Certification Authorities Software Team (CAST)

Position Paper CAST-19

Clarification of Structural Coverage Analyses of Data Coupling and Control Coupling

Completed January 2004

(Rev 2)

NOTE: **This position paper has been coordinated among the software specialists of certification authorities from the United States, Europe, and Canada. However, it does not constitute official policy or guidance from any of the authorities. This document is provided for educational and informational purposes only and should be discussed with the appropriate certification authority when considering for actual projects.**

Clarification of
Structural Coverage Analyses of Data Coupling and Control Coupling

1.0 Introduction

Numerous misinterpretations exist regarding the purpose of structural coverage analyses of data coupling and control coupling, and acceptable approaches to satisfying RTCA/DO-178B (and EUROCAE/ED-12B) Objective 8 of Annex A Table A-7. This objective ("Test coverage of software structure (data coupling and control coupling) is achieved") references section 6.4.4.2c of DO-178B/ED-12B which states that structural coverage "analysis should confirm the data coupling and control coupling between the code components."

This paper discusses the purpose, benefits, challenges, and future concerns of data coupling and control coupling coverage analyses.

It should be emphasized that this paper is intended to be an educational document that helps clarify DO-178B/ED-12B's intent regarding data coupling and control coupling analyses. It is not policy or guidance.

2.0 Background

Objective 8 of Table A-7 in DO-178B/ED-12B specifies the analysis of data and control coupling for Levels A, B, and C software. DO-178B/ED-12B Annex B defines data coupling and control coupling as follows:

> *"Data coupling - The dependence of a software component on data not exclusively under the control of that software component."*
>
> *"Control coupling - The manner or degree by which one software component influences the execution of another software component."*

Additionally, DO-178B/ED-12B defines a component as: *"A self-contained part, combination of parts, sub-assemblies or units, which performs a distinct function of a system."*

Note: DO-248B/ED-94B, *Final Report for Clarification of DO-178B Software Considerations in Airborne Systems and Equipment Certification*, Frequently Asked Question (FAQ) #67 provides some clarification on data coupling and control coupling. For reference purposes, the FAQ text is included in this paper as Attachment 1.

3.0 Purpose of Data Coupling and Control Coupling Analyses

3.1 Overview of the Purpose

The intent of structural coverage analyses is to provide a measure of the completeness of the testing process of computer software to ensure that the requirements-based testing (R-BT) of a software program exercised that program's functions and structure adequately to an appropriate level of "completeness" depending on that program's software level and needed integrity. For example, structural coverage analyses of Level C software only needs to provide a measure that all statements were exercised; Level B needs a measure that all statements and all decisions were exercised; and Level A needs a measure that all statements, all decisions and all conditions (plus some independence) were exercised (see Objectives 5, 6 and 7 of Annex A Table A-7). These measurements can be taken and analyzed at the computer program "module" level by reviewing test cases and executing requirements-based tests of that module in isolation from other program modules, and examining, either manually or with a tool, that every statement, decision, and condition (depending on the software level of the module) were exercised, and the module functioned correctly as designed.

"Module" is used in this context to denote a piece or component of the software program rather than the entire program. If an airborne software program consisted of one "module," the above structural coverage would likely be adequate to ensure that that the software program functioned correctly and would not have any side effects leading to anomalous behavior. However, because of the size and complexity of embedded airborne software programs, having the program consist of a single, self-contained module is neither practical nor good engineering. Thus, developers construct programs of modules or components (both functional and data components) that interact with one another and depend on one another to perform the program's functions.

The intent of the structural coverage analyses of data coupling and control coupling is to provide a measurement and assurance of the correctness of these modules/components' interactions and dependencies. That is, the intent is to show that the software modules/components affect one another in the ways in which the software designer intended and do not affect one another in ways in which they were not intended, thus resulting in unplanned, anomalous, or erroneous behavior. Typically, the measurements and assurance should be conducted on R-BT of the integrated components (that is, on the final software program build) in order to ensure that the interactions and dependencies are correct, the coverage is complete, and the objective is satisfied.

