The visual web computing platform

RealityServer® Technical Overview

December 2009

Copyright Information

About This Document

RealityServer® provides a platform to collaborate and iterate on virtual 3D designs - from any web-enabled device using a wide range of consumer and enterprise applications. Creating, physically correct, photorealistic images in seconds instead of minutes or hours, RealityServer accelerates review cycles and potentially shaves weeks or months off the design cycle.

This document outlines a solution for 3D visualization; it contains the following sections:

- Functional Overview
- RealityServer Platform Architecture
- 3D Application Service Software Architecture
- More Information
- Glossary of Terms

Related Information

For more detailed information about RealityServer, you can also read these documents:

- RealityServer 3.0 White Paper
- RealityServer 3.0 Solution Areas

Functional Overview

The scalable, server-based RealityServer architecture enables you to create multi-user 3D applications and 3D application services without significant client processing requirements. Instead of putting the resource burden on the client, the application state and rendering are managed by the server. Thus, only a sequence of rendered and suitably compressed images or video data are delivered to the client. These images are displayable even on lightweight wireless mobile devices such as PDAs and smart phones. The bandwidth requirements for transmitting imagery remain bounded by a constant; independent of the complexity of the 3D data the user is interacting with.

In this section you will learn more about:

- RealityServer Architecture
- RealityServer Integration
- Basic RealityServer Runtime Services
- RealityServer Application Development
- Importing 3D Data
- Interacting With 3D Data

RealityServer Architecture

Figure 1 on page 5 shows the architecture of RealityServer. This architecture provides the foundation necessary to create scalable 3D applications and services. Functionality is provided for handling incoming client requests, manipulating the user state, handling rendering operations and transmitting information back to the client.

A typical client for RealityServer is a web browser. The client communicates with the server through the Internet, using either HTTP or Real Time Messaging Protocol (RTMP). After processing the request on the server-side, a response is delivered by the HTTP server to the client. If RTMP is used, then streaming capabilities will be enabled, allowing compressed video streams to be delivered to compatible clients.

Other clients that support the use of the HTTP or RTMP protocols can also be supported. Furthermore, the use of Web Services allows RealityServer to be integrated directly into enterprise infrastructures that use a Service Oriented Architecture (SOA). In such configurations the client can be another server or service that can then easily integrate RealityServer functionality with business logic and client-side presentation logic.

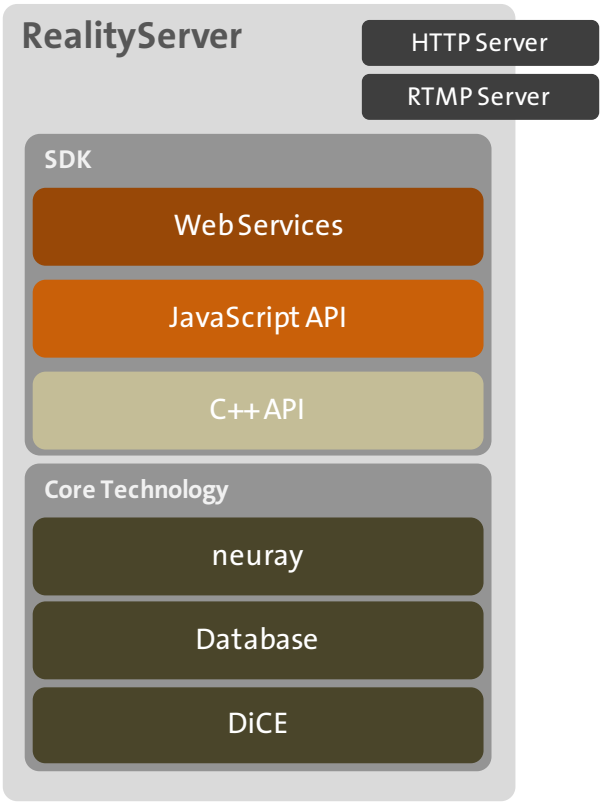

Figure 1: RealityServer architecture overview.

