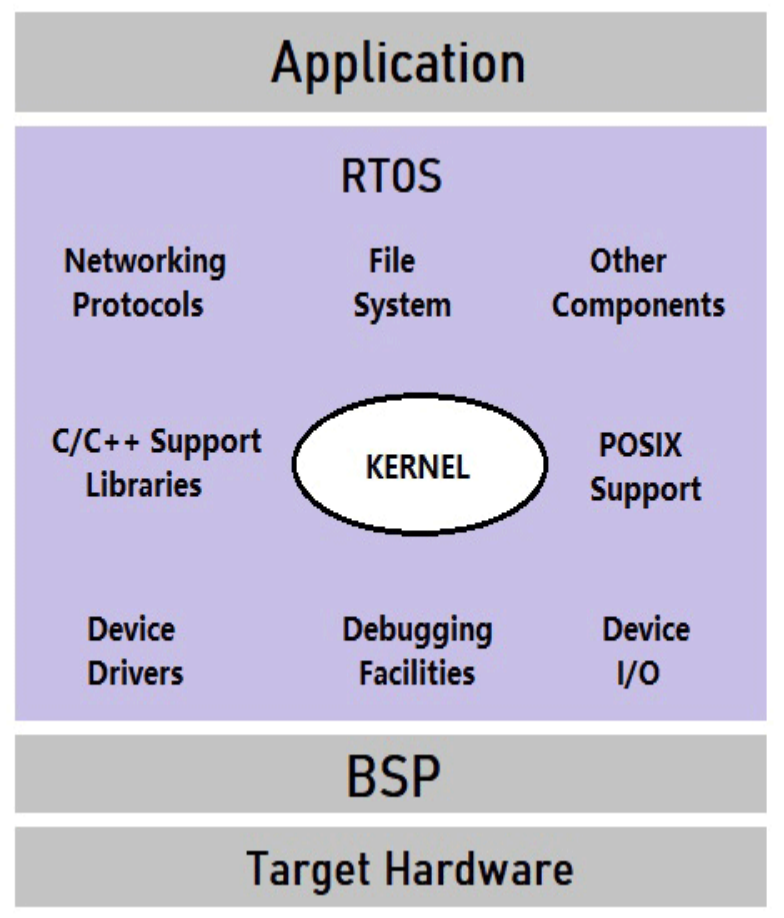
Programming Languages for Real-Time Systems

Contents

Introduction

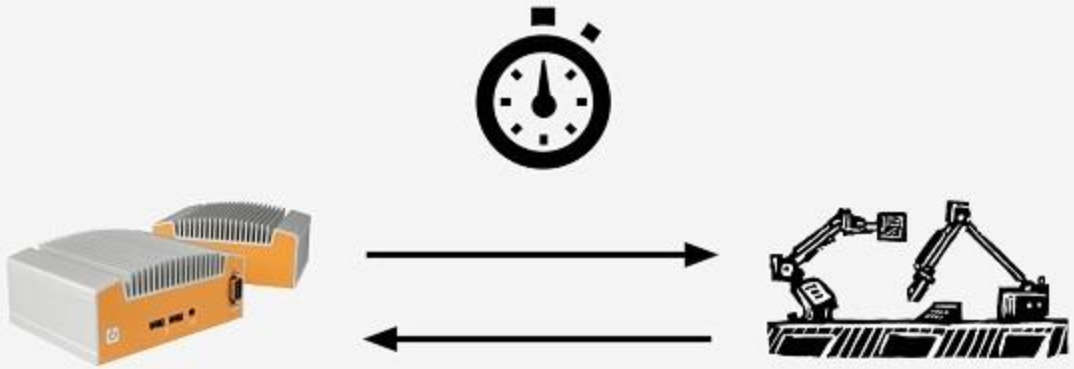


The real-time and embedded systems market is huge and growing all the time. It has been estimated that 100 times more processors are destined for embedded systems rather than the desktop. Embedded real-time systems :

