

## Old EKA1 architecture

Instead of *C++ exceptions*, which were not part of the C++ standard when Symbian OS was designed, the operating system uses a lightweight exception-handling mechanism, called a *leave*. Leaves may occur **as a result of an error condition or abnormal event**, such as *insufficient memory* or *disk space* to complete a request. The **leave propagates the error** to a point in the calling code where it can be handled, called a TRAP harness, unwinding the call stack as it does so.

However, because of the *jump*, any **local resources**, such as memory allocated on the heap, will be **orphaned**, potentially leading to **memory or resource handle leaks**. Developers working on Symbian OS use the cleanup stack to keep track of resources to which the only pointer is an automatic variable. In the event of a leave, the cleanup stack will destroy each of the resources placed upon it.

There are some important places in code which should **never leave** ? namely **C++ constructors and destructors**. Symbian OS classes typically use two-phase construction to avoid leaves occurring in construction code.

## EKA2 architecture ( S60 3rd Edition onwards )

The aforementioned restrictions do not apply anymore starting from the new EKA2 kernel architecture. The Leaves and TRAP harnesses are still there, but they are implemented in terms of normal C++ exceptions and try-catch blocks. This means that the Symbian code still looks the same, but under the hood Leave is an exception and TRAP is a try-catch block. The call stack is not merely unwound anymore. C++ takes care of calling the destructors for all stack objects so the possibility of a resource leak is gone.

Symbian coding convention has not changed even though the biggest reason behind no longer exists.

- T-classes still shouldn't own data even though they would work perfectly fine if they did.
- Two-phased construction is still there even though normal C++ style construction would be enough.
- CleanupStack is still used even though it is not strictly necessary.

In the future Symbian will probably move to more standard C++ so the conventions need to be re-evaluated.