SunSoft, Inc
2550 Garcia Avenue
Mountain View, CA 94043

Solaris ONC+: Network Information Service Plus (NIS+): An Enterprise Naming Service, Part Number 92245-001, SunSoft Inc., Mountain View, CA

Project DOE: Distributed Objects Everywhere, Part Number 91035-0, SunSoft Inc., Mountain View, CA

The ToolTalk Service: An Inter-Operability Solution, Part Number ISBN 013-088717-X. SunSoft Press/Prentice Hall, Englewood Cliffs, NJ

ToolTalk and Open Protocols: Inter-Application Communication, Part Number ISBN 013-031055-7, SunSoft Press/Prentice Hall, Englewood Cliffs, NJ (June 1993)

The Common Object Request Broker: Architecture and Specification, OMG

Request for Technology: Multimedia System Services, Version 2.0, Interactive Multimedia Association, December 1992.

The Power of Multimedia, A Guide to Interactive Technology in Education and Business, Interactive Multimedia Association.

Recommended Practices for Interactive Multimedia Portability, Interactive Multimedia Association.

The Solaris XGL Graphics Library, SunSoft, 1993.

The Solaris XIL Imaging Library, SunSoft, 1993.

Solaris VISUAL Technical White Paper, SunSoft, 1993.

References

Solaris OpenWindows: OpenWindows V3 Collection: Release Reports and White Papers, Part Number 91021-0, SunSoft Inc., Mountain View, CA

Solaris SunOS 5.0: SunOS 5.0 Multithreading and Real-Time, Part Number 91025-0, SunSoft Inc., Mountain View, CA

Solaris ONC: Design and Implementation of Transport-Independent RPC, Part Number 91028-0, SunSoft Inc., Mountain View, CA

Solaris SunOS: SunOS 5.0 Release Report, Part Number 91023-0, SunSoft Inc., Mountain View, CA

Multithreading and Real-Time in Solaris: Terms and Concepts, Part Number 91024-002, SunSoft Inc., Mountain View, CA

The ToolTalk Service, Part Number 91022-002, SunSoft Inc., Mountain View, CA

Introduction to the ToolTalk Service, Part Number 91031-002, SunSoft Inc., Mountain View, CA

Solaris OpenWindows: ToolTalk in Electronic Design Automation, Part Number 91032-0, SunSoft Inc., Mountain View, CA

Tool Inter-Operability: A Hands On Demonstration: A Simple Demonstration of How the ToolTalk Service Works, Part Number 92205-001, SunSoft Inc., Mountain View, CA

Solaris ONC: Network Information Service Plus (NIS+), Part Number 91027-0, SunSoft Inc., Mountain View, CA

OMG

Object Management Group. A consortium of companies committed to establishment of standards for object-based computing.

ONC

Open Network Computing. An industry standard distributed computing model and implementation developed and openly licensed by Sun Microsystems, Inc. ONC is a model for heterogeneous, distributed client-server computing.

ORB

Object Request Broker. A technology specification developed by the Object Management Group for distributing objects across a network.

MPEG

Moving Picture Experts Group. This group has developed standards for compressing moving pictures and audio data and for synchronizing video and audio data streams. The MPEG standard is similar to CCITT H.261, with compression rates in the range of 1-to-1.5 Mbits/second. MPEG images are 352 x 240 pixels.

RPC

Remote Procedure Call. An interprocess communication mechanism for coordinating the use of a resource among many users. processes or machines.

SONET

Synchronous Optical Network. The ANSI standard for a fiber optic transmission system that consists of a specification of a hierarchy of transmission rates, ranging from 51.5 Megabits per second up to 2.4 Gigabits per second.

TCP

Transmission Control Protocol. The Internet standard transport level protocol that provides a reliable, full duplex, end-to-end transport network service.

Thread

A lightweight process that shares an address space with other threads of execution for an application.

WORM

Write Once Read Many optical storage technique that enables information to be written to a disk but prevents further change or erasure.

Interactive Video

Combining video and computer technology so the user's actions determine the sequence and direction the application takes.