Among the unique advantages of the RealityServer architecture you will find:

- Optimized hardware use
- 3D Data Re-use
- Confidential Data Stays Confidential
- Video Compression and Streaming
- Efficient Storage of Multi-user Data
- Server-based Collaboration
- Optimal Performance and Dynamic Load Balancing
- Reliability

Optimized Hardware Use

As demands grow, the provider of the service can easily upgrade the server capabilities. It is no longer necessary to demand that end-users upgrade their computers, use specific operating systems, or download plug-ins. Each of these demands would limit and compromise the size of the target audience, as well as the complexity, security and quality of the 3D data, while the server approach imposes no limits and reduces the total cost of ownership by the optimal use of resources.

3D Data Re-use

3D data is created with software applications such as Dassault Systèmes' CATIA® and SolidWorks®, Autodesk's AutoCAD®, Inventor®, 3ds max®, Softimage® and Maya® in all stages of product design, manufacturing, and in architecture and engineering. The resulting 3D data is much too large to download to a client, but it is manageable on a server. RealityServer enables the import of standard data exchange formats such as COLLADA, OBJ and DWF as well as the mental images .mi file format, which is exportable from many commercial applications that integrate mental ray®[1]. Developers can additionally create plug-ins to implement support for other 3D file formats.

Confidential Data Stays Confidential

The architecture of RealityServer ensures that users never have access to the original 3D data, while still being able to interact dynamically with it. By using additional standard technologies such as HTTPS or secure VPN, imagery generated with RealityServer can also be protected. Integration with such technology is straightforward thanks to the use of standards-based web technology.

Video Compression and Streaming

The use of RTMP when combined with video compression enables RealityServer to provide live video streams of interactive 3D content. The streaming communication is bi-directional, allowing for efficient, low latency exchange between the client and server. With the C++ Plug-in API, developers can add support for streaming any video format required, or, alternatively, for writing video output to disk.

Efficient Storage of Multi-user Data

With RealityServer, developers can create efficient multi-user applications that do not require duplication of large datasets on a per-user basis. Only data that differs between users needs to be stored multiple times. As a result, the storage requirements in multi-user scenarios are dramatically reduced.

Server-based Collaboration

RealityServer allows a virtual 3D environment to be shared among multiple users for purposes of collaborative design, design review, or multi-player games. A key feature of RealityServer is the consistent and efficient management of private data available only to a client, such as the current viewpoint, and public data that is the same for all clients.

[1] For more information about mental ray, see the mental ray white paper available at: www.mentalimages.com/fileadmin/user_upload/PDF/mental_ray_functional_overview-091005.pdf

Optimal Performance and Dynamic Load Balancing

RealityServer exploits data coherence to optimize computation and load balancing, using advanced proprietary caching, scheduling, and data flow management techniques. In situations such as online modification of the database by users with unpredictable impact on scene complexity, RealityServer is able to provide graceful performance degradation of the system.

Reliability

Data is stored in a highly redundant, distributed way in scalable, massively parallel server hardware.

RealityServer Integration

The RealityServer architecture allows the seamless integration of 3D applications services into service-oriented software architectures. As shown in Figure 2, this enables enterprises and service providers to integrate RealityServer into their existing application services development and deployment processes.

