

The CAPTOR GEDAE Development Process

C. Thomson

BAE SYSTEMS Avionics
Sensor Systems Division,
Crewe Toll House, Ferry Road,
Edinburgh EH5 2XS, U.K.

campbell.thomson@baesystem

s.com

ABSTRACT

The CAPTOR project is a multi-national European project to develop the nose radar for the Eurofighter Typhoon aircraft. The first phase of the project, Tranche 1, has developed a baseline radar design. The signal processing software is being regenerated under the second phase of the programme, Tranche 2, and the GEDAE tool from Blue Horizon Development Software (BHDS) has been selected as the main development tool to support this activity.

The signal processing development approach used in Tranche 1 was not ideally suited to use with tools like GEDAE, so a new process was required.

The new development process designed for CAPTOR Tranche 2 will exploit the features of GEDAE and allow engineers to make full use of the GEDAE tool while still working within a well controlled environment to ensure that the workproducts are produced to a standard acceptable to both the company and the customer.

The development process supports iterative, incremental development of the GEDAE model, and is based on decomposition of the signal processing function into building blocks. Each building block can be developed and tested using GEDAE. Integration of these building blocks into higher level structures and testing at each stage allows the full signal processing functionality to be built up.

Introducing each incremental build to a representative target identifies potential problems early and should minimise the final target integration activities.

The proposed development process is designed to be suitable for the joint development of the Signal Processing Software by BAE SYSTEMS and EADS at different sites.

The proposed development process is also designed to allow portable software to be developed, to mitigate re-development cost against future hardware obsolescence and to give the potential for model re-use on other platforms.

The new process is expected to be significantly more efficient than the Tranche 1 development: this is needed to allow the software to be developed in time for the delivery of Tranche 2 radar systems.

The process has been documented in a Software Standards and Procedures Manual. A Code of Practice has been developed to support the process. This defines the way to use the Software Design Toolset, including GEDAE, to support the lifecycle, and carries a significant amount of detailed instructions on how GEDAE should be used on CAPTOR.

A pilot study has been carried out to demonstrate the suitability of the development process by developing the signal processing software for a CAPTOR radar mode using the GEDAE tool. The study successfully demonstrated the process, and recommendations arising from the study have allowed improvements to be incorporated.

Full-scale development of the CAPTOR Signal Processing Software using GEDAE is now underway, and the new development process is being used successfully.

1. INTRODUCTION

The CAPTOR project is a joint European project to develop the nose radar for the Eurofighter Typhoon aircraft.

1.1 The CAPTOR Programme

The project is split into a number of phases known as Tranches. The Tranche 1 (T1) development phase is complete and the radar is now in production and following extensive flight testing on the development Eurofighter aircraft.

Preparation for Tranche 2 (T2) is now underway and unfortunately, as a result of the rapid rate of development of commercial processing systems, many of the original electronic components used in T1 are no longer obtainable. As part of the redesign activity necessary to respond to the component obsolescence the opportunity is also being taken to put in place a development toolset infrastructure that will support future reuse of the signal processing software. GEDAE has been selected as the primary tool that will be used for the re-generation of the Signal Processing Software (SPS) and in addition to the tool a new development process is also required to make best use of the tool and facilitate reuse.

2. The Tranche 1 Process

Tranche 1 signal processing development followed a traditional waterfall lifecycle with strict demarcation of the analysis, design, coding, integration and testing phases of the programme. A first generation RTSA/ SD modelling based tool was used to support this. Extensive documentation was carried out at each lifecycle stage to support reviews. The overhead associated with maintaining this model/ documentation workset through the development is high.

This approach is not ideal for using with a tool like GEDAE, which is best suited to rapid iterative lifecycles, so a new process has been created.

3. The new CAPTOR Process

The development process defined for newly developed CAPTOR Tranche 2 software is incremental and iterative. This approach allows the execution of an efficient development lifecycle that reduces risk and leads to early workable products. This eliminates the disadvantage of the classical waterfall model in which executable products are only available at the very end of the development, delaying the exposure of key problems.