IP

Internet Protocol. The Internet protocol standard that defines the Internet datagram as a unit of transmission passed across the Internet and provides the basis for the Internet connectionless best effort packet-delivery service.

ISDN

Integrated Services Digital Network. A set of digital network interface standards consisting of a signalling channel and a number of 64 kbps digital transmission channels that are used to provide circuit switched connections.

ISV

Independent Software Vendor. A independent company whose core business is the development and marketing of applications software.

I/O

Input/Output. The process, interface or data associated with inputs and/or outputs to a computer system or program.

JPEG

Joint Photographic Experts Group. A joint effort of the CCITT and ISO that developed a standard for compressing greyscale or color *still* images. Actually, the standard defines a number of methods for compressing images. Several of these are methods based on the Discrete Cosine Transform (DCT), and one method is lossless and is based on Differential Pulse Code Modulation (DPCM). JPEG compression can be used for moving images, but does not take into account the frame-to-frame repetition of moving images.

LAN

Local Area Network. A type of network allowing interconnection of computer devices within a relatively restricted area, such as within a building or a compact group of buildings.

MAN

Metropolitan Area Network. A relatively new, extended kind of LAN technology, where local networks connect via high-speed "connections" to other local networks. The higher performance characteristics can be the result of any of a number of emerging technologies including FDDI, ATM, SMDS and others.

Media Object

An abstraction of the data and facilities provided by the Solaris LIVE multimedia environment.

Full Motion Video

Video production at 30 frames per second for NTSC or 25 frames per second for PAL. The showing of a series of related digital images at a rate sufficient to give the illusion that objects in the images are moving naturally.

GUI

Graphical User Interface. A visual metaphor that uses icons representing desktop objects which the user manipulates with a pointing device.

Hypermedia

An extension of hypertext that uses various types of media in addition to text. The information is organized in a manner such that the user can easily navigate through it.

Hypertext

Information linked together through multiple paths allowing the user to cross-reference related objects in a natural manner.

IDL

Interface Definition Language. The language defined by the OMG for describing interfaces in the Object Request Broker. IDL is programming language independent.

Indexed Color

The distinction between true color and pseudocolor has to do with the design of a monitor's frame buffer. If the frame buffer uses 8 bits per pixel to store color information, the monitor can display 256 colors simultaneously. What you see on such a monitor is called pseudocolor because the colors that can be shown at any one time are a small subset of the colors the eye can distinguish. If the frame buffer uses 24 bits per pixel to store color information, the monitor can display over 16 million colors (true color). Pseudocolor is sometimes called indexed color because the values stored in the frame buffer on a pseudocolor system are not the RGB values needed to drive the red, green, and blue electron guns in a monitor. Rather, they are indexes into a colormap, or color lookup table, which stores 256 sets of RGB values.

IMA

Interactive Multimedia Association. With over 250 members, the IMA is an international trade association organized to promote the benefits of multimedia technology. The IMA is committed to this goal through enhancement of the growth of the industry through public education, the development of specifications for hardware, software tools, and applications, the development of industry-wide services, and providing government and media relations.

Interactive Multimedia

The capabilities which provide a human user with real-time control of time-based media.

CD-ROM

Compact Disk Read Only Memory. An adaptation of the Compact Disk Digital Audio for use in general computer use. CD-ROMs are laser readable optical storage media typically holding approximately 600 Megabytes of data.

Cell Encoding

A video compression algorithm, which was developed by Sun. In Cell encoding, a 4-by-4 region of pixels is represented by two colors and a 16-bit mask that indicates which of the two colors to place at each of the 16 pixel positions. The colors and mask are chosen to preserve the mean and variance luminance and the average chrominance for the 4-by-4 block. Cell decoding takes advantage of the fonting hardware commonly found in bitmapped display accelerators.

CODEC

COder/DECoder or COmpressor/DECcompressor: A mechanism that can encode and decode, or compress and decompress, a media type. CODECs are frequently employed in the manipulation of audio and video data. Their most common implementation is in hardware.

