Common System Plane

EION Open IP Environment Common System Plane is one of the four planes that comprise the Open IP Environment Framework. This plane delivers operating system independence for Open IP Environment applications and is designed to allow system integrators and developers to build portable Internet Protocol (IP) enabled products.

Overview

EION Open IP Environment is a portable real-time software suite that IP-enables new and traditional network elements providing high performance interoperability across multiple platforms and products. Open IP Environment is based on a single, open, modular and scalable framework that allows system integrators and developers to incorporate services such as routing, Quality of Service (QoS), security, IP accounting and policy management into any type of device. Open IP Environment is platform and real-time operating system (RTOS) independent and can work on any type of device ranging from high end optical core switches to personal digital assistants (PDAs).

Framework Overview

EION Open IP Environment framework consists of four planes: Common Control Plane, Common System Plane, Common Forwarding Plane and Common Management Plane. Each of these planes contains a set of components that are built to use well-defined interfaces.

The Common System Plane is one of the planes comprising the framework and its main functionality is to provide operating system independence.

Common System Plane Overview

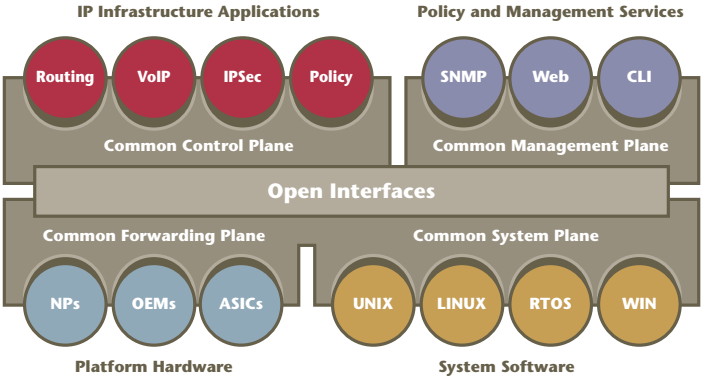
The Open IP Environment Common System Plane delivers operating system independence for Open IP Environment applications and allows system integrators and developers to build IP enabled products. This plane is designed to aid the development of high-performance services for communication software on different platforms such as UNIX and RTOS.

Open IP Environment Common System Plane simplifies the development of network applications and services that use inter-process communication, de-multiplexing, thread management, logging and concurrency.
This plane's functions can be divided into Run-time Environment Capabilities and Run-time Environment Services to showcase the available implementations through its well-defined application programming interfaces (API).

Run-time Environment Capabilities and Run-time Environment Services of Open IP Environment Common System Plane

Run-time Environment Capabilities of Open IP Environment Common System Plane provide platform-independence low-level services to other planes and modules through the Operating System Abstraction Layer (OSAL) and the thread manager.

Run-time Environment Services of Open IP Environment Common System Plane provide an execution environment for high level software by being equipped with software libraries that include a reactor for event de-multiplexing, configuration management, logging, memory management, synchronization, hashing, messaging and timers. Run-time services are a set of portable system components that guarantee common behavior across supported platforms as they are independent of the services provided by the target OS.



Runtime Environment Capabilities

OSAL

OSAL delivers a layer of abstraction that ensures platform independence for all Open IP Environment software modules. OSAL provides Posix compliant APIs that allows Application Developers to write portable programs, therefore reducing the learning time for new classes and maintaining system performance. In the event where the underlying system does not support Posix, OSAL is designed to map Posix calls into native function calls of the selected operating system.

Thread Manager

Thread manager is an enhancement to the threading facility of many operating systems residing in the framework as it manages threads at group levels by suspending and resuming, sending signals, and waiting on a group of threads. The thread manager also provides a configuration mechanism to permit dynamic configure or reconfigure mechanisms of thread creation parameters at run-time by implementing fault resiliency at the thread level through the watchdog facility. This watchdog facility permits the registration of threads for monitoring for the thread manager by checking the status of each registered thread.

The thread manager is configured to force the threads to "punch" themselves periodically to prevent them from being timed out. Furthermore, the thread manager features restart mechanisms that allow a thread to restart automatically after it is terminated. In the situation where a thread terminates, the thread manager will invoke the thread's clean-up method to release appropriate resources through a well-defined interface that registers clean-up routines of individual threads.