3.1 Objectives

The overall objectives for the T2 process are:

- to maximise the effectiveness of the GEDAE tool and efficiently generate a high quality signal processing software product
- to generate an easily maintainable and upgradeable product
- to generate signal processing software which is sufficiently generic to be portable
- to break the signal processing into algorithms which are reusable both in the CAPTOR signal processing and in other radar applications

3.2 Overview

The GEDAE Development Process has three major phases:

- Requirement Analysis
- Architectural Mode Design
- Implementation

The purpose of the Requirements Analysis phase is to identify and document all of the requirements to be met by the system

being developed. These will be documented as Algorithm Design Descriptions (ADDs).

The purpose of the Architectural Mode Design phase is to generate a functionally correct flowgraph representation of the requirement being developed, using GEDAE.

The purpose of the Implementation phase is to port the flowgraph to the required target hardware platform and optimise the model to meet any timing requirements.

Iterations take place especially between the Requirements Analysis and the Architectural Mode Design.

While the Architectural Mode Design process is focused on functionality and performance in terms of user requirements like target detection etc., the Implementation process is about real time behaviour and adaption to interfaces within a specific environment.

3.3 Detail

The software development lifecycle used is incremental, and multiple iterations of the software development process may be carried out for each increment. The first increments will be used to establish a core capability in the signal processing software. Subsequent increments will be used to add functionality until the full required capability of the signal processing software is included and can be demonstrated. Each increment will follow the overall Development Process phases of Requirements Analysis, Architectural Mode Design and Implementation. The Architectural Mode Design process is shown in Figure 1.

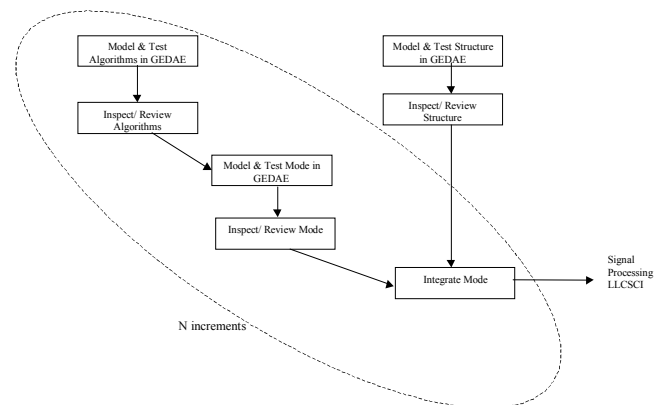


Figure 1 - The Architectural Mode Design Phase

At various stages within the software development lifecycle, interim engineering versions of the signal processing software will be made available for integration and testing purposes. These will not undergo a full delivery process. The scope and timing of the interim engineering versions of signal processing software will be identified in the project plans.

The software development lifecycle consists of the following activities (each iteration will consist of a subset, dependent on its Work Product):

- Generation and Inspection of Algorithm Design Descriptions (ADDs)
- Creation of a GEDAE Host Model of the Signal Processing Software
- Review/ Inspection of the GEDAE Host Model
- Creation of Test Vectors and Test Harnesses for the Signal Processing Software
- Review/ Inspection of the Test Harnesses
- Host based testing of the GEDAE Host Model
- Porting the GEDAE Host Model to GEDAE Target Model on representative target hardware
- Target based testing and optimisation of the GEDAE Target Model
- Updating of the Traceability
- Systems Integration and Test
- Software Delivery

Activities within the software lifecycle, as shown in the above list, may be carried out more than once on each increment of the overall process.

The development process is shown diagrammatically in Figure 2, indicating the process stages and their connectivity.