Compressed Image Sequence (CIS)

Compressed Image Sequence. The XIL library's compressors store (generally related) compressed video frames in structures called CIS buffers. The images may represent frames in a movie, pages in a document, and so on. The data in the image sequence may or may not have undergone compression. If the data is compressed, it may be in Cell or JPEG. The XIL library provides functions that enable users to select frames in a buffer, decompress the frames, store frames into the buffer, and get information about the stored frames.

Compression

The process of converting data from its original format to a format that requires less storage and can be transmitted more quickly. Image data can be compressed from one-tenth to one-fiftieth its original size without visibly affecting the quality of the image.

Digital Audio

Audio represented by machine-readable binary numbers rather than analog recording techniques. Analog audio is converted to digital by sampling the audio signal at a defined rate (typically 4000 samples per second or greater).

Digital Video

Video information stored as numerical data.

FDDI

Fiber Distributed Data Interface. An ANSI standard for 100 Megabit per second local area networks typically based on optical media (cable).

Glossary

API

Application Program Interface. The programmatic software interface to which an application developer writes an application for access to the features and functionality of the software module represented by the API.

ATM

Asynchronous transfer mode. A high bandwidth network protocol based on packet switching.

Baseband

A network where the bandwidth is taken up by a single digital signal. Examples include Ethernet and Token Ring.

Broadcast Video

A video rate of 525 lines displayed at 60Hz (US) or 625 lines displayed at 50Hz (Europe).

CCITT

International Telegraph and Telephone Consultative Committee. The international association and standards body composed primarily of representatives from national telephone agencies. The CCITT promulgates telephony standards, such as X.25, the Group 3 FAX standard, and the H.261 video conferencing standard (Px64).

CCITT Group 3 Standard

A standard that specifies how facsimile machines must compress and decompress image data. The compression method relies heavily on run-length encoding. Runs of white pixels and runs of black pixels are represented with codes from a Huffman table.

sequences, Sun raster and text. Finally, the header includes information on the media rate, which defines the individual edit samples which represent the smallest interval of time recognized on the track.

Media rates are not necessarily equivalent to sampling rates. For example, audio data might have a sampling rate of 44.1KHz (44,100 samples/second), but might be edited in conjunction with video data at a media rate of 29.97 frames per second.

Solaris LIVE! and Project DOE

Solaris LIVE! has been developed to be conformant with SunSoft's future distributed object environment, DOE. DOE describes the Solaris-based object environment, which utilizes the Distributed Object Management Facility (DOMF) - SunSoft's implementation of the Object Management Group's Object Request Broker (OMG ORB) architecture.

Solaris LIVE! Framework's object oriented interface is compatible with the model employed by the DOMF. The framework is designed to migrate to the DOMF for managing its objects in a network distributed fashion.

Solaris LIVE! and The IMA

The Solaris LIVE! Framework is being developed by SunSoft to be conformant to eventual recommended practices being defined by the IMA Services Technical Working Group. This framework is based on work done by SunSoft in response to the Request for Technology (RFT) process currently underway by the IMA. Key industry vendors (IBM, HP, SunSoft, SCO, Univel and USL) have committed to responding to the RFT as part of their alliance to deliver a Common Open Software Environment, which includes multimedia services. This work, in conjunction with other IMA activities and recommended practices will result in clear, definitive standards for cross-platform multimedia APIs, data formats, and services.

Virtual devices, in turn, may perform resource management either by communicating with other virtual devices directly or by using a central clearinghouse for resource usage.

Applications need not have any notion of what physical resources are required by a virtual device. Applications may merely ask to use the virtual device, and the virtual device implementation compares priority information to decide how to allocate resources.

Data Services

At a higher level, it is often desirable to manage data with a file format, specifying the structure of the data together with accompanying information which describes how to interpret it. The data file formats reduce the development burden on the programmer by providing an effective means to support common data formats without the burden of writing code to produce them or manage conversion.

Data services are provided through the Solaris LIVE! framework in the form of media and data file formats supported by bundled virtual devices, made available through Solaris LIVE! and Solaris VISUAL APIs.

Media formats consist of a bit stream representation with defined characteristics for encoding, compression type, sample rate, bits per sample, et cetera. The Solaris LIVE! Framework will support all of the formats defined in Solaris LIVE! today with the addition of new formats as they are defined or endorsed by the IMA.