Satisfaction of this objective is dependent on the detail of the specification of the modules/components' interfaces and the thoroughness of the R-BT for normal range and robustness of the software program. That is, if the interfaces and dependencies are specified in the design requirements, and if those requirements are tested for both normal functioning and robustness, satisfaction of the objective may be a by-product of the design and verification processes. However, if the interfaces and dependencies are not well-specified, and the testing program is minimal, it will be much more difficult to demonstrate the objective has been satisfied.

3.2 Queries of SC-167/WG-12's Members Regarding the Intent of Objective 8 of Table A-7

Several queries about the intent of the structural coverage analysis objective of data coupling and control coupling were made to SC-167/WG-12 sub-group #4 members. The majority indicated that the objective was added to address coverage analysis of the testing of the entire software program related to its architecture and the integration of its modules/components. Before the objective was added, the committee recognized that an analysis existed to ensure that all requirements were examined for proper operation by testing of the binary image (i.e., requirements coverage). Additionally, analysis existed to demonstrate that the testing ensured that all of the statements of the source code were executed and for more critical software all the logic statements were thoroughly exercised. However, there was no guarantee that the integration and architecture were fully exercised in the execution environment. Therefore, the data coupling and control coupling objective was added to address this deficit. Committee members claim that the original notion was that call trees/scheduler graphs (control coupling) and data flow diagrams (data coupling) could be annotated (preferably by some tool) with test cases to demonstrate coverage. However, in reality, it turned out to be more complex than this.

To summarize, SC-167/WG-12 sub-group #4 members claim that the purpose of the structural coverage of the data coupling and control coupling is to evaluate the adequacy of the integrated testing and provides an analysis of the integration activities (e.g., evaluation of call trees, set-use tables, and integration test results). Satisfying the objective occurs during integration and ensures that there is a complete suite of integration tests.

3.3 Design Versus Integration Verification Activity

A number of manufacturers perform activities during software design to minimize the data coupling and control coupling issues during integration. This is recognized as a good engineering practice as the Meiler Page-Jones' book entitled *The Practical Guide to Structured Systems Design* (1980) points out. Page-Jones identifies different kinds of coupling to be considered during design (e.g., data coupling, stamp coupling, common coupling, control coupling, and content coupling). In some cases (depending on the architecture), this analysis in the software design phase can be used to supplement the software/software integration activity.

However, the DO-178B/ED-12B objective 8 of Table A-7 is primarily intended to be a verification of the integration activity; that is, verification that the interfaces and dependencies between the software program's modules/components were implemented as designed and are correct. Satisfying the objective is intended to provide a measure of the completeness of integration verification (R-BT of the integrated software program's structure, interfaces, and dependencies).

Many applicants find that documenting the data coupling and control coupling during software design provides the requirements to verify during the software/software integration and hardware/software integration verification process. That is, good documentation of the design helps to satisfy the objective during the integration testing (R-BT coverage of interface and dependency requirements).

3.4 CAST Perspective on Data Coupling Analysis Purpose

CAST proposes that the purpose of data coupling analysis is to:

- Be a completion check of the integration testing effort. The analysis also provides insight into the structural robustness of the data structures used by the program. Basically, data coupling analysis is intended to enforce good software engineering practices. Data coupling analysis becomes particularly important when partitioning and other protection means are implemented in the software.
- Identify data dependencies. As an example, a data dependence exists between two components when one component defines a data object and the other component uses the definition of that data object under some operational scenario. In this example, the data user is dependent on the data definer.
- Verify data interfaces between modules/components through testing and analysis (test it, then measure it).
- Identify inappropriate data dependencies.
- Define and evaluate the extent of interface depth.
- Determine and minimize coupling interdependencies.
- Determine and maximize cohesion.
- Evaluate need for and accurate use of global data.
- Evaluate input/output data buffers.
- Bound impact of change and requirements effect(s).