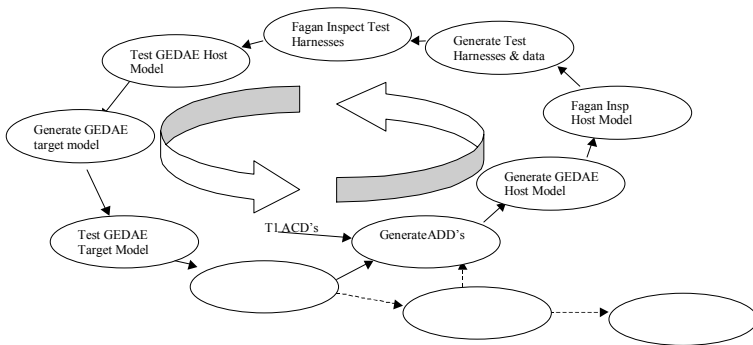


Figure 2 - CAPTOR SPS Development Lifecycle

3.4 Workproducts

The workproducts that can be generated / updated at each lifecycle stage are defined in Figure 3.

Lifecycle Stage	Work Products
Generation of Algorithm Design Descriptions (ADDs)	ADD objects for each algorithm specified for increment ADD objects for each mode or part of mode block diagram specified for increment

Lifecycle Stage	Work Products
Generate GEDAE Host Model	GEDAE Host Model for each algorithm specified for increment Test harness for each algorithm specified for increment Test data for each algorithm specified for increment Test results for each algorithm specified for increment GEDAE Host Model for each mode or part of mode specified for increment
Inspect GEDAE Host Model	Defect list for all GEDAE Host Model algorithms and modes Metrics updated in metrics database Correction of all identified defects
Generate Test Harness and Test Data	Test Harness for each Mode Test data for each mode
Inspect Test Harness and Test Data	Defect list for all GEDAE Host Model test harnesses Metrics updated in metrics database Correction of all identified defects
Test GEDAE Host Model	Host Test Results for each mode or part of mode block diagram specified for increment
Generate GEDAE Target Model	GEDAE Target Model for each mode or part of mode specified for increment
Test GEDAE Target Model	Target Test Results for each mode or part of mode block diagram specified for increment
Update Traceability	All new ADD objects linked to higher level requirements All new GEDAE material linked to ADD All new test data and test harnesses linked All test results linked Metrics updated in metrics database
Systems Integration and Test	Integration Test results
Software Delivery	Version Description contribution

Figure 3 - GEDAE Workproducts

The reviewable and deliverable items generated throughout the CAPTOR T2 development will be available in their native format i.e. models, text objects within requirements traceability tools etc. At the end of the programme, if the customer wishes a full set of documentation, auto-documentation features in the tools will be used to build these documents.

4. Supporting Documentation

A set of process documentation has been created, which define this process and comply with the participating companies' standards for software development documentation.

This has been presented and favourably received by the CAPTOR radar customer.

A Software Development Plan (SDP) defines the basic process to be followed and tools to be used at a high level. This identifies the type of lifecycle to be adopted (iterative incremental) and the GEDAE tool as the primary tool for SPS development, supported by PVCS Dimensions for configuration management and DOORS for requirement management.

A Software Standards and Procedures Manual (SSPM) defines the lifecycle to be followed for developing SPS software, and defines each of the stages in detail.

A Code of Practice (CofP) defines in detail how to use the tools (GEDAE in particular) to carry out the process defined in the SSPM. It also includes a project 'coding standard' for GEDAE modelling which defines project rules for flowgraph layout, headers, comments, naming conventions etc. Where appropriate, this document calls up the following documents:

- Project Specific Coding Standard for the 'C' Programming Language
- Defines standards to be applied to legacy C code
- CAPTOR Radar Tranche 2 Style Guide for Producing GEDAE Primitives
- Defines standards and guidelines for writing the GEDAE primitives – these are based on the project C standards where possible
- Blue Horizon Manuals and Guides
- Provides general guidelines on using GEDAE

The relationship between these documents is shown in Figure 4.