In addition to the individual media types supported by Solaris LIVE! today, the Solaris LIVE! Framework will also provide support for a composite file definition.

Composite Files

Composite files are an orchestrated composition of multimedia files, together with the information which describes how to store or play them. A simple example of a composite file would be an audio and video stream which require synchronized play. A more complex, but equally likely example, is a group of audio tracks which are to be mixed prior to their replay.

Composite files can combine a large number of media types into its composite format. Audio, video, image and text data are represented as media classes which are referred to via the composite file header. This header also contains information regarding the specific media type equating to format (and encoding), such as Sun audio, image

Resources

In the Solaris LIVE! framework, any source, destination or processor of multimedia data that is to be shared between multiple virtual devices is called a resource. Examples of source resources are information gathering objects and devices such as cameras and microphones. Destination resources can be windows, speakers and other input-oriented objects and devices which utilize input for their operation. Processor resources are generally compute-intensive objects or devices accepting input and producing output, such as a CPU-based text-to-speech, speech recognition, media data conversion and compression programs.

Resources often have need for serialized sharing, due to specific hardware or technology dependencies (e.g. a video tuner may provide only one channel at a time) or due to software or policy limitations such as licensing restrictions or numbers of simultaneous users.

Resource Management

The Solaris LIVE! Framework defines a resource management interface used to control shared access to multimedia resources (e.g. devices). Using a common interface simplifies the programmer's view of resources. The resource management interface provides a standard way for virtual devices to observe, reserve, acquire, release, and preempt access to shared resources.

Resource management concerns itself with how applications get, relinquish, and take over non-sharable resources such as speakers, compression processors, phone lines, or other software and hardware devices. Resource management is also concerned with how applications share resources.

Solaris LIVE! allows the management of access to devices by providing a way for applications to set "priority-of-use" for the resources they need. These priorities are based on the following parameters:

- The importance of an application obtaining the resource
- The importance of an application retaining a resource
- The importance of releasing the resource in a timely manner

Solaris LIVE! uses a "central control" model for resource prioritization. In this model, applications do not communicate directly with each other to share resources. Instead, each application tells a virtual device its required terms for obtaining and releasing a resource, as well as when to transfer the resource from one application to another.

Virtual Connections

The most important aspect of the Solaris LIVE! framework is the ability to move multimedia data efficiently from one resource to another. From an application's point of view, this is accomplished by establishing a virtual connection between two virtual devices.

The virtual connection provides an application interface for controlling the flow of data between two virtual devices while maintaining a specified rate. The virtual connection manages the processing of data movement to achieve the desired rate. When adequate transport bandwidth is available, this is straightforward.

The virtual connection is especially important when time-critical data movement cannot be delivered at the desired rate (due to competition for CPU or network bandwidth). As bandwidth, and therefore, data rate for a stream is reduced, a virtual connection is empowered to do what is necessary to gracefully recover. The virtual connection may be instructed to reduce the quantity (and hence the quality) of data for the stream, allowing the stream to be processed faster. A more complex virtual connection may delete, degrade, or temporarily stop data for one stream (or otherwise redistribute transport bandwidth) to speed another stream and thereby keep the streams synchronized.

Time-critical services are managed as part of the Solaris LIVE! framework through the use of the virtual connections connecting virtual devices to one another and to devices. Several factors control the bandwidth of data movement, including data type, rate and synchronization interval (for multiple data streams). Time-critical services are layered above the networking transport layers, thus permitting alternate types of networks such as Ethernet, FDDI, ATM, etc.

Virtual connections ask the "ports" through which data moves about their data rate capability. A port is an input or output to a virtual device. For example, a video player is a source virtual device with a single output port, able to provide data in a defined format and at a specific data rate. The data format might include a compression or encoding type, which in turn affects the data rate. A destination virtual device which has one input, such as a disk (file) virtual device or a video recorder, will also have attributes for data type and recording rate.