Run-time Environment Services

Reactor

Reactor is an object behavioral pattern for efficient event de-multiplexing and dispatching of synchronous events. It acts as an event broker that dispatches defined handlers for events that are delivered concurrently to the application by one or more clients. The reactor can be used in a variety of ways, for example reactor can be used in the publisher-subscriber model.

The reactor for event de-multiplexing and dispatching provides the following enhancements that simply the development of event-driven applications in the Open IP Environment Common System Plane:

- Provides a uniform interface for event de-multiplexing and dispatching: For example, the reactor handles the following types of events: I/O events, timer events, and user-defined events.
- Automates event handler dispatching in the situation where one or many activities occur on handles managed by Reactor. The reactor then automatically invokes the appropriate event handling methods of the pre-registered event handlers.
- Eliminates common error-prone programming details by shielding application developers from programming low-level OS event de-multiplexing calls such as select, setting and clearing bit masks, and detecting and responding to interrupts.

- Reactor is designed to shield applications from the underlying event de-multiplexing mechanism, therefore enhancing portability.
- Designed to ensure thread safety. For example, multiple threads can safely share a single Reactor and in turn, multiple Reactors can run in separate threads within a process.

Configuration Management
Configuration Manager provides a common interface for dynamic configuration of the registered applications (i.e. configured objects) at run-time. It supports both static and dynamic configuration of different object attributes. An example of configurable attributes are IP address, network mask, etc. The Configuration Manager supports configuration through default values, configuration files and command line interface.

Logging
Event logging in the Open IP Environment Common System Plane is used as the initial source of information for detecting and diagnosing system errors. The logging framework simplifies the log generation process and permits formatted log messages to be generated without performing I/O in the application thread's context. The logging framework also supports notification to subscribers of log events, where it acts as a simplified version of an event handler executing user-defined action. The logging component also performs log message filtering based on layering, and logs message outputting to various media such as a disk file, console etc.

Memory Management
The Open IP Environment Common System Plane provides a portfolio of dynamic memory management schemes including the management of pre-allocated chunks of memory, as well as a more flexible memory pool manager. The Common System Plane memory manager permits a more predictable performance of application routines.

The basic memory management scheme allows applications to pre-allocate chunks of dynamic memory from the operating system by controlling the allocation and de-allocation of the individual block internally. At the same time, the memory pool manager scheme offers management of four pre-allocated lists of memory blocks of varying sizes, and in turn, uses the best-fit strategy for allocation of memory to the application.

These memory management schemes have been designed to support the allocation of increased memory block in the event of lists becoming exhausted.

Hashing
The Open IP Environment Common System Plane supports a portable and very efficient (i.e. O (1)) hashing utility with a small footprint, for inserting, retrieving and erasing data elements of any type. The Hash Table is a reusable, and thread-safe chained hash container that permits the thread safety mechanism to be turned off to drastically reduce overhead for the customer. By implementing the Hash Table in a small and non-template base class will prevent template-bloat problems that may manifest themselves in multiple instances and multiple copies in the memory, and in different object files of the customer's device.

Synchronization
The Open IP Environment Common System Plane supports different locking mechanisms, which achieves synchronization. These locking mechanisms include mutexes, semaphores, read/write mutexes, condition variables, and tokens with unified interfaces. An operating system often supports a subset of these synchronization schemes, and Open IP Environment's synchronization can provide emulation services when required. Emulation services deployed by the synchronization will ensure that a uniform interface is achieved for the different synchronization mechanisms that are also compatible to third party operating systems.

Messaging
Modern real-time applications are constructed to use independent tasks that communicate with each other using messages. The most common method of message-based communication is with a message queue. The Open IP Environment Common System Plane supports message queues with the following attributes:

- Provides simplified priority queuing when high priority messages are de-queued before lower priority messages
- Achieves high performance flow control when lower priority messages are discarded to permit queuing of higher priority messages in the instance where the message queue is full
- Allows an accessible message queue with multiple threads for full control by the network operator
- Adapts to the user's application by providing a suitable priority queuing algorithm.

