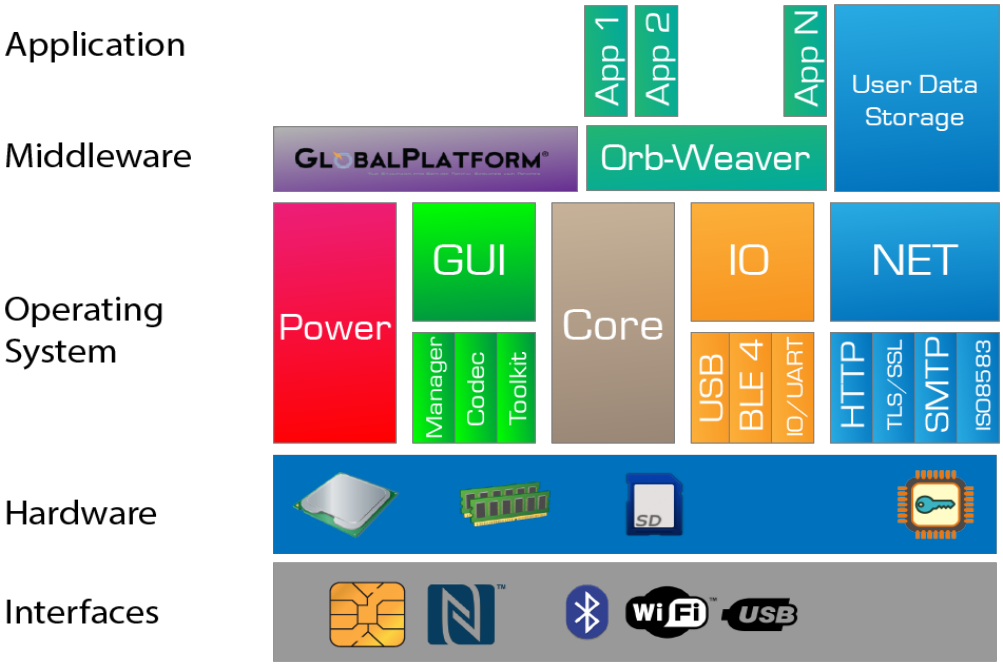
Leaf Operating System



Overview

Leaf OS was designed to run on mid to high performance micprocessor with small memory size, it was intended as host operating system for Orb-Weaver virtual machine, user application can be dynamically install, execute and remove from Leaf OS similar with AppStore mechanism.

Description

Leaf OS is a compact GUI based operating designed to run on top of ARM Cortex-M Microprocessor, it uses single framebuffer for small memory footprint to render all UI objects, it also support user application installation and execution through a specialized virtual machine, it multitasking feature also support execution of several user application altogether.

For security purpose application installation are based on Global Platform mechanism, which the application package transfered through a secure channel using a precomputed session key between application server and host platform.

Application are installed as virtual machine bytecode instead of native (using OrbLeaf's Orb-Weaver Virtual Machine), this is because ARM Cortex-M doesn't provide MMU for logical access for execution of multi-tasking native application on different address space, also to provide better portability for user application between different hardware platform (OrbLeaf does provide OSes for several hardware platform in which our virtual machine reside, π-CardOS for smart card and LeafOS for active hardware platform).

Native Operating System APIs are wrapped by virtual machine layer enabling invocation of native APIs from user application. GUI engine is integrated with image decoder for decoding PNG and JPG image during runtime, also can be used to create custom UI from user application.

Hardware Abstraction Layer (HAL) Auto-Config provide mechanism of identifiying host platform interfaces configuration automatically without having to re-build/re-configure the operating system sourcecode through enumerating it's pre-registered driver collection.

Virtual Machine provide exception handler which will terminate user application in-case of application error, garbage collection also available to prevent memory leakage caused from dynamic allocation by the virtual machine. In-case of system error, the operating system will generate a Red-Screen of Death, all current task executed stack trace are send to our error service for reporting and debugging purpose.

Network stack support different type of transport/session and application layer ranging from TCP, IP, UDP, SSLv3, TLS1.0, TLS1.1, TLS1.2 for secure session, SMTP (mail protocol), NTP (network time synchronization), ISO8583 (financial messaging protocol), NETBIOS (for peers discovery on LAN network, service automatically run when LAN network available on HAL) and HTTP/HTTPS.

Features

- Small Footprint, less than 1 MB, if you need update firmware feature you need to double the size of flash memory
- Multi-Tasking, pre-emptive based multi-tasking for system task and round-robin for user task
- Integrated Exception Handler, in-case of system error, OS automatically restarted
- Global Platform v2.2, for secure application installation
- Integrated GUI + Image Codec, load any PNG or JPG image for your custom UI (as long as the resource available it will be loaded by GUI engine)
- Integrated Orb-Weaver VM + APIs, application are run inside sandbox
- Automatic Error Reporting, in-case of error, a reporting service is transmitted to our server
- Integrated Cloud Service, download your application from our cloud service
- Auto-HAL Configuration, no need to use different OS binary for different hardware configuration
- Inter Process Calling (IPC), each task could send message one another, no need to block other task when waiting for IO.
- Network Stack, basic network stack for network operation.
- Auto-Application Synchronizaton, new application update are automatically installed without user confirmation.

Application Programming

User application programming for LeafOS can be done through OrbLeaf's Organ IDE, it uses JavaScript based programming language which makes development less painful compared with native C/C++, each variable are allocated dynamically by virtual machine, several features of our Organ IDE are :

- Javascript Syntax, easy migration for javascript developer
- Imperative Style, readibility also enabling collaboration between package, see our online library feature.
- Context Oriented, makes it easier to re-direct function according to each context
- Lazy Expression, who's doing i = i + 1 when you can i++
- Lambda Expression, declare, pass and execute your function everywhere on code
- Data-Type Extension, custom mathematical operation on specialized datatype
- Syntax Analyzer, if its not colorful then you're using the wrong language
- Intelli-Sense, learn faster by writing code, who need documentation anyway
- Device Simulator + Debugger, simulate your program first before deploying it
- GUI Designer, MVC based GUI designer
- Deploy Now, who need a JTAG/SWD or USB cable anyway when you have cloud
- Online Library, need a 3rd party library for easy integration to cloud service or device, search our library and add it to your project, 3rd party cloud/device integration as simple as calling a function.