3.5 CAST Perspective on Control Coupling Analysis Purpose

CAST proposes that the purpose of control coupling analysis is to:

- Be a complementary completion check of the integration testing effort (i.e., it complements data coupling analysis). The analysis also provides insight into the structural robustness of the execution, timing, and scheduling. Basically, control coupling is intended to enforce good software engineering practices. Control coupling becomes particularly important when partitioning and other protection means are implemented in software.
- Identify control dependencies. A control dependence exists between two components when the execution of one depends on the other. For example, one module/component calls the other under some operational scenario (i.e., the callee is dependent on the caller). Another example is where one module/component defines the data objects that determine the execution sequence taken by the other module/component under some operational scenario.
- Identify inappropriate control dependencies.
- Verify correct execution call sequence (across modules/components/parts/units/objects).
- Define and evaluate the extent of interface depth.
- Assist in verifying scheduling (e.g., detects problems with call sequences that may cause frame overrun).
- Assist in worst-case execution time (WCET) analysis (a side benefit).
- Bound impact of change and requirements effect(s).

4.0 Common Benefits and Problems With Applying Data Coupling and Control Coupling Analyses

4.1 Benefits of Good Design and Integration Practices

Certification authorities have observed that applicants who specify a good design and have well-defined integration practices can identify and address data coupling and control coupling issues, and are able to:

- Provide a better awareness of functionality.
- Reduce the number of test cases needed to cover functionality and the supporting code structure, code interfaces, and requirements.
- Perform more efficient and effective change impact analysis.
- Find errors that are difficult to find in the lab testing and could be costly to fix in the field.
- Perform more effective maintenance.

4.2 Guidelines for Satisfying the Data Coupling and Control Coupling Analyses Objective

Certification authorities have observed a number of problems with the application of data coupling and control coupling analyses to airborne software. The guidelines below help to proactively address these issues:

- Applicants should address the data coupling and control coupling analyses in their plans (i.e., plan upfront how they will perform these analyses). In some cases, it may be distributed among several plans. However it is documented, it must provide complete and accurate rationale (i.e., it must be thorough).
- Applicants should consider data coupling and control coupling as part of their development/design effort (e.g., specify interface (I/O) requirements and dependencies between components).
- Applicants should substantiate their rationale for "analysis" and/or "testing" aspects of satisfying the objective. The objective may be satisfied as a static activity (e.g., looking at a link map or call tree), a dynamic activity (e.g., running tests), or a combination.
- Applicants should realize that data and control coupling analysis are two different analyses (i.e., they should not be combined into one activity).
- If tools are used, the determination of whether they need to be qualified or not should be evaluated and justified.
- If selective linkers are used, their effect on data coupling and control coupling must be analyzed.

5.0 Future Concerns

5.1 General Concerns

As software increases in size and complexity and applicants propose more partitioned and protected approaches, the quality of data coupling and control coupling analyses is a growing concern for certification authorities.

5.2 Concerns With Object-Oriented Technology

The increased use of object-oriented technology has also raised the certification authorities' concerns regarding analyses of data coupling and control coupling. A recent research effort sponsored by the FAA on object-oriented technology revealed data coupling and control coupling are further complicated by OOT (see *Issues Concerning the Structural Coverage of Object-Oriented Software*, FAA Report: DOT/FAA/AR-02/113). The report highlights, among other things, the following:

- OOT encourages the development of many small, simple methods to perform the services provided by a class (input to output transforms distributed throughout the code).
- OOT encourages hiding the details of the data representation (i.e., attributes) behind an abstract class interface (being able to access attributes only through methods makes the interaction between two or more objects implicit in the code).
- Most of the control flow is moved out of the source code through the use of polymorphism and dynamic binding. In essence, the control flow, and thereby the control coupling, becomes implicit in the source code, as opposed to explicit. There is a corresponding effect on data flow/coupling.

Because of these and other concerns (e.g., operator overloading in C++), the certification authorities are sponsoring further research in the area of data coupling and control coupling analyses of object-oriented design.

Note: Certification review items or issue papers may be used to address these and other OOT issues (see also CAST-8 and the FAA's *Handbook for Object-Oriented Technology in Aviation*).

