Provisioning of Highly Reliable Real-Time Systems

Harold Lawson, Kurt-Lennart Lundbäck

Provisioning of Highly Reliable Real-Time Systems

Harold (Bud) Lawson[1] and Kurt-Lennart Lundbäck[2]

[1] Lawson Konsult AB, Albavägen 25, 181 33 Lidingö, Sweden
bud@lawson.se
[2] Arcticus Systems, Box 530, 175 26 Järfälla, Sweden
kurt.lundback@arcticus-systems.com

Abstract. The development and deployment of highly reliable real-time systems is an important technical as well as societal concern. Many systems are dependent upon timely and correction function; for example in key automotive components, aircraft components, military equipment, and so on. Based upon some early architectural developments in Automatic Train Control, collaboration has led to further development of the concepts and products for highly reliable real time systems. Arcticus Systems has provided products based upon the concepts for several Swedish system suppliers in various branches. In this paper, the history of these developments is described, the consequences of a missed opportunity to take a leading role in the automotive branch are discussed and potential future directions are presented.

Keywords: Automatic train control, real time systems, safety-critical, vehicle components and systems

1 Introduction

In this paper, experiences with the provisioning of highly reliable real-time systems are described from the perspective of two small actors; namely Arcticus Systems AB and Lawson Konsult AB. During the 1970s, Harold Lawson was an architect for the software system of the world's first microprocessor based Automatic Train Control (ATC) system delivered to the Swedish Railways (Statens Järnvägar, SJ). The concepts and principles upon which this design was based proved to be a highly reliable and extremely small robust real-time solution. In 1990, a project was started by Mecel AB to explore how to develop a distributed real-time architecture for vehicles. Nutek supported this project and it brought together several companies and university researchers. It was at this point that co-operation between Lawson Konsult AB and Arcticus Systems AB began. Since then Arcticus Systems have continued to refine the concepts and have delivered their real time operating system Rubus OS to amongst others, Haldex, Volvo Construction Equipment and Hägglunds. Further, Arcitcus Systems has had a continual relationship to research work in this area with Mälardalen University.

2 Automatic Train Control

The experiences with the Automatic Train Control (ATC) system have been described in two publications [1] and [2] the latter of which was presented as a keynote address at HINC2. Only a brief summary is provided here. In 1975, the consultant services of Harold Lawson were contracted by Standard Radio to assist Sivert Wallin, chief designer, in the conceptualization of the ATC architecture. Following a review of the work done to date on the software, Harold Lawson and Sivert Wallin re-examined the fundamental requirements of the ATC function and developed the problem oriented architectural concepts that has successfully provided product stability as well as a sound basis for further development under the entire life cycle of the ATC on-board system product. The following three core concepts were developed and have been driving factors during the product life cycle.

Time Driven: The major conceptual aspect of the design is the treatment of the system as being continuous in time as opposed to being discrete event driven. This was motivated given the fact that a 250 ms resolution (*dT*) of the state of the train in respect to its environment was determined to be sufficient to maintain stability. It became clear that the simplest approach was to execute all relevant processes (procedures) simply during this period.

Software Circuits: As the result of the time driven concept, a cyclic approach became the basis for the solution where short well-defined software procedures performing required transformations behave like timed circuits. The naming of this concept was developed later when the concepts of the architecture were applied in the research project described later in this paper.

Black-Board Memory: In order for Software Circuits to have access to key status information, variables are retained in a black-board where both reading and writing are permitted.

This simplification of concepts led to the fact that the processors only needed to be interrupted by two events. One interrupt to keep track of time (1 ms) and one interrupt when information from transponders (located on the track) is available. The 250 ms *dT* is more than adequate to perform all processing. Adding more structure to the problem, for example, via the use of an event driven operating system approach would have had negative consequences in terms of complexity, cost as well as reliability, testability and risk thus affecting safety.

The “circuit like” structure of software led to highly simplified coding of processes (procedures). While it would have been useful to deploy a higher-level language in the solution, it was deemed unnecessary due to the low volume of code that was expected. Experience has indicated that this was a reasonable decision at that time. On the other hand, it was decided to comment the code in a higher-level language. In earlier versions of the product, the Motorola MPL (a PL/I derivative) was employed. In later versions, a more Pascal like annotation has been consistently employed. In system tests, MPL, respectively Pascal versions have been executed in parallel with the execution of the assembly language version in order to achieve system verification.

