Exploring Frameworks and Representations for Domain Specific Automatic Code Generation

Scott Arthur Moody
Autocode Research Group
The Boeing Company
MS 6Y-07
Box 3999
Seattle, Washington 98124

email: scott@shuksan.boeing.com

Abstract

The paper is being used to present our concerns at the research session of the 4th Annual Software Reuse Workshop. We are trying to integrate reuse concepts into an environment that exhibits Domain Specific Reuse. The concept of application generators and automatic code generation utilizing graphical specification techniques must be explored, as well as possible formal language definitions. One of our biggest challenges will not be technical, but process oriented, as we try to transfer new technology onto working projects, including acceptance of reuse technology in safety critical systems.

Keywords: Reuse Frameworks, Production Quality Software, Automatic Code Generation, Embedded Avionics Software, Application Generators, Software Qualification Process

1 Introduction

This position paper is being used to discuss some of the research areas and concerns our group in Boeing is working on. We are called the Autocode Research Group, a program within the Avionics Software Engineering organization of BCAG (Boeing Commercial Airplane Group). Our charter is to explore the application of the latest technology in automatic code generation and software reuse to the problems faced in the existing BCAG avionics software development processes. Over the next few years, we plan to change our emphasis from research to actual process and product deliverables to be used in embedded software development.

Our goal will be to incorporate graphical application specification and generation tools into a development framework that utilizes domain specific software reuse and code generation techniques.

The end goal is the generation of 'production quality' embedded avionics software. This presents many complex problems that are also faced by the entire reuse community:

- Is the quality of the software that has been generated or reused up to the high standards required of production embedded software? This includes the issue of interfacing with embedded implementations such as runtime or I/O systems.
- What are the airplane qualification and certification issues dealing with software safety, verification and validation of any generated software especially when a tool has been used for the generation? Do we qualify the actual process; qualify the tools themselves, which then qualifies the software they generate; or generate and reuse code but still perform the normal qualification procedures?
- What is the best approach for integration of these new technologies into existing development processes and environments? Can a new approach be phased in while existing development is still being done?
- Can new processes and tools utilize the heterogeneous and embedded computing environments available within BCAG?
- What are the business issues involved with the cost and adoption of new processes?
- Who is responsible for the development and maintenance of reusable software, especially when dealing with many different suppliers?
- Will the new tools and methods be accepted by BCAG, the airline customers, or most importantly the FAA?

BCAG is a large company with numerous and diverse software development environments. In the commercial sector, production of a complex airplane must follow established procedures. The current process requires the designers of the avionics on-board control and display systems to generate detailed requirements documents which are typically sent to outside companies that specialize in embedded avionics software development. These large documents are used as the basis for the software development process. This includes other in-house tasks that range from analyzing contract bids to generating verification tests, and are even used as the basis for future product descriptions.

Because software is such a critical technology, our research charter has been to explore how automatic code generation techniques could be utilized to increase the productivity of the software development process. These are best directed at those areas within BCAG that contain detailed domain expertise.

We have been looking at methods that would augment the current development process with tools that would allow the system engineers the ability to specify their requirements utilizing graphic CAD/CAE like tools. The information specified in some of these requirements documents is of fine enough detail that automatic generation of software is achievable with current technology.

2 New Tools and Methods are Needed

BCAG's current practice uses tools and methods that do not exploit the latest electronic technology. Usually the system requirements are delivered in paper format, digested by the software developers who then write software specifications that satisfy the system requirements. Embedded software is ultimately produced that must then be verified against the original requirements and software specifications. We will explore methods of electronic specification transmission (or transmission of specific semantics), reducing possible errors, followed by methods for automatic generation of the specifications themselves as more electronic representation tools are utilized.

Our vision plans for developing a framework where prototyping and other design testing concepts can become an integral part of the design and requirements document generation. Our vision should make use of active electronic books, both as more usable documentation and for better traceability in automatic generation of production quality software. This can be achieved if the required tools are integrated into an accepted framework.

We plan to develop a prototype that tests concepts for infusion into the current process. As concepts are proven, newer processes can be devised that make the most benefit of the newest technology. These will lead towards more consistent specification representations especially if automatic code generation is to be achieved.

3 Reuse Development Framework

We are exploring how best to utilize current reuse and automatic code generation technology (e.g. application generators). Our approach is to introduce a reuse development framework supporting the different tasks of software development, from requirements specification through design development, testing, and verification. Throughout this process domain capture and domain analysis techniques should be utilized.

3.1 Object Oriented Reuse Development Frameworks

How can existing tools be connected in new and powerful ways? In our domain this might include utilizing an existing code analysis tool as the input to a code generation system. New open architecture frameworks are being developed by the main vendors such as the CASE Integration Services and the PCTE (Portable Common Tool Environment). These are based on the object oriented paradigm utilizing type hierarchies and supporting interactive prototyping environments and inter-process or inter-tool communication. Formalization of these frameworks and of the management decisions for their adoption are being explored.

3.2 Common Interface Environments

We should utilize common interface environments which support heterogeneous platforms and provide higher leverage for reuse libraries. BCAG is actively participating in APEX (Application Software Executive interface) proposed by the AEEC (Airlines Electronic Engineering Committee). Other important work includes CIFO (Catalogue of Interface Features and Options) for the Ada

Runtime Environment. This result of this effort should be a specification for a runtime executive that can be utilized by code generation techniques and lead to a FAA certified product.