Timers
The Open IP Environment Common System Plane provides a timer facility to satisfy the needs of real-time applications and communication protocols. This feature is embedded within this plane to address the inadequacies of timer facilities supported by certain operating systems. Timers in the Common System Plane are provided by the callout facility with the following functions:

- Allows a function and an argument to be registered and called at a future time when the associated timer expires
- Supports granularity of microseconds
- Provides portability across platforms currently supported

Common System Plane Interactions

EION Open IP Environment Common Control, Common Management and Common Forwarding planes use the Common System Plane run-time environment capabilities such as OSAL and Thread manager in addition to the run-time environment services such as memory management, message queues and synchronization. These functions of the Common System Plane provide the other planes and modules with low-level system services to enhance portability to different platforms.

Common System Plane Features

EION Open IP Environment Common System Plane demonstrates the following key features:

- Lowers total cost of ownership by enabling large-scale extensibility of software
- Improves accuracy with a "type-safe" object oriented interface
- Allows for "faster time to market" by de-coupling application-independent components from application-specific components
- Achieves scalability through Common System Plane Application Programming Interface (API) for these services:
  - Operating System Abstraction Layer (OSAL)
  - Thread Management
  - Reactor for Event De-multiplexing and Dispatching
  - Configuration Management
  - Logging
  - Memory Management
  - Hashing
  - Synchronization
  - Messaging
  - Timers
- Enhances portability by means of OSAL and platform independent interface which uses OO features
- Provides high performance, reliable and tested components

Ease of Portability

EION Open IP Environment provides a set of interoperable modules that are available for use in both established and "greenfield" products. The customer has the choice to pick and choose Open IP Environment modules to incorporate into the customer's established products, preserving the investment in prior development. The customer also has the option to use modules within the Open IP Environment framework to develop a new software base to address going-forward opportunities. It is also possible to compile the software for a variety of target processors. Therefore, protocol composition can be statically changed by modifying the configuration to suit your needs.

Established products typically have a well-developed architecture and an existing suite of applications, and these products will be looking to Open IP Environment for additional capabilities. The portable and modular Open IP Environment components can be integrated into an existing execution environment to work within an existing code base, with minimal modifications to the customer's environment.

Greenfield products typically require a full suite of applications plus the Open IP Environment framework to provide an appropriate execution environment. The Open IP Environment framework and modules are well-positioned to address such greenfield opportunities.

Benefits

In a market that demands ever-increasing IP support, it is difficult to maintain sufficient in-house expertise in every area. EION Open IP Environment framework and the Common System Plane solve this problem by:

- Allowing OEMs to focus on their real value added solutions, not underlying infrastructure
- Reducing the length of time to market via ease of integration of key components such as the Common System Plane
- Enabling the freedom to choose among different software and hardware platforms
- Enabling ease of portability to traditional and new network enabled devices
- Enabling accelerated development of highly customized IP-enabled products via well documented APIs
- Enabling a pick and choose approach to Open IP Environment modules via a flexible open framework addressing various devices and applications from PDAs to carrier grade optical switches
- Delivering components of the framework that are scalable, modular, and portable that consistently demonstrate high performance attributes
- Delivering standards-based interfaces and common programming languages such as C, C++ and Java to developers, enhancing overall productivity with a small learning curve.
- Delivering configured and managed modules that use one or several of the following management capabilities:
  - EION Command Line Interface
  - Simple Network Management Protocol (SNMP)
  - Web-based management.

EION Inc. Locations Worldwide

United States
EION Inc.
CT Corporation System
101 Federal Street
Boston, MA 02110
United States
Ph: 613-715-9067 x224
email: global_sales@eionsoft.com

Asia Pacific
EION Inc.
Room 1405, 14/F
China Merchants Building
No. 303 Des Voeux Road
Central, Sheung Wan
Hong Kong, SAR, China
Ph: +852 9314 3023
email: asia_sales@eionsoft.com

Canada
EION Inc.
945 Wellington Street
Ottawa, Ontario K1Y 2X5
Canada
Ph: 613-715-9067 x224
Fax: 613-722-0039
email: global_sales@eionsoft.com

Europe, Middle East & Africa
EION Inc.
Claridge House
29 Barnes High Street
London SW13 9LW
UK
Ph: +44 (0)20 8741 5377
email: europe_sales@eionsoft.com