The Standard Radio developed ATC system has been functioning in most all locomotives in Sweden since the early 1980s. Ansaldo of Italy now owns this on-

board system. It has been implemented in a few new versions reflecting special needs that developed including the introduction of the X2000 high speed trains as well as the need to run trains over the Öresund Bridge to Copenhagen. The modifications continued to apply the concepts that were earlier developed. We can conclude that there has never been a train accident that could be attributed to faults in this control system. Ansaldo utilized the concepts from this original solution in a product based upon the Ada programming language that was provisioned to and is still operating on New Jersey Transit in the USA.

3 Early Arcticus Products

Arcticus Systems originated in 1985 and its first product called O'Tool was influenced of the rendezvous concept of the Ada programming language. At that time, it was a state of the art solution for Real Time Operating System (RTOS). Several languages had evolved during the 1970s including Pascal and C, followed by Ada that provided for real time task management. Another language that was modified in order to provide RTOS capabilities was EriPascal (developed by Ericsson).

O'Tool became a viable RTOS solution for microcontrollers implemented in the C programming language. This development transpired in close cooperation with IAR Systems in Uppsala that was a pioneer in applying C in microcontrollers and where worlds leading at that time. An increasing number of microcontroller applications (embedded systems) where developed and the market grow quickly in the late 1980s and early 1990s.

Many system developers where not accustomed to utilizing an RTOS and experienced problems in respect to excessive overhead and especially with fault diagnosis. RTOS technology in general and even O'Tool provided event driven functionality. An external event provided the trigger to initiate a task/function that after processing decided about reaction.

Event driven systems have the difficulty of not being able to guarantee response time and behavior, especially in the case where multiple events happen in close proximity or in a sequence that was not planned for in the system solution. We where often involved in customer contact to resolve faults in their applications. Thus, our experience from customers convinced us that reliable event driven systems are not achievable via event driven solutions.

The need for university education in respect to RTOS systems grew during the late 1980s and O'Tool, due to its small size became popular in courses at the KTH Mechatronic institution as well as at Mälardalens University.

IAR Systems marketed O'Tool world wide as a complement to their C compiler. Thus, we had several customers from various countries.

4 Collaboration Leading to New Concepts

Via an agreement that Arcticus had with Mecel in 1990–91, we participated in an EU research project Prometheus that was a supported by amongst others, Renault. This resulted in a preliminary implementation of an RTOS concept called VDX (Vehicle Dynamic Executive) based upon communication on a CAN/VAN-bus. VDX was a traditional event driven RTOS that had the problems that were discussed earlier.

Based upon Mecel's and our own experience and analysis and even based upon the VDX project it was concluded that embedded real time systems should not be based upon an event driven approach. Such systems tend to be complex and difficult to guarantee their behavior. They concluded that the RTOS contributed significantly to the systems complexity.

Mecel together with a number of small companies including Lawson Konsult and Arcticus Systems as well as researchers from Chalmers, SICS and Uppsala University initiated a research project for constructing of distributed safety-critical vehicle system as a part of the Nutek Swedish Road Traffic Informatics program. The project was called the Vehicle Internal Architecture (VIA) and it transpired between 1992 and 1995. The project budget was about twenty million SEK.

4.1 Vehicle Internal Architecture Project

Based upon experiences from the Automatic Train Control system as well as research performed at the Vienna Technical University under the direction of Professor Herman Kopetz, the project group defined an architecture based upon time synchronized communication and execution (Time Triggered execution model) in a distributed environment. The project incorporated the fundamental program architecture for the construction of safety-critical systems and related methodology. Even hardware construction for safety-critical systems was considered in the scope of the project.

Now, fifteen years later we can see that the VIA architecture is very actual and that the vehicle industry has begun to implement according to the concepts that were developed earlier. Within a few years, distributed safety-critical functions will be put into mass production.

In addition to Time Triggered execution, Event Driven execution was incorporated in the VIA architecture for parts of the application that only had a soft real time demand.