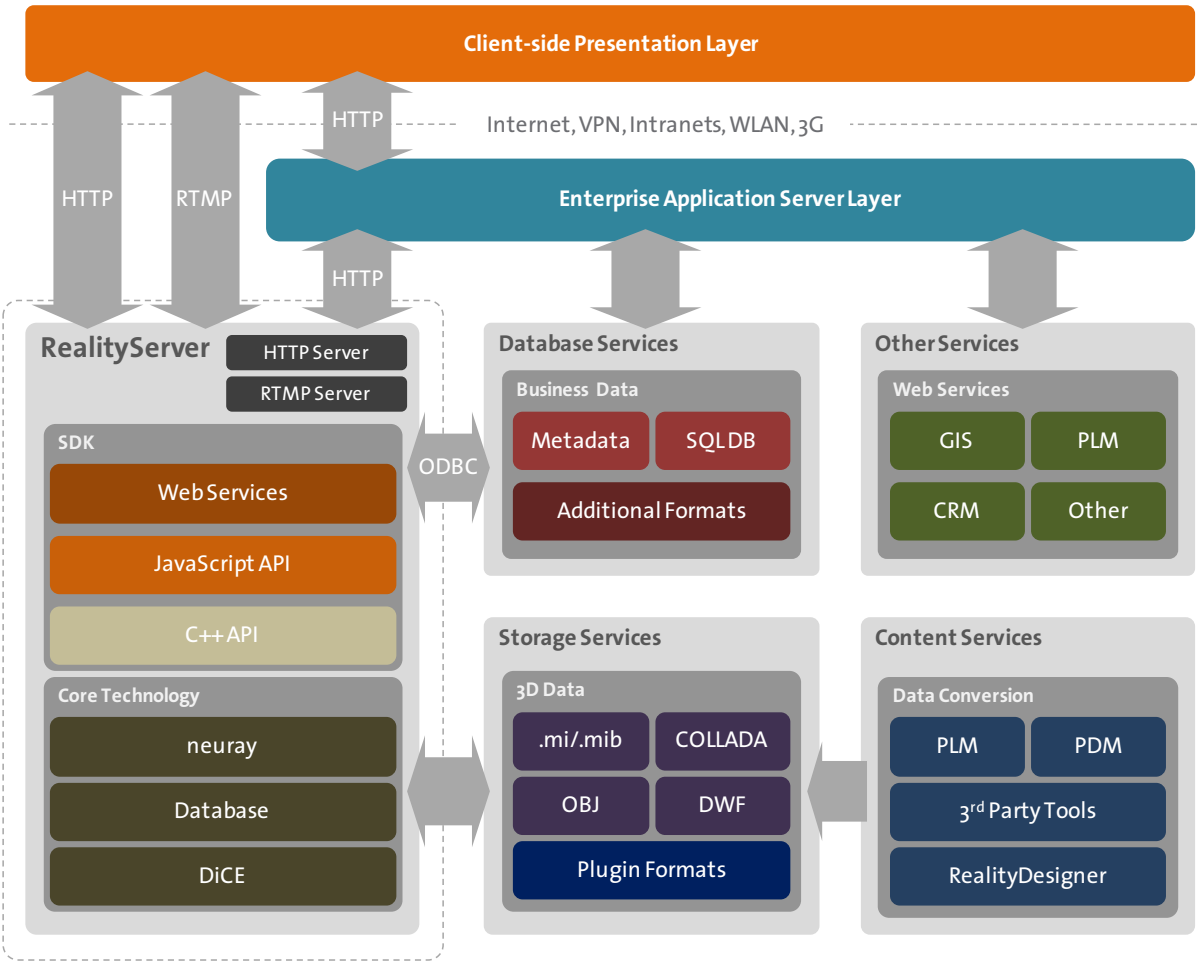

Figure 2: Example deployment of RealityServer within an Enterprise using a Service Oriented Architecture (SOA).

The integration of RealityServer into existing enterprise development processes provides the following fundamental benefits:

- Leverage Existing Applications and 3D Data
- Seamless Integration of 3D Applications
- Scalability

Leverage Existing Applications and 3D Data

By using 3D data created with Computer Aided Design (CAD) and Digital Content Creation (DCC) software products, optionally in conjunction with Product Lifecycle Management (PLM) solutions and other enterprise software such as Customer Relationship Management (CRM), enterprises can leverage existing applications and 3D data. This allows for timely, efficient and cost-effective use of 3D data that may change over time.

Seamless Integration of 3D Applications

The RealityServer platform allows for developing and deploying 3D applications with line-of-business applications and services. These applications support new service-oriented online processes in-house, with business partners, and with or among external users. Enterprises can complement and extend existing or newly developed Web Services, as well as B2B, B2C, and C2C processes that are built upon these Web Services, with interactive 3D data components.

Scalability

As RealityServer is scalable, 3D applications built on it can support thousands of simultaneous, online users. This is of particular value to all companies; including companies specialized in manufacturing, retailing, navigation, entertainment, exploration and research. Indeed, RealityServer benefits companies that have large and ongoing investments in original 3D data and require remote access to it or want to provide a platform for the creation of 3D Web Services and mash-ups, for example, by combining search with 3D information or data.

Basic RealityServer Runtime Services

RealityServer 3.0 provides basic runtime services. Note that features that are planned for availability with a later version of RealityServer are identified accordingly. In addition, application developers can create application-specific services using RealityServer's extensive APIs.