This interface defines the methods that access the object (syntax and function) as well as expected behavior (semantics) of the virtual device object. The implementation of the interface to the virtual device is available through a client-side library.

Using the published interface, an application program creates a virtual device object and invokes its methods. The object that the programmer sees in the application's address space may provide the implementation for some or all of the published behavior. The local object (virtual device) is just a "stub" that creates a remote *service virtual device object*. Figure 4.3 illustrates this concept for an audio application.

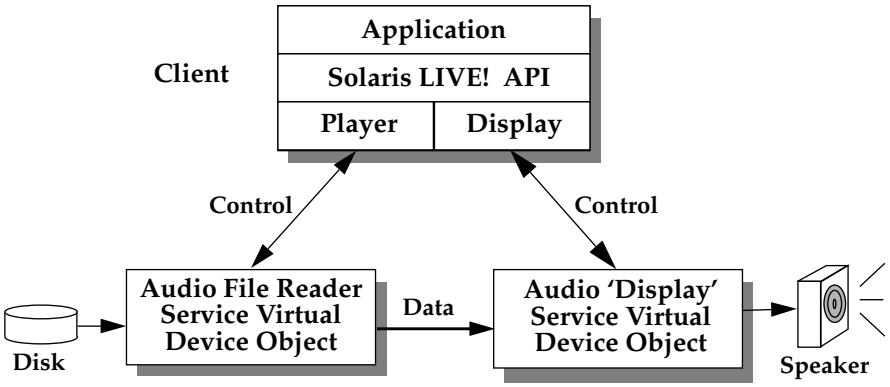

Figure 4.3 Example of an Audio Virtual Device with Player Controlling 'Display'

Controllers

The runtime creation or instantiation of a service virtual device object is handled by a controller. Each controller is associated with a collection of virtual devices (and possibly resources) that it will manage. The existence of a controller is not explicitly exposed to the application programmer. The client-side API code contacts the appropriate controller and handles the protocol for instantiating a virtual device.

A controller process may provide the execution environment for the virtual devices it manages. In that case, the virtual device(s) shares the same process, code and data. Each virtual device may be just the particular "context" seen by a client. In such cases, the context may be just a simple data structure, or it may be the full context of an execution thread. Typically the controller gives each virtual device its own address space.

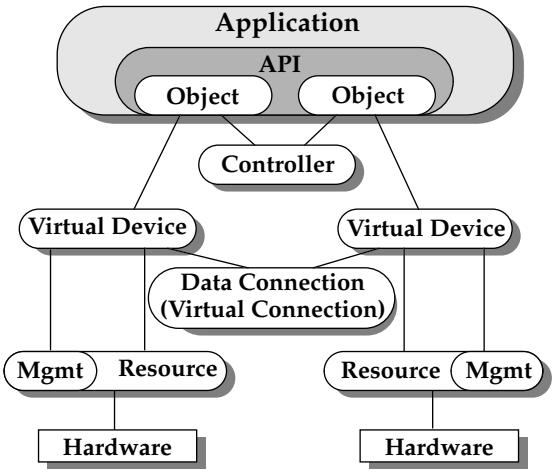

Figure 4.2 The Solaris LIVE! Architecture

Virtual Devices

The basic unit in the Solaris LIVE! framework is the *virtual device*. Virtual devices are the programmer's interface for the access, control, and processing of multimedia data.

The application programmer instantiates virtual device objects. Each virtual device object represents an abstraction of a hardware or software resource that the application wants to control. The application interacts with the object by invoking methods on the virtual device object.

The execution environment of a virtual device is typically on the same host as the resource it represents. But, the application controlling that virtual device may be on a different host. Negotiation for contention of shared resources happens both at virtual device object instantiation time and dynamically over the life of the virtual device object. Policies for handling resource allocation are encapsulated in the virtual device object interface.

Virtual devices are nonpersistent objects; they exist while a client application is using a resource. The persistent aspect of a virtual device is the description of its interface and its capabilities. Each Solaris LIVE! virtual device conforms to a published interface (the virtual device's API) which is accessible to Solaris LIVE! applications.

streams between multimedia devices. The framework also allows the sharing of data and devices among cooperating processes and ensures security and privacy when necessary.