4.2 Red and Blue

Our colleague Professor Jan Torin from Chalmers was drawing a picture to illustrate the difference between the Time Triggered and Event Driven functions. He happened to use Red and Blue markers and so these two categories became differentiated via color. This differentiation has continued and been extended with a green color in the Rubus OS. Rubus OS provide three categories of run-time services:

Green Run-Time Services. External event triggered execution (interrupts).

Red Run-Time Services. Time triggered execution, mainly to be used for functions that have hard real-time requirements, i.e. meeting their deadlines is critical to the operation.

Blue Run-Time Services. Internal event triggered execution, to be used mainly for functions that have soft real-time requirements, i.e. meeting their deadlines is not critical to the operation.

4.3 Software Circuits

The implementation of small short procedures performing transformations that was integral in the ATC application carried over into the implementation of the Time Triggered functions defined in the VIA project and in the Rubus OS.

Due to their timed nature and the transformation properties, they behave much like timed hardware functions and thus in the VIA project where given the name Software Circuits.

4.4 Distributed Control

An important aspect of the VIA architecture was the introduction of distributed control; that is, no coordinating centralized function synchronized activities. Synchronization occurs due to the time functions provided by the communication bus. There are now several standards and products based upon this concept such as FlexRay, TTCAN, and TTP. The results of the VIA project where reported in two international publications [3] and [4].

5 New Arcticus Products

The positive experiences with ATC as well as the VIA project has led to the development of new products. Thus, our history has influenced further developments.

5.1 Haldex Four Wheel Drive Coupling Device

Haldex made an agreement in 1996 with Volkswagen to provide a new type of product called the Limited Slip Coupling Device for four-wheel drive vehicles. Since Haldex AB did not have the required resources and competence to construct the hardware and software of this control system, and at the suggestion of project leader Anders Cedeberg, Arcticus AB was contracted to provide the software platform based upon Rubus OS. The role for Arcticus was to construct a platform for the selected 16-bit microprocessor from Infineon with drive routines for the I/O units that where

incorporated. Arcticus also developed a simulator for the functional testing in a PC environment.

Arcticus and Lawson Konsult participated in the discussions with Volkswagen (the acquirer of the product) concerning architecture and implementation that contributed to the fact that Haldex was selected to supply the base platform. Originally, Volkswagen's position was that Haldex would only deliver the mechanical parts and hardware electronics.

Because Arcticus and Lawson Konsult had a central role, we convinced Haldex to develop a set of processes based upon the ISO/IEC 12207 standard on software life cycle processes as well as a Safety Case indicating that the product and its development were verifiable.

Haldex having a long history of developing mechanical products and was not accustomed to this new type of product development. This resulted in the fact that budgeted development costs and product planning were not realistic and caused problems for the project and Haldex corporate leaders due to delays. After a few years though Haldex could demonstrate the functioning coupling device and eventually received confidence from their customer (Volkswagen) to also take responsibility for the control functions that were implemented in software.

Haldex have now gone through five generations of the coupling device for Volkswagen and have many other automotive manufacturers as customers. Rubus OS with associated development environment are still utilized and continue to function in an excellent manner.

5.2 Model Based Development for Volvo Construction Equipment

Volvo Construction Equipment, like Haldex, had limited resources in the mid 1990s for the construction of software based control systems. Volvo CE selected to base their software development upon Rubus OS and to build a software platform based upon the Rubus concept. Arcticus responsibility was to assist in the construction of a platform for the selected 16-bit microprocessor from Infineon as well as the associated I/O units.

Approximately 1997 Volvo CE decided to promote a component based development model that was developed by researchers at Mälardalans University and successively has been further improved by Arcticus as the Rubus Component Model. To support this development a language and compiler was developed that automatically generates an execution schema for the Red Kernel. Both the component model and the related tools were further developed into a second and third generation release around 2000 and 2008, respectively.

The generation of products containing ECU (Electronic Control Units) developed by Volvo CE since 1997 have been based upon the Rubus component model and its development environment. The environment provides a graphical representation with data flow between software circuit input and output ports.

The unique property of the Rubus component model and related development tools is that non-functional requirements like deadline controller and automatic schema generation including the program execution disturbance caused by interrupts are considered in the analysis.

Volvo CE has via the Rubus Component Model and tools been able to exploit nearly 100 percent of the theoretic capacity of the processor that is utilized.