This section provides you with details on the basic runtime services of RealityServer:

- Remote and Secure Access to 3D Data
- Multiple Online Data Sets
- High-Quality Visual Feedback Using Server-Side Rendering
- Simultaneous User Access
- Collaboration Using Shared Data
- User Interaction with 3D Data
- Client Interactivity Using Synchronization

Remote and Secure Access to 3D Data

RealityServer allows 3D data to reside and be processed entirely on the server, thereby allowing the largest of machines, or groups of machines, to process actions on very big 3D databases and display the results on remote clients. By never letting the original 3D data leave the server, or giving clients direct access to it, your data remains secure.

Multiple Online Data Sets

RealityServer allows for an unrestricted number of 3D data sets, known as "Worlds", to be accessible simultaneously.

High-Quality Visual Feedback Using Server-Side Rendering

RealityServer can reflect client interaction by rendering an image of the most recent state of the 3D data on the server and by sending this image to the client. The only client requirement is an HTML browser.

Simultaneous User Access

The RealityServer database manages user access to its data and maintains its state. Based on the application code that supports data sharing, the RealityServer database can create copies of 3D data objects such as cameras to provide personalized user experiences.

Collaboration Using Shared Data

Multiple users can also have collaborative access to the same data. The RealityServer database manages user access to the data based on the application code that supports collaboration. Given that data is shared and "private" instances of 3D objects are created only as needed, an increase in the number of participants in a collaborative session has a small impact on server resources.

User Interaction with 3D Data

Users can act on 3D data remotely from the client. Based on client application code written for the User Interface, user actions may trigger processing requests to the server. Depending on the parameter values sent with the client request and the server application code that is executed in response to the request, 3D data on the server may be updated, and an image rendered (or a stream of images in interactive mode), and sent to the client to provide visual feedback.

Special client software is not required to view the visual representation of RealityServer 3D data on user devices. A browser that supports HTML and JavaScript is sufficient to be used as a front-end interface and display for a wide range of RealityServer applications.

The imatter® module of the RealityServer software platform allows to interactively create and modify geometric shapes in RealityServer applications remotely from any Internet connected device capable of running a browser.

Client Interactivity Using Synchronization

Applications requiring minimal latency can use an optional third-party client player that synchronizes changes of simplified representations of 3D data on the client with the original 3D data on the server. Synchronization allows users to request high-quality renderings of the original 3D data at any time.

RealityServer applications can use mental mesh™ functionality provided with RealityServer to optionally simplify 3D data for use in an optional third-party client player, based on properties specified in the application code. RealityServer applications can handle synchronization of the 3D data but the synchronization request must be triggered from the client application code. Optionally, third-party client applications can permit de-synchronized off-line operations on the basis of reduced 3D data sets alone.

RealityServer Application Development

The RealityServer platform allows the import of 3D data in RealityServer applications. With the RealityServer APIs, developers can write application code to interact with 3D data, and to integrate RealityServer applications with other enterprise applications and databases.

Importing 3D Data

3D data is created using leading Computer Aided Design (CAD) or Digital Content Creation (DCC) software products, or both. Many of these software products can export content from native formats to the .mi format, which is supported by RealityServer. The .mi format preserves as much of the original 3D data as possible.

Because the .mi format is an open and extensively documented standard 3D scene description format, developers may additionally input 3D data in the .mi format from other sources and via API-based integration.

Furthermore, RealityServer supports a number of 3D formats natively. These include, DWF, OBJ and the COLLADA Digital Asset Exchange (DAE) format.

Shading-related definitions are defined in MetaSL™, mental images's universal shading language.[2]

[2] For more information about MetaSL, see the mental mill® White Paper available at www.mentalimages.com/fileadmin/user_upload/PDF/mental_mill_features_031208.pdf. The last chapter in this document explains what MetaSL is and how it works.

Interacting With 3D Data

Developers must write the application code that enables users, individually or collaboratively, to visualize, manipulate, or modify 3D data.

The Web Services Framework, JavaScript and C++ APIs provided by RealityServer allow developers to interact with the 3D data loaded into the RealityServer database at runtime and create applications that present static or interactive 3D data to users. Both collaborative as well as individual interactions are fully supported. Modification of the appearance of objects, their arrangement and visibility, the environment, and the viewpoint can also readily be achieved.

By default, data is shared but the APIs can be used to define private data such as a camera for each user.

