

## 4.2.1.3 Runtime Platforms

[Return to Platforms](#)

### About

A **Runtime Platform** is the software components that are required to run application software that are not part of the [application](#) itself, [Hardware Platform](#) or [Operating Systems Platform](#). Some examples of Runtime [Platform](#) components are language specific libraries, or language specific virtual machines.

In general, an application is a sequence of steps or events directed and coordinated by the application. Some of the steps are actually contained within the application while others are executed by the Runtime Platform, [Operating Systems Platform](#) or the [Hardware Platform](#) on behalf of the application.<sup>1)</sup>

When an application is translated into a target hardware instruction set by a [compiler](#), there would be an extreme enlargement of the executable code if all the reused code from the software and hardware platforms were added to the executable. Alternatively, compilers often use compiler-specific external functions in pre-compiled code and assembled into runtime libraries that are linked during execution. These libraries are generally optimized for efficiency and for segregation of functionality for improved accuracy, prevention of common runtime errors, or security concerns.

Runtime libraries provide functionality by accessing the underlying operating or hardware platforms. For example, the use of floating point arithmetic or array/vector processing that use array processors or multiple cores. These considerations can often blur the boundary between standard application runtime libraries and language specific runtime libraries. Often compilers provide copies of runtime libraries optimized for the features of the OS and the hardware.

The concept of a runtime library should not be confused with an ordinary program library like those created by application programmers or delivered as third party such as [Dynamic Link Library \(.dll\)](#) or [Shared Object \(.so\)](#), meaning a program library linked at run time. For example, the programming language C requires only a minimal runtime library (commonly called `crt0`) but defines a large standard library (called `C standard library`) that each implementation delivers.

Consequently, the successful execution of an application depends on the proper versions and revisions of all other the Relocatable Objects. It also means that some kinds of errors can only be caught at execution or run time. These kinds of errors are best caught using strategic testing methodologies and strategies such as [regression](#) or [unit](#) testing (see: [testability](#) for more on testing).

<sup>1)</sup>

Mansourov, Nikolai and Campara, Djenana; 2011, Accessed: 10 December 2020,  
<https://www.sciencedirect.com/topics/computer-science/runtime-platform>

Last  
update: 2021/08/13 11:56 dido:public:ra:1.4\_req:1\_func:platform:sw\_arch [https://www.omgwiki.org/dido/doku.php?id=dido:public:ra:1.4\\_req:1\\_func:platform:sw\\_arch&rev=1628870187](https://www.omgwiki.org/dido/doku.php?id=dido:public:ra:1.4_req:1_func:platform:sw_arch&rev=1628870187)

---

From:  
<https://www.omgwiki.org/dido/> - **DIDO Wiki**

Permanent link:  
[https://www.omgwiki.org/dido/doku.php?id=dido:public:ra:1.4\\_req:1\\_func:platform:sw\\_arch&rev=1628870187](https://www.omgwiki.org/dido/doku.php?id=dido:public:ra:1.4_req:1_func:platform:sw_arch&rev=1628870187)

Last update: **2021/08/13 11:56**