Arcticus has participated in a research project at Mälardalens University and Volvo Construction Equipment aimed at refining the component model that was developed for Volvo to even include Event Driven programs in the model. The project was called MultEx and was performed between 2005 and 2008. Arcticus has now after fifteen years of model based development in industrial environments a combined structural modeling with integration of function modeling, for example, SIMLINK and MathLab, plus a unique execution model.

5.3 Further Development of the Component Model for Hägglunds

The military system provider Hägglunds decided in 1996 to base its real time software architecture upon the Rubus OS. In 2007, Hägglunds decided to work towards model-based development based upon our MultEx project and this resulted in the use of Rubus Component Model, version 3. Thus, the tool was restructured for supporting Rubus CM3 as well as to handle many of the problems from our ten years of experience with Volvo CE. The first version of the tool and model was delivered to Hägglunds in 2008 and has been further refined in cooperation with Hägglunds. This forms the basis for their new platform for military equipment based upon Rubus products.

Together with Hägglunds we now participate in a project that is aimed at supporting the analysis of communication between network nodes called EEMDEF (2009–2012) sponsored by KK-Stiftelsen (the Knowledge Foundation). This work will lead to the next generation of component model; namely Rubus Component Model 4.

This vision of EEMDEF is to develop a framework that allows modeling and analysis of execution requirements (e.g. response times, deadlines, jitter, memory consumption and other metrics relevant for control systems) on an abstraction level that is close to the functional specification (i.e. abstracting away implementation details such as physical and logical allocation of functionality).

5.4 Industry-Academic Research

Arcticus continuously has cooperated with state of the art research programs at Mälardalen University. This has provided a strong input for the development of Arcticus products and also provided the research project with practical relevant industrial requirements. For further information about this collaboration, consult; http://www.mrtc.mdh.se/projects/multex/ and
http://www.mrtc.mdh.se/projects/eemdef/.

6 A Missed Opportunity

Interestingly that more than fifteen years after the VIA architecture project the architectural features developed then are currently in focus. Some reflections concerning this development are as follows.

- Development proceeds slowly with a time lag of perhaps more than 10 years from research to practice for embedded system development companies. It is a question of developer competence and corporate leadership, all the way from initial education to the time that they progress to decision-making positions in a development project.
- It takes time to develop commercial products that achieve customer acceptance (10+ years). This concept is especially for a small company that provides a new type of product that is unproven and does not follow existing standards.
- Combating the "Not Invented Here" (NIH) phenomenon.
- Sweden is a leading nation in the area of real-time research and we can retain this competence lead and amplify it. If we do not this, many Swedish inventions will die out or move to other countries. We seem to be poor at commercializing and supporting the excellent research and development efforts.
- Large companies like standards. In many cases, standards are sought that do not provide technical advantages. If we follow the standard, we cannot go wrong and take no chances even if there is a better technical alternative.

7 Conclusion

Think if the VIA project had been pushed forward in Sweden. The opportunity existed in the mid 1990s. We see that European standards today are largely based upon the German auto industry such as OSEK and Autosar for real-time systems that are built upon the traditional event-driven solution. Sweden could have had this leading role.

References

1. Lawson, H., Wallin, S., Bryntse, B., Friman, B.: Twenty Years of Safe Train Control in Sweden, Proceedings of the International Symposium and Workshop on Systems Engineering of Computer Based Systems, Washington, DC (April 2001)
2. Lawson, H.: Provisioning of Safe Train Control in Nordic Countries, Keynote address appearing in the Proceedings of HiNC2, History of Nordic Computing (2008)
3. Hansson, H., Lawson, H., Strömberg, M., Larsson, S.: BASEMENT: A Distributed Real-Time Architecture for Vehicle Applications, Proceedings of the IEEE Real-Time

Applications Symposium, Chicago, IL, May 1995. Also appearing in Real Time Systems, The International Journal of Time-Critical Computing Systems, vol. 11, no. 3 (1996)
4. Hansson, H., Lawson, H., Bridal, O., Ericsson, C., Larsson, S., Lön, H., Strömberg, M.: BASEMENT: An Architecture and Methodology for Distributed Automotive Real-Time Systems, IEEE Transactions on Computers, vol. 46, no. 9 (1996)