RealityServer allows for straightforward integration with enterprise processes, applications, and databases. Integrated application software may include for example, data importers, interpolation engines, physics engines, state machines, 3D modeling and 3D data creation systems, PLM and CRM systems and services.

The RealityServer Web Services Framework can be used to create custom services, allowing RealityServer applications to integrate with existing online platforms and applications through industry standard invocation protocols such as JSON-RPC, SOAP and REST.

RealityServer Platform Architecture

RealityServer features highly modular software architecture. The main modules, as shown in Figure 1 on page 5, are described in this chapter.

- HTTP Server
- RTMP Server (Streaming)
- Web Services Framework
- JavaScript API
- C++ Plug-in API
- Video Compression Plug-in API
- Visualization Solutions
- DiCE™
- RealityServer Database

HTTP Server

RealityServer applications use HTTP to communicate between the server and its clients. SHTTP can be used by adding a standard reverse proxy server in front of RealityServer.

RTMP Server (Streaming)

In addition to a standard HTTP server, RealityServer also includes a Real-Time Messaging Protocol (RTMP) server to provide video streaming and bi-directional communication among Adobe Flash® based clients or custom clients implementing the RTMP standard. A video compression plug-in API is also for customers to develop plug-ins for RealityServer. These plug-ins provide video codecs to be used for compression and streaming.

Additionally, RTMP provides a bi-directional stream that allows persistent server communication in a simple manner. This is extremely difficult to achieve with conventional HTTP methodologies.

Web Services Framework

Web Services are applications or services that primarily support application-to-application communication over a network. The underlying Web Services standards and technologies enable these applications or services to be described and deployed in a consistent way, and invoked in a secure and reliable manner.

The Web Services standards and technologies offer a number of benefits over traditional application-to-application communication technologies by simplifying the software development process, communications infrastructure, and system maintenance.

The RealityServer Web Services Framework enables developers to create and consume RealityServer Web Services easily and efficiently.

The RealityServer Web Services Framework provides access to the majority of the functionality defined in the RealityServer JavaScript API. Indeed, it supports protocols that enable communication between RealityServer Web Services and other Web Services. These protocols are JSON-RPC, SOAP, and REST.

Example implementations and re-usable software libraries are provided for Adobe Flex Builder® and Microsoft® Silverlight™, and Microsoft® .NET. The majority of application development using RealityServer can be performed with the Web Services Framework. For situations requiring additional functionality, developers can access the JavaScript and C++ Plug-in API.

JavaScript API

The JavaScript API allows developers to create new functionality that can be exposed to the Web Services Framework or called directly if required. This functionality is implemented as server-side JavaScript and can be developed as a *User Command* that will be automatically documented and exposed to the Web Services Framework. The JavaScript API is typically used to extend the services offered by RealityServer.

The JavaScript API provides:

- A server-side JavaScript interpreter which is an ECMA-compliant open source implementation of JavaScript,
- A direct interface to the RealityServer database, and
- Rapid development of new services that can be easily exposed by the Web Services Framework.

C++ Plug-in API

The C++ Plug-in API enables you to extend RealityServer with additional custom functionality that can then be made available to server-side JavaScript, and in turn to the Web Services framework. Usually the C++ Plug-in API is used for specific tasks requiring the functionality outlined below.

The C++ Plug-in API provides:

- The ability to create geometric entities at runtime,
- Creation of custom file format importers and exporters (including both image and 3D file formats),
- Image manipulation and processing or rendered output,
- Integration of third party C++ libraries and tools, and

- Very efficient execution of performance-critical custom application code.

Video Compression Plug-in API

Developers can add support for their own video formats to RealityServer using the Video Compression Plug-in API. The output of frames from RealityServer may be encoded to any format and either passed to the RTMP module for streaming, or handled directly by the plug-in (e.g., for writing to disk).

RealityServer provides you with a simple video codec, which is supported by Adobe® Flash® clients. However, more sophisticated codecs, such as H.264, can be added by recompiling the video compression plug-in with new libraries. The source code for this plug-in is provided. By using the streaming infrastructure and video compression plug-ins, RealityServer is able to provide full streaming video to RTMP-enabled clients such as Adobe® Flash® Player 10.

Visualization Solutions

The following table provides you with an overview of the visualization modes provided by RealityServer's renderers.