- *are mainly small (for example, mobile phones) but can also be extremely large and complex (for example air traffic control systems)*
- *have potentially complex mathematical models of their controlled environment*
- *must be dependable*
- *are inherently concurrent*
- *must interact within the time frame of the environment*
- *must interact with low-level mechanisms such as hardware devices and memory management faculties.*

Landscape

- Rather than consider all possibly real-time programming languages, this section focuses mainly on three representatives groups of the landscape: C/C++ based languages, Ada and Real-Time Java (in particular the Real-Time Specification for Java). Sequential languages such as C and C++ are not reviewed in their own right as (a) their advantages and disadvantages are well known (for a full discussion) (b) they do not support the main characteristics of embedded real-time systems and consequently (c) their use requires direct interaction with the facilities of an underlying operating system .



Real-Time Operating Systems

Ada 95

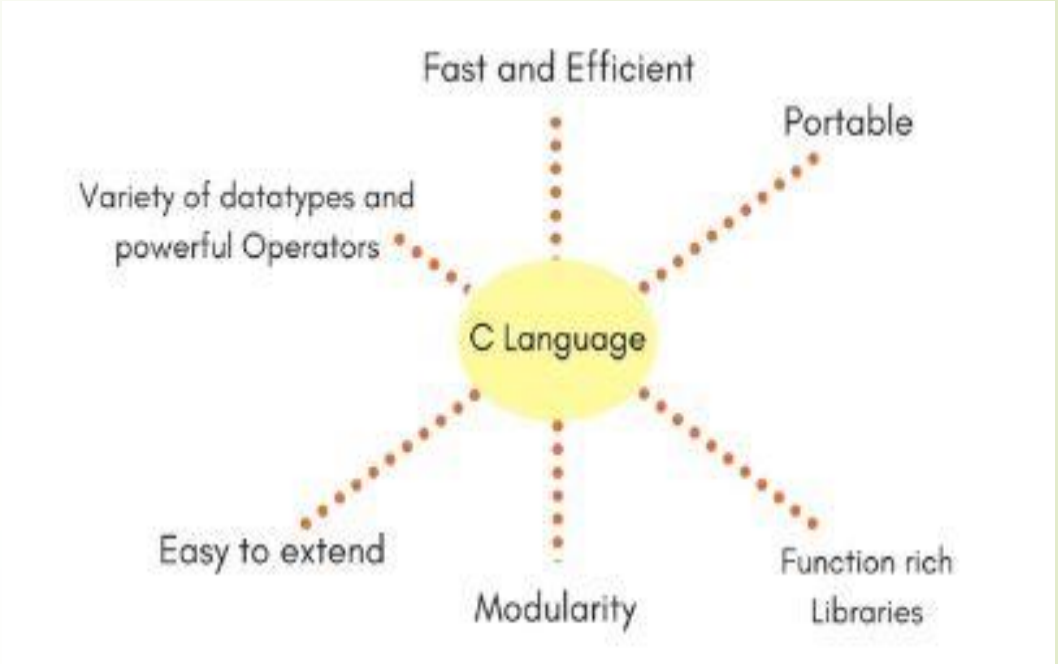
- The development of the Ada programming language forms a unique and, at times, intriguing contribution to the history of computer languages. As all users of Ada must know, the original language design was a result of competition between a number of organizations, each of which attempted to give a complete language definition in response to a series of requirements documents. This gave rise to Ada 83. Following ten years of use, Ada was subject to a complete overhaul. Object-oriented programming features were added (through type extensibility rather than via the usual class model), better support for programming in the large was provided (via child packages) and the support for real-time and distributed programming was enhanced. The resulting language, Ada 95, is defined by an international ISO standard.

