Protect Your Software Investment

OSABSTRACTOR®

Design Better.
Reduce Risks.
Ease Upgrades.

Protect Your Software Investment

The Difficulty with Embedded Software Development

Developing embedded software is complicated. A software project has several critical elements contributing to its success. Issues with budgets, software tools, project schedules, hardware choices, and application requirements are just a few of the many elements involved in software projects. All of these factors are necessary, but not sufficient, for your project to finish successfully. Coordinating these project aspects and moving forward can be overwhelming, and missing one minor piece can make or break your product launch.

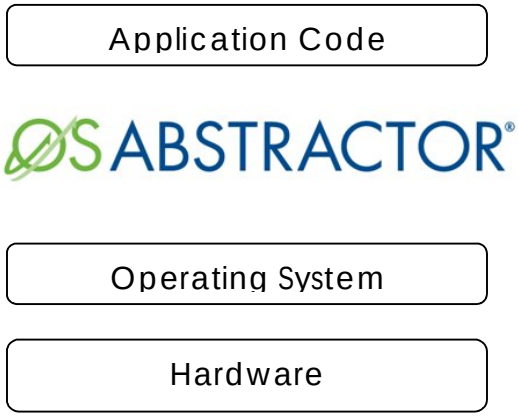
When you are choosing specific software tools for the project, there are several variables to consider. You must spend precious time evaluating the capabilities, verifying availability, confirming pricing, and making an overall decision for the fit of the tool. Significant time is invested in this process. You want time invested in evaluation to lead to an efficient and practical solution. A poor choice in tools could lead to project delays or a project cancellation.

One of the major software decisions you must make is the type of operating system (OS) that will be used in the project. The OS is a core component of the application design, and the OS capabilities must adequately fit the needs of the project. The choices surrounding the OS are numerous. You have over 100 different commercial OS vendors to evaluate. You must take into consideration many variables: What is the OS functionality? Has the OS been certified? What is the OS code size? How fast is the kernel? What is the interrupt latency? Are the price and business model right for my project? If any one of these variables goes awry, then your project has the potential for getting derailed.

By using OS Abstractor you gain a solid architecture for application development and eliminate the risk associated with the OS choice. By acting as a safety zone between your code and the application, the abstraction technology creates a seamless separator from the application code to the operating system, shielding you from any switching or upgrading costs with the operating system. OS Abstractor alleviates key issues surrounding your OS choice, including the support of new hardware. If you may need to move to another OS or potentially change hardware vendor during the life of your project, OS Abstractor should be a key component of your software design.

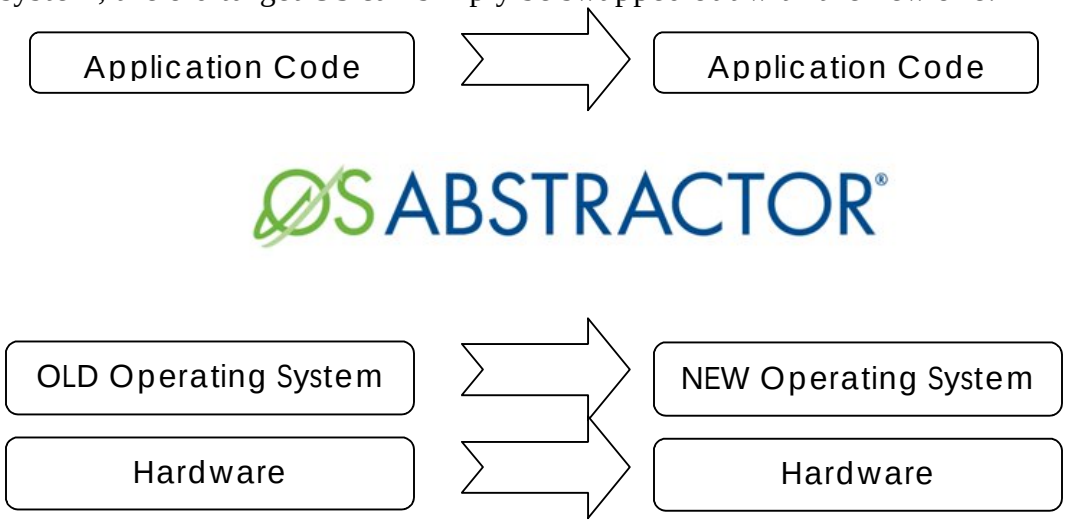
OS Abstractor - Overview

In software programming practices, defining the software architecture is a vital element for proper application design. A strong architecture simplifies future activities like adding new features, maintenance, bug fixing, and optimization. In a software project with OS Abstractor, your architecture will look like this:



Your application has a clear and distinct separation according to solid programming practices. Your application code still makes calls to the underlying operating system, but the untidy coupling between your application and the OS is eliminated.