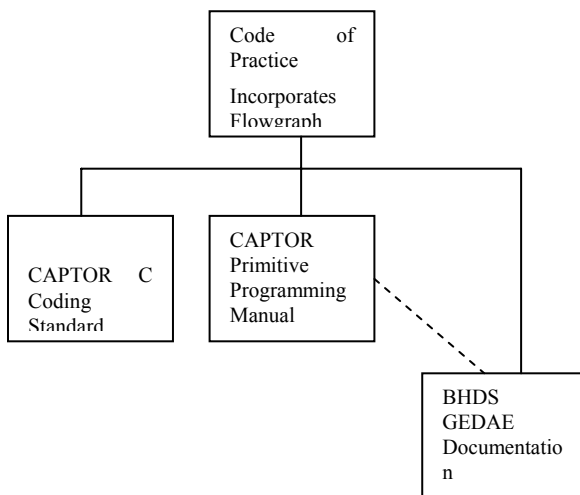


Figure 4 – Hierarchy of Standards

5. Process Evaluation – the captor pilot study

A pilot study was carried out to evaluate the new Captor Software Development Process and find out how well it works using the GEDAE tool.

One of the CAPTOR radar modes was selected and taken through the main lifecycle (not including the software release stages).

A Requirements Analysis phase identified the algorithms to be implemented in the selected radar mode. The potential for common algorithm reuse was also investigated.

The original requirements for the algorithms (from T1) were re-expressed in a generic form, as Algorithm Design Descriptions (ADD).

The radar mode was also expressed as an ADD, which is basically a block diagram defining the interconnection of the algorithm constituents.

During the Architectural Mode Design Phase each algorithm was modelled using GEDAE. For some algorithms, several different strategies were used to evaluate the differences (e.g. primitives versus flowgraphs).

Iterations of the generate ADD/ generate GEDAE Algorithm model were used to ensure that each ADD and algorithm matched by the end of the process.

Test vectors were generated for each algorithm and the algorithms were tested to verify successful operation.

The radar mode model was then built in GEDAE and again iterations were used to match the mode model and the ADD.

Test vectors for the mode were generated using radar data generator tools in BAE SYSTEMS [2], and mode testing was carried out to verify correct operation of the mode.

The model was then ported to a target platform, in this case a Sky Computers Rapid Development System with two quad G4 cards, and the mode testing was repeated first on a single G4.

Benchmarking was carried out on the G4, and then a variety of optimisations of the GEDAE host graph to run on a multi-processor target were evaluated.

5.1 Process Improvements

Some process improvements came as a result of the pilot study.

One of the problems identified during this study was that it is possible to develop large, complex models which compile and run on the host, but then either fail to compile or fail to schedule on the distributed target environment. This led to debugging and rework being needed.

As a result, the proposed development process was modified to show each increment of development having a host modelling and a target mapping phase, so that any potential problems in the target implementation are identified and solved as early as possible.

6. Modelling Standards in GEDAE

The Code of Practice for using GEDAE on the CAPTOR program provides detailed instructions and/or guidelines on how to use the

tool. Certain areas have been identified as key areas which must be controlled at project level to ensure that the model generated is a high quality product which is robust and maintainable.

6.1 Directory Structure

The directory structure which all of the associated GEDAE files are placed into is very important, particularly so for CAPTOR where the model will be very large (estimated >10K files) and development is split between two sites.

Two categories of model elements were identified:

- project specific elements (algorithms, modes etc)
- generic elements (complex vector comparison etc)

All project specific elements were placed in the following directory structure:

```
\FGlibraries\boxes\CAPTOR\algorithms\...
```

```
\FGlibraries\ boxes\CAPTOR\modes\...
```

```
\FGlibraries\ boxes\CAPTOR\SPS Framework\...
```

with corresponding test harnesses and data in

```
\FGlibraries\ boxes\CAPTOR\testsuite\algorithms\...
```

```
\FGlibraries\ boxes\CAPTOR\testsuite\modes\...
```

```
\FGlibraries\boxes\CAPTOR\testsuite\SPS Framework\...
```

A similar directory structure was set up inside Parameters, Groups etc.