Architecture

Traditional distributed computing models provide for the sharing of resources via server processes. Servers in this context provide network-accessible protocols and access to them via defined APIs. Servers generally exist as network resources and are capable of simultaneous communication with multiple clients.

While servers provide an effective means of distributing processing, they do not address the need of applications which want to efficiently share data between various services. For example, when one service generates data, the application must deliver it to another service for the processing or display - there is no server-to-server data path without the application.

To meet the integration needs of multimedia applications, the functions of a server (e.g. resource management, execution of client requests, network presence, etc.) can be broken into various components. The APIs can help clarify interfaces and makes replacement of functionality easier.

In the Solaris LIVE! framework:

- A multimedia data or service-providing object is called a *virtual device*
- The persistent network presence is encapsulated in a *controller*
- Data flows between applications via *virtual connections*
- Common *resource management* interfaces are used by all Solaris LIVE! clients

The Solaris LIVE! Framework is a refinement and extension of a distributed computing architecture which provides for common services and interfaces for distributed multimedia. Solaris LIVE! is based on object oriented interfaces that largely hide the details of its distributed nature. Most importantly, Solaris LIVE! provides time regulated, high-bandwidth transports for continuous, time-critical data. The use of resources, especially transmission bandwidth, is carefully controlled, or scheduled, to optimize performance and provide graceful degradation when demand exceeds available capacity. Figure 4.2 illustrates the Solaris LIVE! architecture and the relationship between components.

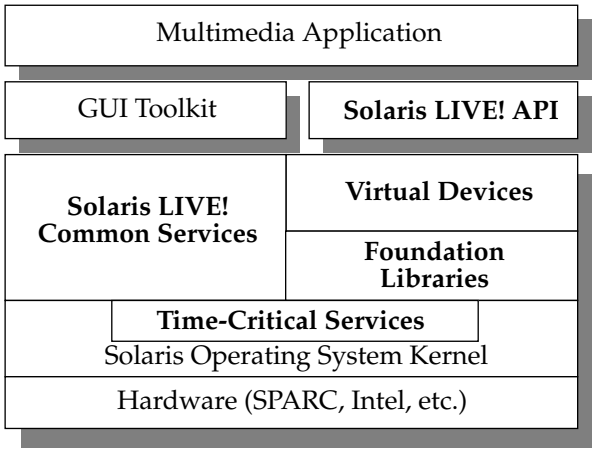

Figure 4.1 The Solaris LIVE! Framework

Application programs access the functionality of Solaris LIVE! using a high level application program interface: they are not required to understand the details of the model or its implementation.

The principal goals of the Solaris LIVE! Framework are:

- Integration of multiple media types within an application
- Provide support for time-critical continuous media transport
- Distribution of data and computation over networks
- Sharing of resources by multiple applications and users
- Provide security of data and resources
- Provide an object oriented interface consistent with DOE
- Provide common API and programming model for different media
- Provide common and consistent way of managing multimedia resources in a device-independent manner
- Synchronization of different media data streams
- Interoperability through IMA recommended practices

The Solaris LIVE! framework is based upon a model of distributed objects. The various objects cooperate to provide access to multimedia resources. The framework defines the means to control and synchronize the flow of real-time continuous data

The Solaris LIVE! Framework 4

The Solaris LIVE! Framework

The Solaris LIVE! Framework heavily leverages the API capabilities already present in Solaris while extending it to support key needs of distributed multimedia.

The goal of the Solaris LIVE! Framework is to assist developers in migrating to the full spectrum of multimedia services. Existing applications using Solaris LIVE! facilities today will run with minimal change under Solaris LIVE! in the future. Figure 4.1 illustrates the Solaris LIVE! Framework APIs and services.

The Solaris LIVE! Framework consists of a a set of common services, a set of APIs, and associated multimedia services, including:

- Media Handling Objects (known as *virtual devices*)
- Time-Critical Services
- Data Services
- Resource Management Facilities

The Solaris LIVE! Framework provides a programming environment for media-independent services which can be used to build multimedia-aware applications and interfaces.