Should you need to move your code to another operating system or upgrade to a new system, the old target OS can simply be swapped out with the new one:



Your application code is completely unaffected and you move your code to a new OS without modifying your application. Potential delays from unnecessary porting work are eliminated with OS Abstractor.

OS Abstractor – Design Philosophy

OS Abstractor provides an important level of abstraction to your application. It also provides performance enhancement features like re-using kernel resources and allowing optimization of the abstraction code that is specific to your application.

Many roll-your-own abstraction solutions add significant overhead to the application code. OS Abstractor's code size is minimal, with overhead as little as 10K depending on the architecture, tools, and operating system. Since the Abstractor product is written in C, it easily integrates into your existing C/C++ and Ada software projects.

Many home-grown products perform the abstraction by using a wrapper implementation. Wrappers are easy to implement and straightforward to understand, however they suffer from several problems. First, in-house wrapper methods do not always cover the majority of the operating system function calls. OS Abstractor provides strong coverage for the most popular functions used in the modern commercial real time operating system.

Another drawback of an in-house wrapper is that a proprietary solution only abstracts the function calls. OS Abstractor abstracts all aspects of the operating system, not just the function calls. The control blocks, APIs, header files and data types are all abstracted, providing you with complete separation from your operating system.

When possible, OS Abstractor uses the compiler preprocessor to parse your application code for the MapuSoft function calls, and then translates them to the appropriate operating system call. By leveraging the compiler preprocessor, we are able to eliminate the overhead associated with the wrapper solution. The preprocessor performs the necessary mapping and translation between the application code and operating system calls. As an application programmer, you write your code using C and let OS Abstractor perform all the necessary translation for your specific operating system.

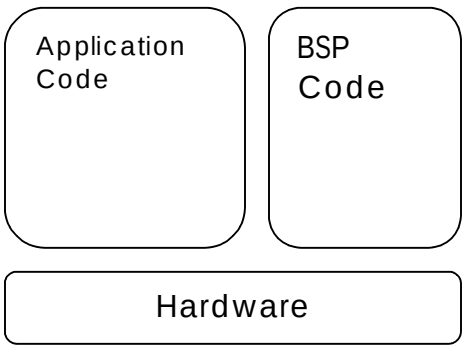
Changing Hardware due to Industry Consolidation

One major factor that disrupts embedded software development is changing hardware. Bringing in new hardware, whether it's a new CPU or development board, can wreak havoc with your application development. Hardware changes can mean hours of re-work and code updates. OS Abstractor alleviates these problems and reduces your future work.

Most embedded software development is divided into two groups: low-level programming and application development. The low-level programmers deal with hardware issues like writing software drivers, start-up configuration, hardware boot, soft / hard device resets, and device initialization.

The application development team creates the high-level functionality of the product. These are the features that will be used by the end user of the application. Application developers focus on the core functionality of the application. They make several assumptions about the state of the hardware – they assume it's working properly and has been initialized. They don't worry about memory integrity checks, start-up sequences, initialization, or other power issues with the hardware. Those issues normally fall into the hands of the programming team handling the BSP and device driver development. Unfortunately, there isn't a clear division of the software layers.

When the hardware changes, there are going to be some software changes. It really doesn't make much sense to re-work high level application code. With OS Abstractor changes are only required to the low-level code. Unfortunately most software designs don't provide this separation which protects the original application code.



Poorly designed architecture will create future coding issues

Well-designed architecture create stable base for future coding and maintenance

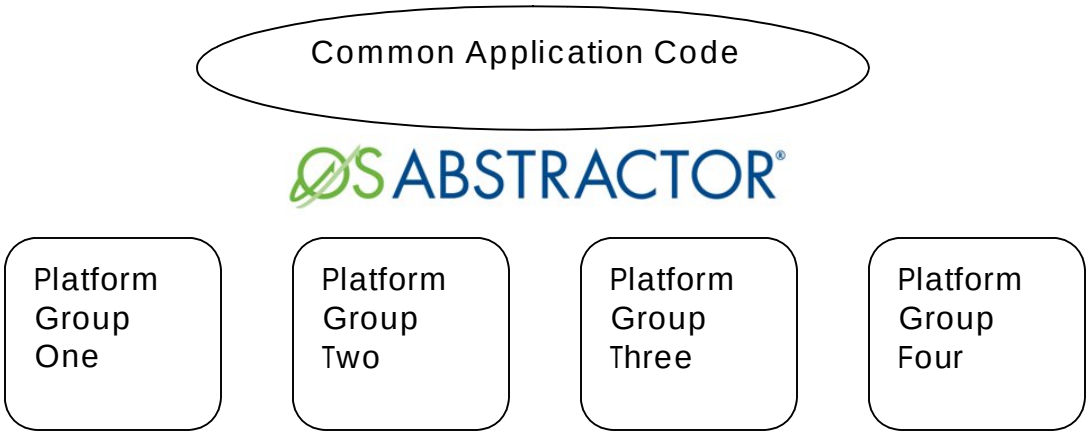
OS Abstractor provides a blanket of security for these situations. It allows you to continue your application development while the BSP team deals with the hardware change. It provides a layer of separation between your BSP development and your application development. So when you do change hardware, you can continue with your application development safely and on schedule. Using OS Abstractor, you create a software layer that eliminates the need to modify the application code.