RealityServer Renderer	GPU	CPU
iray®	■	(■)[3]
MetaSL Raytracer	■	(■)[4]
MetaSL Rasterizer	■	
Sketch	■	

iray

The iray technology is available with RealityServer 3.0. iray generates photoreal imagery without introducing rendering algorithm-specific artifacts and without requiring the use of renderer-specific parameterizations. Users do not need to use complex shaders to approximate real, physical lighting effects. iray is a true "push-button" solution. When coupled with highly parallel processing platforms such as NVIDIA® GPUs, iray can deliver results in a progressive manner, providing a single process, which smoothly combines interactive pre-visualization and final frame rendering.

[3] Fallback
[4] GPU Cuda-based acceleration for AO, IBL and soft shadows.

MetaSL Raytracer

The MetaSL Raytracer renderer uses smart, versatile algorithms to support progressive, photorealistic rendering of large and dynamic scenes. This raytracer provides Global Illumination, Image-Based Lighting, and Ambient Occlusion shading effects. MetaSL® 1.1, surface, light, and environment shaders are supported.

MetaSL Rasterizer

MetaSL Rasterizer is a fast renderer that uses the most advanced hardware graphic acceleration technologies to render production-quality images. It is particularly notable for supporting the rendering of images of unlimited size; its very high sampling per pixel, and its soft shadow maps and sample filters. This renderer provides a high dynamic range, and production-level shaders, including reflection based on environment shaders. MetaSL 1.1 surface, light, and environment shaders are supported.

Sketch Renderer

The Sketch Renderer is a special-purpose rasterizer renderer for technical and architectural designs. It uses sophisticated algorithms to enhance object assemblies by rendering only the perceptually important edges of objects such as the silhouette, border, and crease edges. The RealityServer architecture allows the support of additional, application-specific renderers.

DiCE

mental images's Distributed Computing Environment (DiCE) is responsible for distributing the computational tasks of RealityServer, including rendering, across all available resources, both locally and remotely (when operating within a cluster).

RealityServer Database

The RealityServer database is designed to support the efficient storage of data required by applications that support multiple and possibly collaborating users. RealityServer data management is based on the following concepts:

- Transactions
- Worlds
- Sessions
- Scopes
- Scope Levels
- Scope Stacks

Transactions

Transactions bracket operations such as running scripts and rendering. For the duration of the transaction, the view of the database remains unchanged, regardless of how many other transactions are modifying the database at the same time. This behavior enables scripts or renderers to run safely even if another script run by another request is deleting or modifying the same data.

Scripts normally do not need to be aware of transactions; they are automatically started when an incoming request or other event starts a script, and stopped when that script terminates. RealityServer functions are provided however, for scripts that need to publish changes before they have finished running.

Worlds

Worlds are scenes loaded by scripts. A RealityServer function is provided for this operation. A script can work with a single world only at any time.

Database accesses automatically apply to the most recently loaded world. Worlds do not interfere with each other even though each world may be using similar names for database elements such as "cam" for the camera.

A script can load “sub-worlds” named assemblies into a world. Subworlds can be used for example, to populate a showroom with objects.

Sessions

Sessions represent user sessions. Sessions provide continuity between successive HTTP requests made by a user. A session is identified by a Session ID (SID) in the URL. RealityServer uses the SID in the URL, to load the script that is run in response to the request into the correct session. A login script typically creates sessions.

Scopes

Sessions not only join a world but also join a scope that defines how data is retrieved from the database. After a world and scope are joined, scripts run in a user session are automatically placed in this world and scope.

Scope Levels

Scope levels define which sessions can share database elements. Initially, the entire world is in the default public scope. This means that when the current transaction ends, all the changes it has made to the scene are published to all new transactions that start after that point, regardless of which session they are part of.

Conversely, if data is in a private scope, then each session sees its own private copy. If a transaction makes a change, only future transactions started in the same session will see that copy; all other sessions see the original data or their private

changes. A RealityServer function is provided to make database elements such as the camera private.

In addition to the public and private scope levels, there are intermediate scope levels for the database elements of a scene. For example, in a board game where the shapes of the pieces are public, the positions of the pieces are set to an intermediate scope level, and the cameras are private. Defining these three scope levels enables multiple instances of the board game to exist, players in a game to share pieces, and the player views of the board game to be private.

