Java beans, which in turn interface with the business logic in the Reef applications themselves.

Because both Java and HTML are such pervasive and open technologies, the skill set necessary to use and create JSPs is typically found within the workforce.

## *Advanced: Generic and specific beans*

Reef Internetware allows the use of two sets of Java beans-generic and specific beans-both of which are accessed directly by the JSP tags. Generic beans are, as the term implies, standard components of the standard Reef offering. The generic beans and how to use them are documented in Reef integration manuals.

Specific beans, on the other hand, are developed against an internal API to provide functionality specific to a customer requirement or application. Reef is developing a common, public API to the generic beans and Java classes supporting specific bean development. This upcoming API will provide intelligent and structured access from any Reef application to all of the functionality provided by the Reef Engine.

---

## CONCLUSION

### *Business is dynamic. Take control with Reef.*

Designed from the ground up to meet the rapidly emerging requirements of e-business, Reef Internetware comprises an unprecedented environment for the rapid development and deployment of Internet-enabled applications. Reef's integrated approach to e-business applications management enables organizations to:

- Empower non-technical business professionals to be Web contributors, administering ebusiness applications anywhere, anytime, using an easy, familiar browser interface
- Enable increasingly decentralized organizations to enhance communications and improve business processes with customers, suppliers, partners, and employees in dynamic, net worked environments
- Speed up deployment, accelerate time to market, and optimize IT efficiencies by applying common skills and consistent technology interfaces across multiple e-business applications
- Easily integrate e-business applications with existing third-party and legacy solutions, and add new technologies in the future, without having to change the underlying foundation
- Use today's location- and platform-independent standards and technologies to position themselves for future technologies as e-business requirements evolve beyond the Internet

WWW.REEF.COM

The information in this document is subject to change without notice. © Copyright 2000. REEF S.A./N.V. All Rights Reserved. Reef, the Reef logo, and Reef Internetware are registered trademarks of Reef S.A./N.V. Java is a registered trademark and Solaris is a trademark of Sun Microsystems, Inc. in the United States and other countries. All other trademarks are the property of their respective owners.