Generic elements were placed in the directory structure:

```
\FGlibraries\ boxes\CAPTOR\embeddable\...
```

with a structure below this similar to the standard GEDAE embeddable directories.

This directory structure allows the CAPTOR application files to be kept separate from the GEDAE standard model files while still being in a suitable place as far as the tool is concerned on both NT and Solaris platforms. All of the CAPTOR specific subdirectories can be maintained in the CM tool and installed on top of any standard GEDAE installation.

6.2 Test Harnesses

For CAPTOR it is planned to have a test harness for each of the main GEDAE workproducts (algorithms, modes, etc).

These test harnesses will have two purposes:

1. Verifying that model components are unaffected by change due to
 - GEDAE upgrades
 - Different platforms
 - Modification of common elements
2. Testing model components¹

¹ Inspection is the primary method of verification of model components, and the test harness does not attempt to carry out

All of the test harnesses are being set up to allow automatic testing to be carried out from scripts. The CofP identifies guidelines to be followed in terms of generation of datasets, comparison of results and the layout of test harnesses to ensure that a uniform approach is used for these test harnesses.

6.3 Flowgraph Layout

The layout of flowgraphs in GEDAE can make the difference between having a maintainable, reusable product and having major problems. The interfaces to model components are also important to minimise integration problems.

While it is impossible to impose an exact and complete set of rules on such a subjective feature, it is important to have a basic framework of guidelines, and these are set out in the CofP.

The guidelines can be summarised as follows:

- data should flow from left to right
- minimise clutter to ensure that the data streams do not become unreadable
- the connecting lines should be straight with any bends at right angles
- connections should not pass over other objects
- streamed inputs/ outputs will be input/output, var and untyped and have the same name as the input in the ADD
- the data flow type of each input and output will be matched to the data type defined in the ADD i.e. vector complex inputs will be streamed in as complex vectors etc
- algorithms will all be generated as fixed size algorithms for situations where this is appropriate at the algorithm design level
- every canvas should have a title, security classification and copyright
- comments to be added to the canvas to aid clarity

6.4 Primitive Coding

A standard has been generated for writing GEDAE primitives. This contains rules for writing CAPTOR specific primitives and is based on the project C Coding standard. It prohibits the use of certain constructs, and gives recommended layout, naming conventions etc. It is designed to supplement the GEDAE Primitive Programmers' Manual [1].

It promotes the writing of primitives in as generic a way as possible to maximise the potential for reuse. As an example, it recommends useful variable names are selected but not project specific variable names (e.g. num_rows, num_columns rather than m,n or num_range, num_doppler).

7. Process Improvements

The iterative incremental development approach being adopted for T2 allows for continuous monitoring and improvement of the process as the product is developed.

complete unit testing; it does nevertheless provide evidence of correct execution over a reduced set of scenarios.

Metrics are being gathered and monitored (e.g. defects found at each stage of the lifecycle) and these are being used to identify improvements to the process.

8. CONCLUSIONS

An efficient workable process has been designed to allow GEDAE to be used effectively as the primary software development tool for a major defence contract. The process allows incremental development to be controlled and managed.

Full scale development of the T2 SPS is now underway, and the development process is working well. Some minor tailoring of the process has already happened as a result of the pilot and the initial development. Metrics are being gathered to allow continuous process assessment and improvement.

The knowledge gained from the pilot study, and careful planning of how the model will be partitioned on the target hardware, have

been used to influence key design decisions in the host model. This strategy does not invalidate the concept of a 'host model' nor the portability of the model, but minimises the impact of porting to the target and optimising the performance.

1 ACKNOWLEDGEMENTS

The author acknowledges the support of the various colleagues at SSD Crewe Toll and EADS Ulm who contributed to this work.

9. REFERENCES

- [1] BHDS, GEDAE Primitive Programmers' Manual
- [2] Yarker, S. Testing Radar Mode Graphs with the ARES Data Generator. GUC2003