5.3 Concerns from the Industry Brainstorming Session to Be Addressed in the Future

In 2003, a brainstorming session on the topic of data coupling and control coupling analyses was conducted. Nearly 50 people from industry and certification offices attended the session. The group indicated that they did not want CAST to provide prescriptive guidance on data coupling and control coupling at this time. Instead, they would like to see it addressed in the future by an industry committee (e.g., in DO-178C). Some of the following suggestions were made for future consideration:

- There needs to be an activity to define what is meant by "component" for data and control coupling. Some define a component as a module, some define it as a function, some define it as a pseudo-partition, some define it as executable files to be linked, and so forth. The granularity of "component" varies by implementation – it depends on the architecture being implemented. It would help to get some additional guidance in this area in the future. In the interim, it is suggested that applicants define what they mean by

"component" in their specific architecture, in order to better apply the current guidance of DO-178B/ED-12B.

- The topic of aliasing, as it relates to data coupling, should be addressed in future guidance.
- Perhaps it could be possible to define different degrees of data and control coupling analysis for Levels A, B, and C.
- A discussion on the impact of linkers and smart linkers, along with naming and searching algorithms, in the control coupling analysis area might help.
- Discussion on when data coupling analysis and control coupling analyses are most critical would be helpful. These activities help to identify things that might not be caught by traditional requirements-based testing (e.g., interrupts).
- The analyses get harder as systems grow in size. In the past, a control system may have been the only software embedded in a particular line replaceable unit (LRU). However, the current trend is to use operating systems and applications where data coupling and control coupling must be verified on different parts of the system and their connections verified as a whole. As we move to the increased use of partitioned systems, the coupling objectives between an application and the operating system that supports it are still applicable, but we may need to show a new means of verification. I.e., data and control decoupling between partitions.

5.4 Future Direction

CAST believes that additional guidance in the analyses of data coupling and control coupling will be required in the future. Research efforts and practical lessons learned will be used as input to the future guidance.

6.0 Summary

This paper has discussed the purpose, benefits, and challenges of data and control coupling coverage analyses (i.e., satisfying objective 8 of DO-178B/ED-12B Annex A Table A-7 for Levels A, B, and C). Additionally, some future concerns regarding analyses of data coupling and control coupling have been identified. CAST plans to continue monitoring this technical area and providing clarification, as needed.

ATTACHMENT #1

3.67 FAQ #67: What are data coupling and control coupling and how are they verified?

Reference: DO-178B/ED-12B: Sections 2.3.1 and 6.4.4.2, Table A-7 of Annex A, and Annex B

Keywords: data coupling; control coupling; verification; coupling

Answer:

The DO-178B/ED-12B glossary (reference Annex B of DO-178B/ED-12B) contains the following definitions:

"Data coupling - The dependence of a software component on data not exclusively under the control of that software component."

"Control coupling - The manner or degree by which one software component influences the execution of another software component."

An example of data coupling is a software component that utilizes parameters with a value that is calculated by a different software component, perhaps being executed at a different iteration rate.

An example of control coupling is a real-time software executive that initiates execution of a software component depending upon external parameters or influences.

Specific testing of data or control coupling is not a DO-178B/ED-12B objective, although this may be an acceptable approach to meet the related coverage objective. DO-178B/ED-12B does have an objective (reference Table A-7, objective 8) that data and control couplings are confirmed (reference Section 6.4.4.2c). This is typically accomplished by review and analysis of test cases and results. Data and control coupling are typically exercised during requirements-based testing (reference Section 6.4.3). Any deficiencies should be addressed by additional verification, as stated in Section 6.4.4.3 of DO-178B/ED-12B.

The verification of data and control coupling depends on the hardware/software interface, software architecture, code structure, and source code language. The verification of data and control coupling involves a combination of:

- Reviews and analysis of Software Architecture, as stated in DO-178B/ED-12B Sections 6.3.3b, 6.3.3c, and 6.3.3f;
- Reviews and analysis of Source Code, as stated in DO-178B/ED-12B Sections 6.3.4b and 6.3.4f; and
- Testing.