Real-Time Java

- Since its inception in the early 1990s, there is little doubt that Java has been a great success. The Java environment provides attributes that make it a powerful platform to develop embedded real-time applications. Since embedded systems normally have limited memory, an advantage that some versions of Java (for instance J2ME) present is the small size of both the Java runtime environment and the Java application programs. Dynamic loading of classes also facilitates the dynamic evolution of the applications. Additionally, the Java platform provides classes for building multithreaded applications and automatic garbage collection; these make it an attractive environment to develop embedded real-time applications. Unfortunately, the problem with garbage collection is that it introduces random pauses in the execution of applications. Consequently, Java does not guarantee determinism nor bounded resource usage, which are needed in embedded real-time systems.

Real-Time C

Real-Time Concurrent C extends Concurrent C by providing facilities to specify periodicity or deadline constraints, to seek guarantees that timing constraints will be met, and to perform alternative actions when either the timing constraints cannot be met or the guarantees are not available



Real-Time POSIX.

POSIX (Portable Operating System Interface) is a set of standard operating system interfaces based on the Unix operating system. The most recent POSIX specifications -- IEEE Std 1003.1-2017 -- defines a standard interface and environment that can be used by an operating system (OS) to provide access to POSIX-compliant applications. The standard also defines a command interpreter (shell) and common utility programs. POSIX supports application portability at the source code level so applications can be built to run on any POSIX-compliant OS.

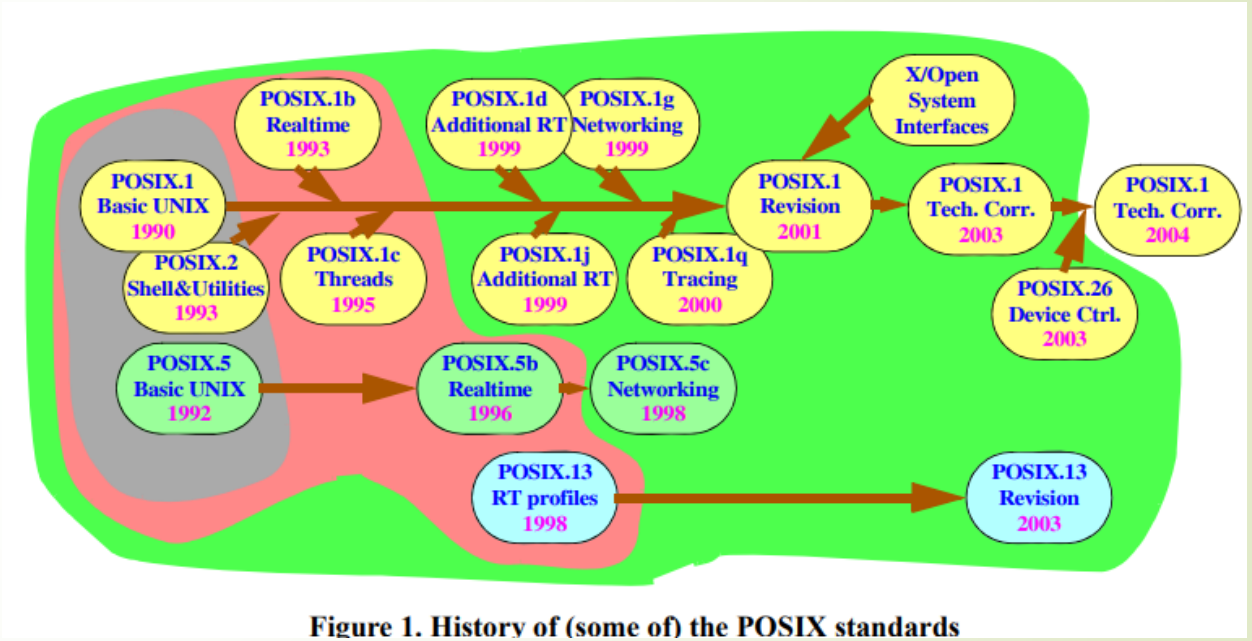
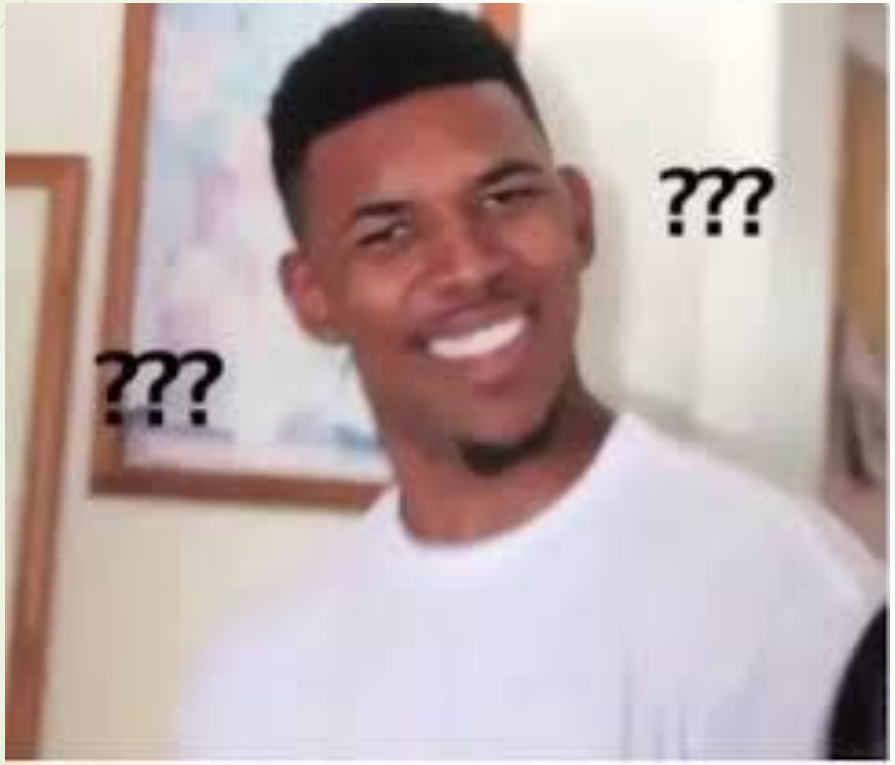