Scope Stacks

When an application uses intermediate scope levels, a scope stack is created when a user session joins these intermediate scopes. The scope stack is used by RealityServer to determine which sessions can access and share the data elements in intermediate scope levels. Using the board game example, the scope stack enables RealityServer to determine which board game a player belongs to.

3D Application Service Software Architecture

Figure 2 highlights the 3D application components of a RealityServer-based 3D application or 3D application service and their linkage to RealityServer system components, which was described in RealityServer Platform Architecture.

The layered 3D software architecture with the 3D application components in each layer is described in more detail in this section:

- Presentation layer
- Enterprise Application Service Layer
- Resource Layer
- Content Workflows

Presentation layer

The presentation layer represents the client-side User Interface (UI). This User Interface may be:

- One or more web pages, dynamically generated on the server and sent to the client. These web pages are generally displayed by a browser application that supports dynamic HTML, optionally with third party client frameworks such as Adobe® Flash® Player and Microsoft® Silverlight™,
- An application installed on the desktop that requires no browser and connects to the server intermittently.

Client-based User Interfaces typically use form elements, navigation bars, mouse events, and the arrow keys on a keyboard to observe and interact with the images rendered from the 3D data stored on the server. All standard methodologies for developing web applications work well with RealityServer, including straightforward JavaScript and HTML pages, with or without Ajax style functionality.

For applications that require sophisticated Graphical User Interfaces (GUIs), companies such as Adobe® and Microsoft® provide software development kits and frameworks such as the Adobe® Flex™ SDK and the Microsoft® Silverlight™ and Microsoft® .NET Framework to create Rich Internet Applications (RIAs).

You can use any technology that can access a web server and download an image to create 3D applications and services with RealityServer.

Enterprise Application Service Layer

To allow interaction and customization, additional information and application logic must be developed. This typically includes specialized business logic residing in the Enterprise Application Service Layer. This logic is likely to use atomic Web Services as building blocks for a larger application.

For example, a car manufacturer may want to develop a web-based car configurator using RealityServer to provide high quality visuals in a scalable fashion. Application logic, which allows potential customers to interact with the car choose arbitrary viewpoints. For example, potential customers can select parts of the car, open the car doors and the hood, and choose certain colors and materials and so forth. This would not only modify the state in RealityServer but also in other services, for example an ordering system, to allow the configured vehicle to be ordered.

Resource Layer

This layer represents the CAD and DCC software products used to create 3D data, the 3D data itself, and other enterprise applications, platforms and databases that may be accessed by RealityServer-based applications or which may use RealityServer as the 3D component of other SOA applications.

Content Workflows

RealityServer workflows from Autodesk's Maya® and 3ds Max® are currently available. These workflows leverage the mental ray integration of these applications to create .mi formatted 3D data. Simple to use viewing and interaction tools are provided with RealityServer for inspection and editing of the data.

There are a number of products that provide high fidelity .mi output. Third party software tools that support the export of .mi data can be used for converting

various data formats or exporting specialized data. Alternatively, RealityServer application developers can use RealityServer plug-ins to import data directly with the C++ API.

Data in COLLADA, OBJ and DWF are also natively supported and can be loaded directly into RealityServer for viewing and manipulation. COLLADA in particular allows a rich set of functionality to be expressed and stored within the 3D data, including materials, lighting and animation.

More Information

Contact Us, Visit Our Website and Forum

If you need more information about availability, pricing and licensing terms, and for a demonstration of real-world customer projects, please contact mental images at the e-mail addresses listed below, or visit our websites.

For	Contact us or visit
Software developers and end users,	sales@mental.com
- Companies that develop applications and services to be licensed to third parties, - System integrators, - Application service providers,	oemsales@mental.com
More information,	www.mentalimages.com/realityserver
Help and advice, visit our Developers' Forum,	www.forum.mentalimages.com/

Download RealityServer Developer Edition

To download the fully functional, free, Developer Edition of RealityServer, go to:

www.mentalimages.com/products/realityserver/downloads.html

Glossary of Terms

3D Web Services

3D application service that is internet-based. In a 3D SOA, it may be used by software applications written in various languages and running under different operating systems to exchange data over networks.