Many times, this hardware change is unexpected. Either a board or CPU is discontinued, a new peripheral is added, or the performance is improved so drastically that there's no alternative but to jump to the next version of the hardware.

Additional reasons for changing hardware are:

- A new development board is released
- A new set of peripherals are added
- A CPU upgrade for speed, functionality, or manufacturing
- A marketing mandate to support multiple processors
- A CPU has become popular with customers

There is one additional scenario for supporting multiple hardware platforms. Sometimes application code needs to be shared across multiple groups or divisions. The individual groups may be targeting different markets or could be working on product enhancements. Sharing the common application code across these groups creates significant work for supporting each individual platform. By using OS Abstractor, this sharing of code is made easier, since all groups are using a common API.



Hardware Changes from industry consolidation

A fragmented market is characterized by multiple vendors and products. In the embedded world, there are hundreds of options for an operating system and hardware platforms. Among other things, the options vary greatly on performance, business model, and product features. These variations make for a textbook example of a market that's fragmented.

In a fragmented market, there is one thing that always occurs - consolidation. Industry consolidation is typified by mergers and acquisitions (M&A) between related as well as rival firms. You can watch the business news each week and see announcements in various markets. The embedded and semiconductor markets have seen consolidation over the years. Wind River acquired ISI back in the late 1990s. Renesas came from Hitachi and Mitsubishi. AMD acquired ATI. Mentor Graphics acquired Accelerated Technology. And, most recently, Intel acquired Wind River.

Industry consolidation through M&A is more important than a simple partnership or alliance, because there is a transfer of ownership between two companies. This allows a single management team to drive the direction of the newly acquired company. From a management point of view, this provides more control and, in theory, produces better results when compared with a simple partnership.

Despite the positives, there is also a dark side to mergers and acquisitions. The dark side occurs when existing operating systems support fewer hardware platforms. If your hardware is no longer supported, then your customers may be forced to change their platform, potentially moving to your competitor's hardware architecture.

OS Abstractor provides a migration path away from the discontinued operating systems. It allows your hardware to immediately support up to twenty commercial operating systems from major OS vendors.

CPU vendors / silicon vendors

Silicon and CPU vendors are in an interesting position. When they develop a new CPU, there is the task of getting customers to use this new platform. These new processors offer the potential for new, innovative devices because of their improved performance, but there is limited software support.

From a customer's perspective, there may be resistances to these upgrades from the old CPU because of one thing – their application software is written to one individual platform. Moving to a new platform means weeks and months of updating and re-writing software. OS Abstractor eliminates the changes needed to the application code, making the change to a new CPU virtually seamless.

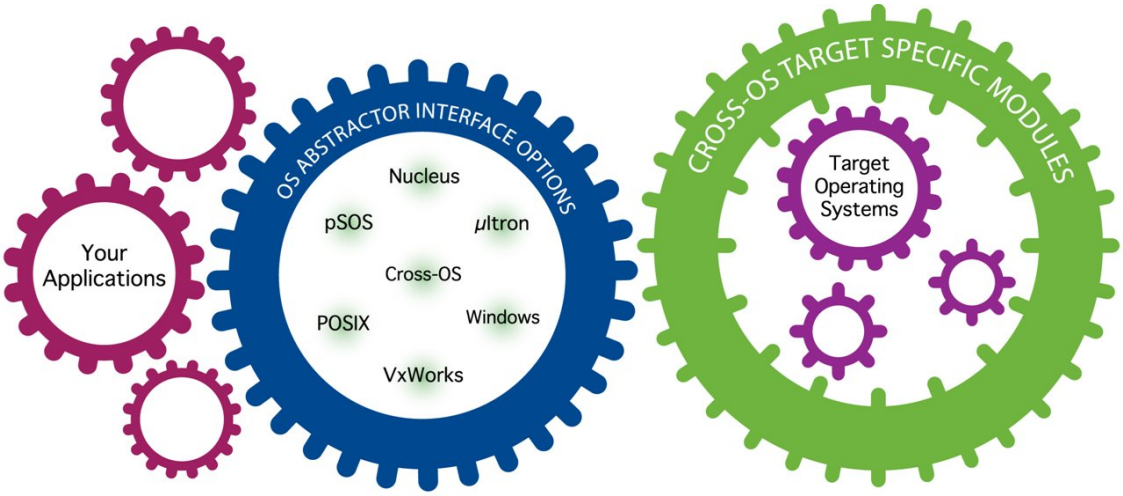
When new hardware platforms are introduced, OS Abstractor provides a software migration path to the new platform. An example of this would be the introduction of the Atom architecture from Intel. This new architecture represents a future path for Intel and it's customers. The Atom is designed for low power devices and targets mobile applications, Netbooks, and entry-level PCs. If you, as a developer, want to move your software from the x86 architecture, there will be significant work associated with the transition. However, OS Abstractor provides you with the capability to quickly move your application code to this new architecture with little rework. The low-level drivers and BSP will require modification, but OS Abstractor eliminates changes to your application code.