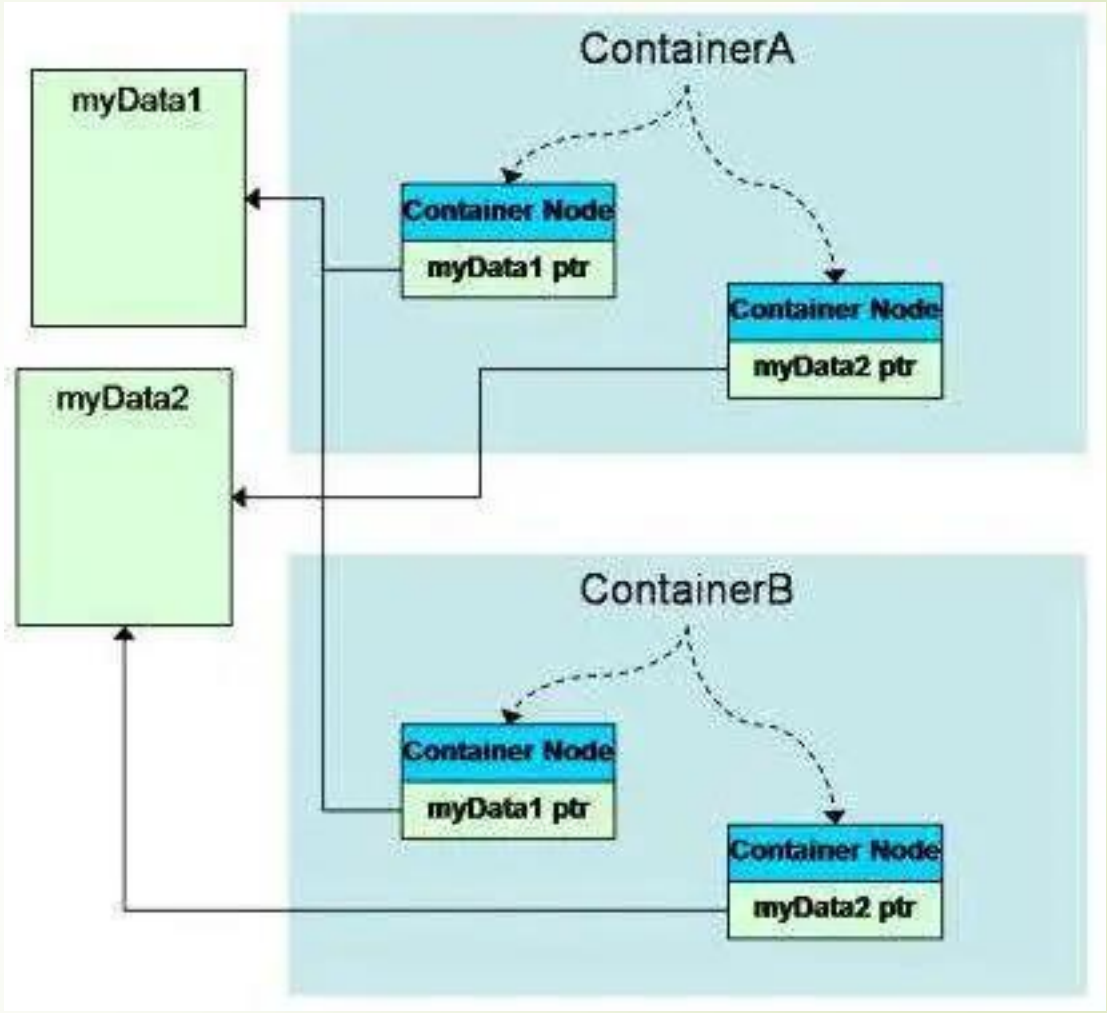
Figure 1. History of (some of) the POSIX standards