In this document Internet-based 3D application services are also referred to as 3D Web Services. Due to its 3D Service Oriented Architecture (3D SOA), RealityServer can be integrated as the 3D component into service-oriented software platforms. It also provides the platform for the deployment of 3D Software As A Service (3D SaaS). While Service Oriented Architecture and Software As A Service are by now well-established terms, the 3D variants made possible by RealityServer are introduced in this document.

B2B

Business-to-business is a term that is used to describe commerce dealings between businesses.

B2C

Business-to client is a term that is used to describe the activities of businesses providing end customers with products and services.

C2C

Consumer-to-consumer is a term that is used in electronic commerce. C2C concerns the electronic transactions that take place amongst consumers through another third party source.

CAD

Computer-aided design is about the design of objects, real or virtual using computer technology. CAD may be used in two-dimensional (2D) space to design curves and figures; or in three-dimensional (3D) objects to create curves, surfaces, or solids.

COLLADA

COLLADA is a COLLAborative Design Activity for establishing an interchange file format for interactive 3D applications without loss of data. COLLADA defines an open standard XML Namespace and database schema.

CRM

Customer Relationship Management describes methodologies, Internet capabilities and software applications that companies use to maintain relationships with customers in a structured way.

DAE

Digital Asset Exchange is the filename extension (.dae) that COLLADA documents use for describing digital assets information.

DCC

Digital Content Creation is a term that is used for the creation and modification of digital content, such as animation, audio, graphics, images and video before it is delivered in its final output.

DiCE

mental images's Distributed Computing Environment (DiCE).

DWF

Design Web Format is a secure file format developed by Autodesk® for the efficient distribution and communication of rich design data for viewing, reviewing, or printing design files.

ECMA

Ecma International (Ecma) is an international, private (membership-based) non-profit standards organization for information and communication systems. For more information, go to: www.ecma-international.org/.

H.264

H.264/MPEG-4 AVC is a standard for video compression. H.264/AVC is the latest block-oriented motion-compensation-based codec standard developed by the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG).

imatter

With the imatter module of the RealityServer software platform, you can interactively create and modify geometric shapes in RealityServer applications remotely from any Internet connected device capable of running a browser. See also: mental mesh™ Functional Overview.

iray

iray is the world's first interactive and physically correct, photorealistic rendering solution. With iray, users can quickly create life-like images of their creations by using intuitive, real world approaches and interactively exploring their results through the processing power of NVIDIA® graphics processing units (GPUs). See also: iray® Photorealistic Rendering Technology.

JSON-RPC

JSON-RPC is a remote procedure call protocol encoded in JSON. It is a very simple protocol that allows for bi-directional communication between the service and the client. It also allows multiple calls to be sent to a peer, which may be answered, out of order.

mental mesh

mental mesh is a software library providing the most efficient 3D geometry compression algorithm currently available. It allows for large amounts of data reduction in a variety of applications, such as the transmission of 3D data on mobile devices. See also: www.mentalimages.com/products/mental-mesh.html.

OBJ

OBJ is a family of declarative "ultra high-level" programming languages.

PLM

Product Lifecycle Management is a systematic approach to managing the entire lifecycle of a product from its conception, design and manufacture, to its service and ultimate disposal.

RIA

Rich Internet Applications are Web applications. RIAs have most of the features of desktop applications but they are usually delivered by way of standards based Web browser plug-ins, or delivered independently by way of sandboxes.

RTMP

Real Time Messaging Protocol is a proprietary protocol developed by Adobe Systems Incorporated. RTMP is used for streaming audio, video and data content over the Internet, between a Flash player and a server.

SaaS

Software As A Service. A model of software delivery where customers access applications remotely using the web. Operations are managed from central locations, relieving customers of the accessibility, maintenance, and upgrade issues related to locally installed applications.

SOA

Service Oriented Architecture defines the use of common services that are used by middleware such as RealityServer to implement processes. SOA was formerly known as Distributed Objects Architecture (DOA).

SOAP

SOAP or Simple Object Access Protocol is a protocol specification for exchanging structured information in the implementation of Web Services in computer networks. SOAP uses XML as its message format. SOAP usually relies on other Application Layer protocols such as Remote Procedure Call (RPC) and HTTP to transmit data.

REST

Representational State Transfer is a mode of software architecture for distributed hypermedia systems such as the World Wide Web.