3.3 Reusable Software Libraries

Our goal is not to generate prototype software, but instead to generate production quality software as part of the requirements documentation phase. This raises numerous problems that we feel are not being sufficiently addressed, or sufficiently answered. What is the definition of production quality real-time software? What level of prior domain knowledge must be available to automate code generation? If software is being reused from a library, can it be trusted? Can previously accepted software be used in a different way such as using different generic parameters without invalidating this acceptance? How can technology be transferred between BCAG and their suppliers in a way that leverages off the available expertise and exhibits improved efficiency? Is it possible to obtain service credit on previous automated tool use? The draft DO-178B document addresses many of these issues.

3.4 Electronic Hyper-linked Books

The software design and development process should incorporate dynamic electronic books for tracking design to requirements and other traceability between the development phases. It should incorporate the dynamic nature of prototypes and design testing performed during the development of the specification. Consideration should be made for incorporating these active and living documents into the delivered specification product.

BCAG has been utilizing some of this technology in different portions of the airplane development process. These range from CAD/CAM/CAE tools for designing the structures, to the REDAR project where on-line access to all engineering drawings is available throughout the production facilities. We are looking at ways of transferring this technology into the software engineering field.

4 Airborne Systems Considerations

Software systems developed for airborne applications must meet very detailed specifications as detailed in the draft FAA document, DO-178A, "Software Considerations in Airborne Systems and Equipment Certification". At least three important areas dealing with software reuse and automatic code generation must be addressed:

- Systems Safety
- Software Testing
- Verification

4.1 System Safety Issues

Use of techniques and methods in the development process, including automation tools, must follow specific qualification criteria that ensure a level of safety that is in compliance with airworthiness requirements. Any tool that is used by this process must be qualified when its use could introduce or fail to reveal errors in the airborne software. This is especially true when the output of this tool is not subject to a full verification suite each time the tool is used. This directly effects candidate development tools ranging from compilers to reuse of commercial-off-the-shelf software (COTS), as well as graphical code generation techniques.

4.2 Software Testing Philosophy

Since testing is derived from the software requirements document working with electronic representations should be beneficial. As automatic code generation techniques are applied, how these certification credits are derived and accumulated must be examined.

4.3 Verification Criteria for Reused Software

Verification is the process of checking the software against the requirements. Reusing system development information is currently constrained to how it was previously used. Reuse and automatic generation of software will have a major impact on the verification process.

5 Autocode Status

The Autocode project has been spending the last 6 months studying the current practices involved in developing requirements documents. This involved classifying the different types of systems along with the different airplane system architectures and how current and perceived tools could be applied (e.g. displays, controls, etc).

Our study has analyzed Specification Control Drawings (SCD), which are the contractual requirements document, and broke them down into different representations used and technologies we perceive could be applied to them. Current specification representations include: structured and unstructured English, data tables, flow charts, PDL, state transition diagrams, and control law block diagrams. We now plan to map existing or proposed tools into these representations in a way that is both acceptable, and one that exploits software reuse libraries and code generation techniques.

Fostering reuse at BCAG is a very difficult problem but the Autocode Research Group feels the approach we are taking, solving complex but domain specific applications, should help lay a foundation for a Boeing wide reuse effort.

References

[APEX 91] Draft 1 of Project 653, "Avionics Application Software Standard Interface", AEEC, Airlines Electronic Engineering Committee, October 3, 1991.

[CIFO 91] [CIFO] "Catalogue of Interface Features and Options for the Ada Runtime Environment", in Ada Letters, Vol XI, No. 8. Fall 1991.

[178B] Fourth Draft of Proposed Revision to DO-178A/ED-12A "Software Considerations in Airborne Systems and Equipment Certifications.", Requirements and Technical Concepts for Aviation, 1140 Connecticut Ave, N.W., Suite 1020 Washington, D.C. 200036

[Weye 85] S. Weyer, A. Borning. "A Prototype Electronic Encyclopedia", ACM Transactions on Office Information Systems Vol 3, No. 1, Jan 1985

[CONO 91] US40 Domain Specific Environment/Repository STARS Reuse Concept of Operations, Volume 1 Version 0.5 DRAFT #GR-7670-(NP), 30 August 1991

[Burt 87] B. Burton, R. Aragon, S. Bailey, K. Koehler, L, Mayes, "The Reusable Software Library", IEEE Software, July 1987.

[Andr 88] K. Andrews, R. Henrey, W. Yamanoto, "Design and Implementation of the UW Illustrated Compiler", Proceedings of the SIGPLAN '88 Conference on Programming Language Design and Implementation, June 1988

[Engl 90] D. Engelbart, "Knowledge-Domain Interoperability and an Open Hyperdocument System", Proceedings of the Conference on Computer Supported Cooperative Work, October 1990

6 About the Author

Scott A. Moody is a senior software engineer researching automatic code generation concepts for the Boeing Commercial Airplane Group. His primary work has been in designing a framework that promotes reuse of tools, domain knowledge and production quality software.

Scott was previously the lead engineer on the RAPID project, a collection of rapid prototyping tools for generic C3I graphics workstations. He worked on automatic code generation techniques transfering the prototypes into requirements and code.

Scott's research interests include information reuse frameworks, code generation techniques, multi-media technology, distributed graphical prototypes, and application generators utilizing reusable software technology.

Scott received an MS and BS in computer science from the University of Washington. He is a member of the ACM and IEEE Computer Society.