Without C++ ???



Of course C++

- Operating Systems. C++ is a fast and strongly-typed programming language which makes it an ideal choice for developing operating systems. ...
- Games. ...
- GUI Based Applications. ...
- Web Browsers. ...
- Embedded Systems. ...
- Banking Applications. ...
- Compilers. ...
- Database Management Software.



Conclusion

The book covers use of Ada 95, the Java Real-Time System and realtime POSIX extensions (programmed in C). None of these is directly a domain specific language.

Ada 95 is a programming language commonly used in the late 90s and (AFAIK) still widely used for realtime programming in defence and aerospace industries. There is at least one DSL built on top of Ada - SparkAda - which is a system of annotations which describe system characteristics to a program verification tool.

This interview of April 6, 2006 indicates some of the classes and virtual machine changes which make up the Java Real-Time System. It doesn't mention any domain specific language extensions. I haven't come across use of Java in real-time systems, but I haven't been looking at the sorts of systems where I'd expect to find it (I work in aerospace simulation, where it's C++, Fortran and occasionally Ada for real-time in-the-loop systems).

Realtime POSIX is a set of extensions to the POSIX operating system facilities. As OS extensions, they don't require anything specific in the language. That said, I can think of one C based DSL for describing embedded systems - SystemC - but I've no idea if it's also used to generate the embedded systems.

Not mentioned in the book is Matlab, which in the last few years has gone from a simulation tool to a model driven development system for realtime systems. Matlab/Simulink is, in effect, a DSL for linear programming, state machines and algorithms. Matlab can generate C or HDL for realtime and embedded systems. It's very rare to see an avionics, EW or other defence industry real-time job advertised which doesn't require some Matlab experience. (I don't work for Matlab, but it's hard to over emphasis how ubiquitous it really is in the industry)