There is another issue when changing CPUs or hardware platforms - How can you make sure your application is performing at the same level?

By using the profiler with OS Abstractor, you're able to monitor your application performance as you make the transition. Using the OS Abstractor profiling options, you can record multiple sessions and track the metrics for performance. Then, as you make the change to the new hardware system, you can keep a running record of the application's performance. As you are progressing, you generate a Timing Comparison Report to compare two different sets of measurements and reports. You make changes as needed to maintain your particular performance metrics. This will help guarantee that your application is performing efficiently and accurately.

OS Abstractor - The Solution

OS Abstractor provides an excellent choice for your software project and offers several benefits for your software application.



Reduce your the learning curve

One of the most under-estimated tasks in a software project is the ramp-up time developers must go through to become productive on a new operating system. The cost of learning a new operating system is significant. OS Abstractor offers an easy-to-learn interface that can be re-used across projects, reducing your team's learning curve and increasing their productivity. Alternatively, you can also continue developing using your existing API and let OS Abstractor handle the work of supporting the new operating system.

Good programming techniques reduce un-necessary bugs and help maintenance

Having a solid architecture brings many benefits to a software project. A solid architecture allows for better testing during both unit testing and system testing. Project activities of adding new features and maintenance are significantly easier when an application has a solid architecture.

Reduce your risk and dependency on a single OS vendor

Coupling too closely to the OS can lead to dependency on the future of the operating system vendor. As vendors drop support for your hardware platform or prices change, your project could be jeopardized. Using OS Abstractor helps reduce dependency on the OS vendors.

Provide additional functionality

OS Abstractor also provides additional functionality to your development platform and OS. There are profiling features allowing you to optimize your application's performance. You can also add software based processes and shared memory functionality to your OS, even if they do not support those features.

Reduces work when changing hardware

Hardware changes can cause major re-writes of your application code. Using the OS Abstractor helps isolate your application code from the CPU which significantly reduces the work required for future hardware changes. By providing a separation between your application and the hardware, only low-level hardware changes are needed. There is no need to modify the application code when you move to a new hardware platform.

About Mapusoft

MapuSoft Technologies (MT) is the number one provider of embedded software re-usability solutions and services that are designed to protect software investment by providing customers a greater level of flexibility and control with product development. In addition to off-the-shelf tools, MT offers porting, integration, support and training services to help developers easily migrate from legacy platforms to the next generation. We believe that our advanced software and vision will revolutionize the embedded software industry. We are working hard to provide software that is practical, familiar, financially reasonable, and easily operable. We provide full source code with no royalty fees. Our licensing strategy makes it extremely affordable for you to incorporate our products into your embedded applications. In addition, our attention to engineering detail provides you with robust software and requires minimal technical maintenance.

About OS Abstractor

OS Abstractor is a C/C++ source-level virtualization technology that provides a flexible and robust real-time application development framework, which prevents your software from being locked to a specific operating system (OS) or version. This negates future porting issues because your software will support multiple operating systems and versions from the start of your project. It also eliminates the risk associated with the OS selection process, since the same application can be tested on multiple platforms for comparison and won't be locked-in to the chosen OS.

For more information

To download MapuSoft's free software evaluation visit:
http://mapusoft.com/downloads/

To learn more about our licenses and request a quote visit:
http://mapusoft.com/downloads/request-a-quote/

OS Abstractor Development Kit Overview Datasheet

OS Abstractor Development Kit Technical Datasheet

Contact Information

For more information about OS Abstractor or Mapusoft, please contact us at:

US Headquarters

MapuSoft Technologies, Inc.
1301 Azalea Road
Mobile, AL 36693

Tel: (251) 665-0280
Toll Free: 1-877-MAPUSOFT (1-877-627-8763)
Fax: (251) 665-0288

www.mapusoft.com

E-mail: info@mapusoft.com or sales@mapusoft.com

For a complete listing of International Offices, please click here.

Mapusoft's OS PAL: