

CAPE²

Configurable Avionic Platform for Embedded System & Software

Avionic Design Service GmbH
www.ads-avionics.com



CAPE²
CONFIGURABLE AVIONIC PLATFORM
FOR EMBEDDED SYSTEM & SOFTWARE

Introduction

In the media we can read about daily reports on schedule and cost overruns in military and defence programs. We must go into reverse in order to re-establish Germany's reputation as a centre of excellence for development of military and defence systems. Those programs need to be realized in time and budget to regain credibility and to increase the competitiveness of our military and defence companies.

According to the following study [Gray-Report] a **typical military and defence program is about 80 percent above the original cost estimates and arrives around five years late.** The delays are mainly caused by technical difficulties in realizing those complex avionics systems.

Gray-Report

„Review of Acquisition for the Secretary of State for Defense“

Executive Summary

Procurement and support of military equipment consumes around 40% of annual defence cash expenditure and is of immense importance to the nation. Nonetheless, the Ministry of Defence (MoD) has a substantially overheated equipment programme, with too many types of equipment being ordered for too large a range of tasks at too high a specification. This programme is unaffordable on any likely projection of future budgets.

The result is that programmes take significantly longer than originally estimated, because the Department cannot afford to build them at the originally planned rate. They also cost more than they would otherwise, because the overhead and working capital costs of keeping teams within industry and the MoD working on programmes for a much longer period soaks up additional cash. Across a large range of programmes, this study found that the average programme over-runs by 80% or circa 5 years from the time specified at initial approval through to in service dates. The average increase in cost of these programmes is 40% or circa £300m. This study also estimates that the “frictional costs” to the Department of this systematic delay are in the range £900m – £2.2bn pa.

This paper deals with means to avoid technical difficulties in developing such embedded systems which will help to reduce costs and time to market.

From concept to realization

Many of today's avionics systems perform very similar functions despite their complexity and different underlying system architectures. One or more central on-board computers process data of the distributed devices and sensors and forward status information to Human Machine Interface (HMI) devices, such as Multi Function Displays (MFD) and Control & Display Units (CDU).

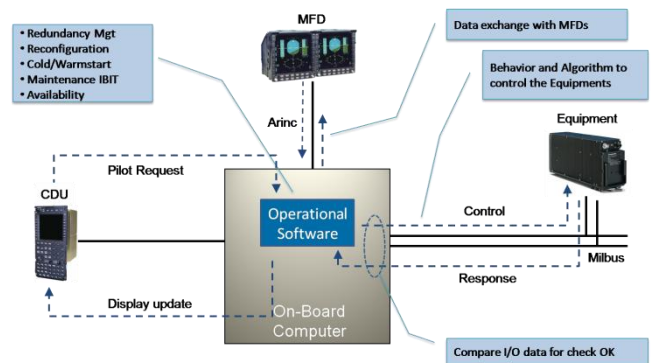


Figure 1- General Behaviour of Operational Software

Using this HMI equipment the crew controls the settings of the devices and sensors. The on-board computer provides an I/O interface manager for data transfer. Device or sensor specific processing of data is realized by the embedded operational software that is part of the on-board application software. Each device is controlled with a set of functions, also known as operational functions (OPF). As an example, the default usage of a VUHF radio in an avionics system implies the presence of a corresponding VUHF radio OPF within the operational software of the on-board computer.

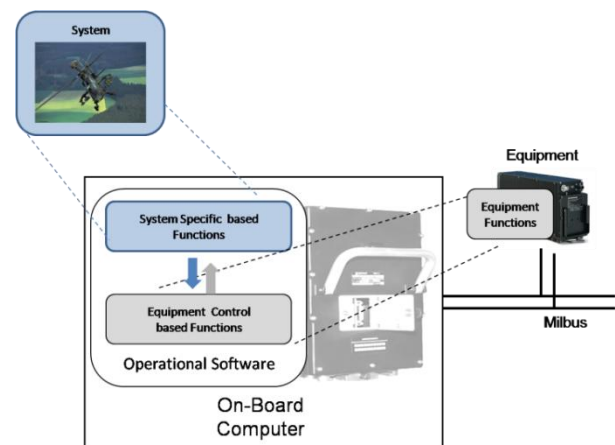


Figure 2 - System & Equipment Functions

In addition to the control of sensors, devices and device health status information via continuous (CBIT) and maintenance built-in tests (IBIT, PBIT) the OPF also implements device-independent system functions, such as redundancy management or HMI control and moding via lists and menus. A strict separation of device specific and system specific functions increases reusability of those components in similar applications.

CAPE² separates device from system specific functionality and provides the following advantages:

- Ability to transform specific I/O interfaces into platform and application independent data flows.
- Control of application specific software with a defined interface.
- Ability to easily embed operational functions.
- Mechanisms to port existing application code to different hardware environments using an encapsulating framework.

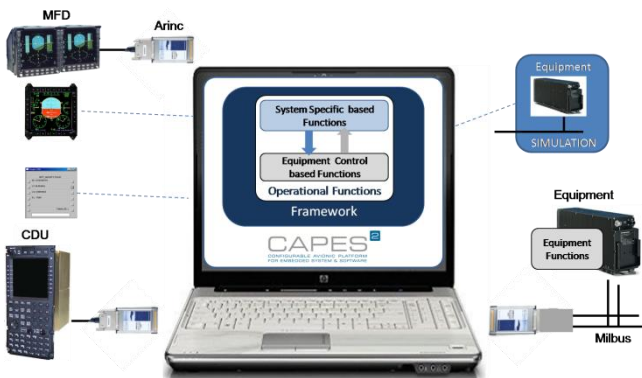


Figure 3 – Concept of CAPE²

In addition, CAPE² provides an interface to a simulated HMI interface. This interface is easily extendable by the user and allows full integration of CDU and MFD functionality (see figure).

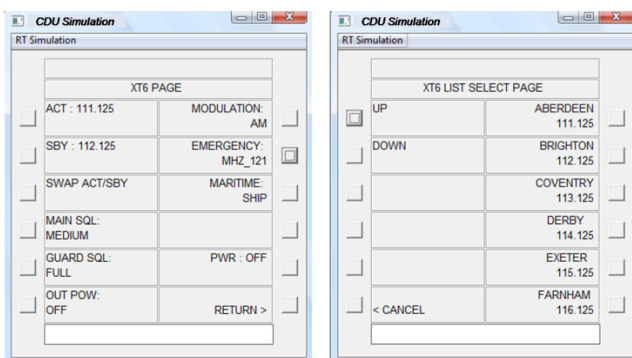


Figure 4- HMI Interface: CDU Simulation

Construction of the Framework

The requirement for CAPE² to be easily adaptable and extendable drove the implementation of the top level architecture. CAPE² is based on the ARINC 653 standard and uses partitioning to isolate different functionality. This is in accordance with the requirement to strictly separate system from application code. The system is configured according to required use case.

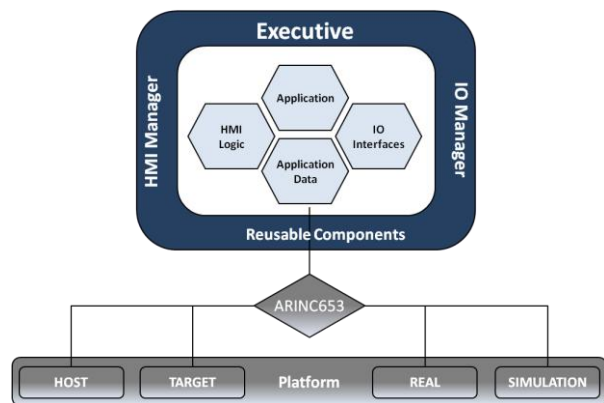


Figure 5 - Framework Components

The framework components are the basis of the CAPE² environment.

Executive

The executive is responsible for the complete real-time behaviour of the framework in which the entire application can execute.

HMI Manager

The HMI manager is responsible for handling of user commands and displaying status information. The HMI is based upon a freeware commercial of the shelf (COTS) graphics package. Other possible implementations could be based upon ARINC 661 graphics platforms.

IO Data Manager

The IO manager constitutes of a communication layer whose responsibility is the control, coordination and management of the data exchange between the external world and the framework. The HOST version of the data manager simulates I/O with queues and buffers. Specific data managers accommodate COTS I/O cards e.g. MILBUS, AFDX or ARINC 429. ARINC 653 based implementations use APEX sampling and queuing ports to communicate with the provided I/O drivers.

Reusable Components

The reusable components provide a library of standard and project specific elements like queues, lists, stacks, maps, specific types, components that cover instrumentation, exception handling, data access mechanisms and a mathematical library.

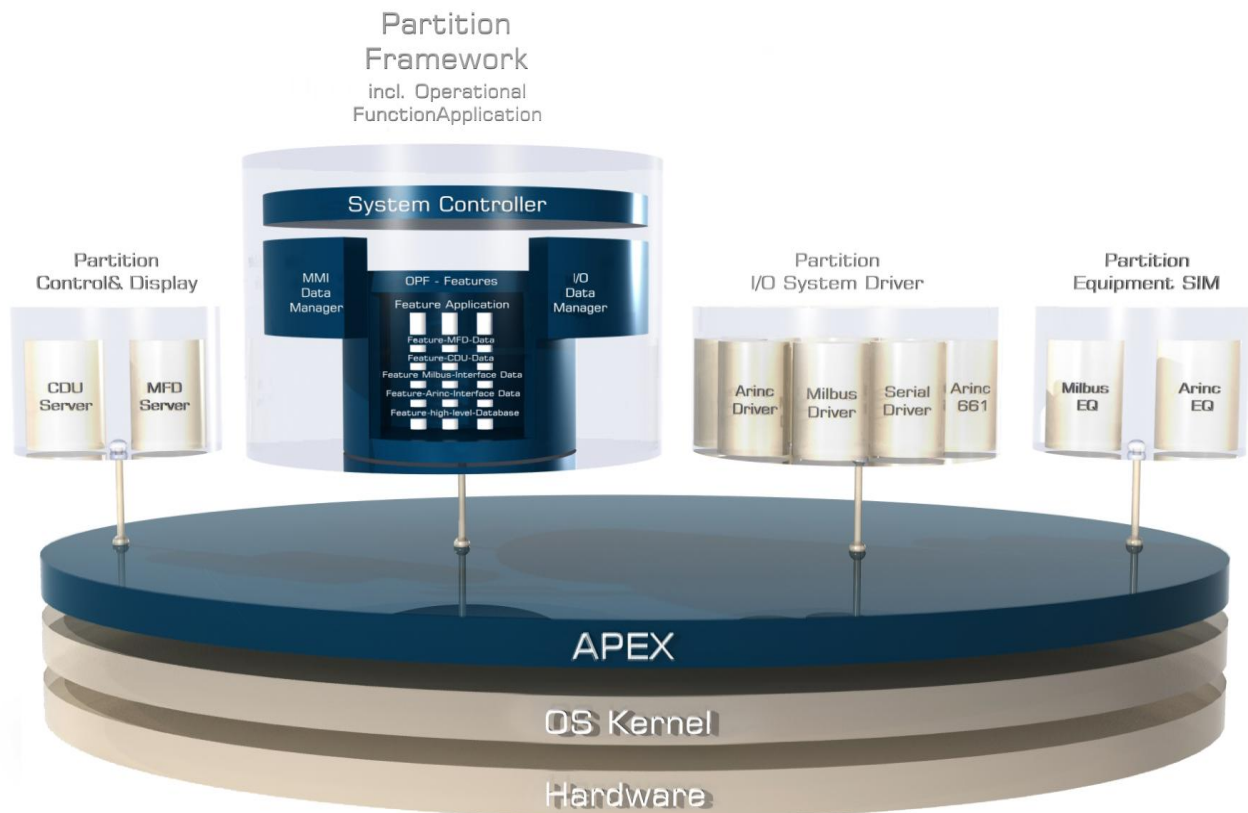


Figure 6 - CAPES² Framework

Application Components

The application components represent the software modules of the operational software. These should be developed in accordance with the feature model approach described in the System Software Product Line [SSPL]. Individual components are divided into the following areas:

HMI Logic

This software component contains the page definitions and logic for the system to be developed. The architecture of this component is such that standard control and moding elements can be generated from simple page definition documents.

IO Interfaces

This layer covers the messages and data structures required to define the interface to the outside world.

Application Data

This object models the visible data for the system to be developed.

Application Simulation

Host simulations of the controlled equipments can be included to allow closed loop tests of the application under development.

Business Cases

During the implementation of avionics systems delays are often caused by severe problems in

- developing the operational software,
- interacting with the devices to be controlled and
- integrating different (sub) systems.

Setting up a new defence program doesn't necessarily require re-developing the complete system down to subsystem or device level. This has clearly been demonstrated by some non-European defence companies in the recent development of avionics systems.



Figure 7- Various Systems - Same Equipment

The described situation allows deriving the following business cases for CAPES².

Business Case 1: **System Developer/Integrator - Prototyping, Stabilization of Requirements**

System Prototyping is the rapid development of a system. The main use is to help customers and developers understand the system requirements. Rapid software development can be used for requirements elicitation where users can experiment with a prototype to see how the system supports their work. Requirements validation allows revealing errors in the prototype or omissions within the requirements. Prototyping can be considered as a risk reduction activity.

Problem: Very late detection of problems, nearly at the end of development lifecycle.

Use Case: Quick availability of isolated functions on the target. Support for rapid prototyping on rigs. Usage of isolated functions

on pre-integrated products.

CAPES² allows quick derivation of prototypes from a set of reusable components by applying some mechanism to "glue" components together.

The composition mechanism includes control facilities and means for component communication. Individual components are integrated within the CAPES² framework. The prototype is developed by creating a user interface from standard items and associating components with them. CAPES² contains such a set of reusable components that can be adapted to provide an execution framework including standard HMI based upon industrial standard CDU control and moding.

Benefit: Time to market, system stability; minimized cost for software refactoring to fulfil flight worthiness standards. Allows to profit from model based engineering.

The main benefits of prototyping consist in eliminating possible misunderstandings between developers and software end users. Missing services may be detected and confusing services may be identified. A working system is available early in the process and the prototype may serve as a basis for deriving a system specification. The prototype can support user training and system testing. Prototyping allows the end user to be involved in system development. Not only is the system more likely to meet user requirements, but also they will commit to using the system.

Business Case 2: **System Developer/Integrator – Equipment Integration & Testing**

Modern weapon systems are heavily tailored to customer requirements. In many cases devices which are required to be integrated into an aircraft were not originally designed to be installed in an avionics system with its specific interfaces. Complicated algorithms and protocols need to be implemented in the OPF to enable communication between the equipment and the operational software. Integration becomes rather difficult because of the very dynamic behaviour of those algorithms.

Problem: Equipments those are difficult to be integrated into complex avionic systems. Static interface control definitions (ICDs) lead to problems in the dynamic behaviour and in realizing equipment control. Due to increasing equipment complexity a more comfortable integration environment is needed for simulation of the target system

Use Case: A wide range of configuration options allows to set up an appropriate integration platform and to integrate devices with application specific onboard processing functions easily and quickly.

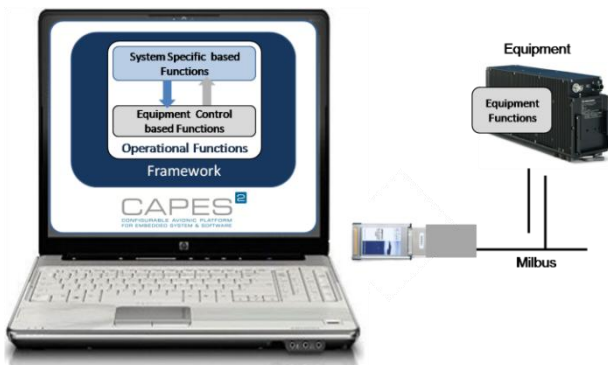


Figure 8 – Configuration for Equipment Integration

Benefit: CAPES² provides an easy-to-use integration platform that allows quick verification of the ICD and to establish confidence in the equipment behaviour.

Business Case 3:
System Developer/Integrator – Strategic reuse of Assets

The same equipment, e.g. a VUHF radio of a renowned German manufacturer, is adopted largely unchanged as COTS equipment in various weapon systems. However, in the past, the operational software that controls such a device has been developed and integrated separately for each system without taking the benefit from previous development activities. This attitude of “re-inventing the wheel” is no more acceptable in today’s economic environment where defence budgets are cut each year. From an economic point of view there needs to be a more sustainable usage of already developed assets.

Problem: Low level reuse of already developed functions – no reuse across projects:

- operational functions on different platforms are developed several times
- reuse of onboard software for simulation (rehosting / retargeting)

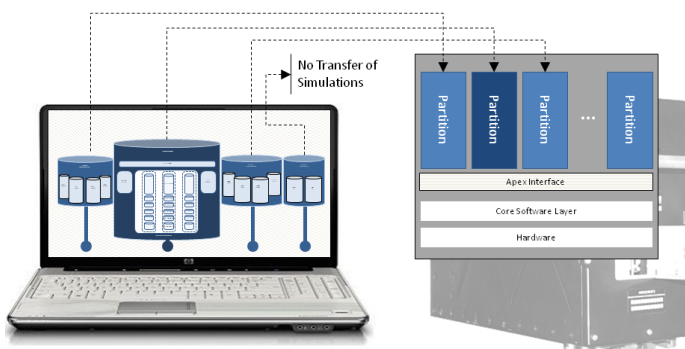


Figure 9 – Reusing & retargeting

Benefit: Time to market, system stability; Maximum profit from economies of scale: cost only depend on number of variations, not on number of products. Operational functions are independent of the target platform. Quick availability of isolated or complete set of functions on the target; Usage of isolated functions on pre-integrated products.

Business Case 4:
Equipment Developer – Product Testing

Problem: The requirements for avionics systems and thus the number of functions to be implemented increases with each weapon system. This applies to both the operational software of the on-board computer, as well as to embedded devices. In order to come up with the increasing complexity of equipments a more powerful test environment is needed that allows simulating the interaction of the target system with equipment.

Use Case: The reusability of operational software, specifically the functions for device control may in future be ported not only in new weapon systems; they can also be used by the equipment manufacturer in its product development lifecycle to provide a representative testing environment.

Benefit: CAPES² provides a user-friendly platform for equipment (pre)integration and product testing.

Business Case 5:
Equipment Developer – Strategic reuse of Technology Know-How - Equipment Integration into target System

Problem: Developing the Interface and System Requirement Specifications, which are subsequently used as a basis for the Software Requirements Specification and finally for implementing the operational functions, demand a great effort to the system integrator. This is a main driver for the overall development cost. There is still a high risk to misunderstand a certain specification level and thus to end up with a wrong implementation.

Use Case: The development of equipment software and the knowledge how to control devices can be made more efficient in adopting operational functions developed by the equipment manufacturer by the system integrator.

Benefit: CAPES² enables rapid porting of features and offers a suitable development platform for equipment manufacturers. It opens a new market for the equipment manufacturer as being a service provider that is able to provide a complete package.

Summary

*CAPES*² can be used to provide a concrete picture of the system's capabilities to end-users. It allows strategic reuse of assets across different platforms, by

- building a stable framework for the reuse of embedded software,
- establishing a platform for rehosting,
- giving possibility to perform rapid prototyping,
- providing a comfortable Integration & Test Environment and
- creating new business cases for equipment suppliers.

Glossary

*CAPES*² Configurable Avionic Platform for Embedded System & Software

CBIT	Continues Built-In Test
CDU	Control & Display Unit
COTS	Commercial of the shelf
IBIT	Initiated Built-In Test
ICD	Interface Control Definition/Document
I/O	Input and Output
MFD	Multi Function Display
HMI	Human Machine Interface
OPF	Operational Function
PBIT	Power-up Built-In Test
SSPL	System Software Product Line
VUHF	Very Ultra High Frequency

References

[SSPL]
ADS-Whitepaper "System Software Product Line"

[Grey-Report]
Review of Acquisition for the Secretary of State for Defence.
An independent report by Bernard Gray"