

---

# SAE Embedded Software Activities Update

Bruce Emaus

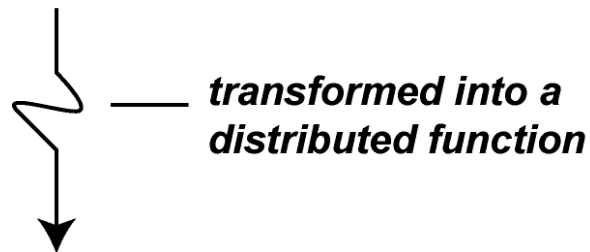
- Chairperson of SAE Embedded Software Task Force

Bob Gruszczynski

- Chairperson of SAE J2632 – C Coding Practice

# Origin - from the distributed function

## *Transformation into the Distributed Form*

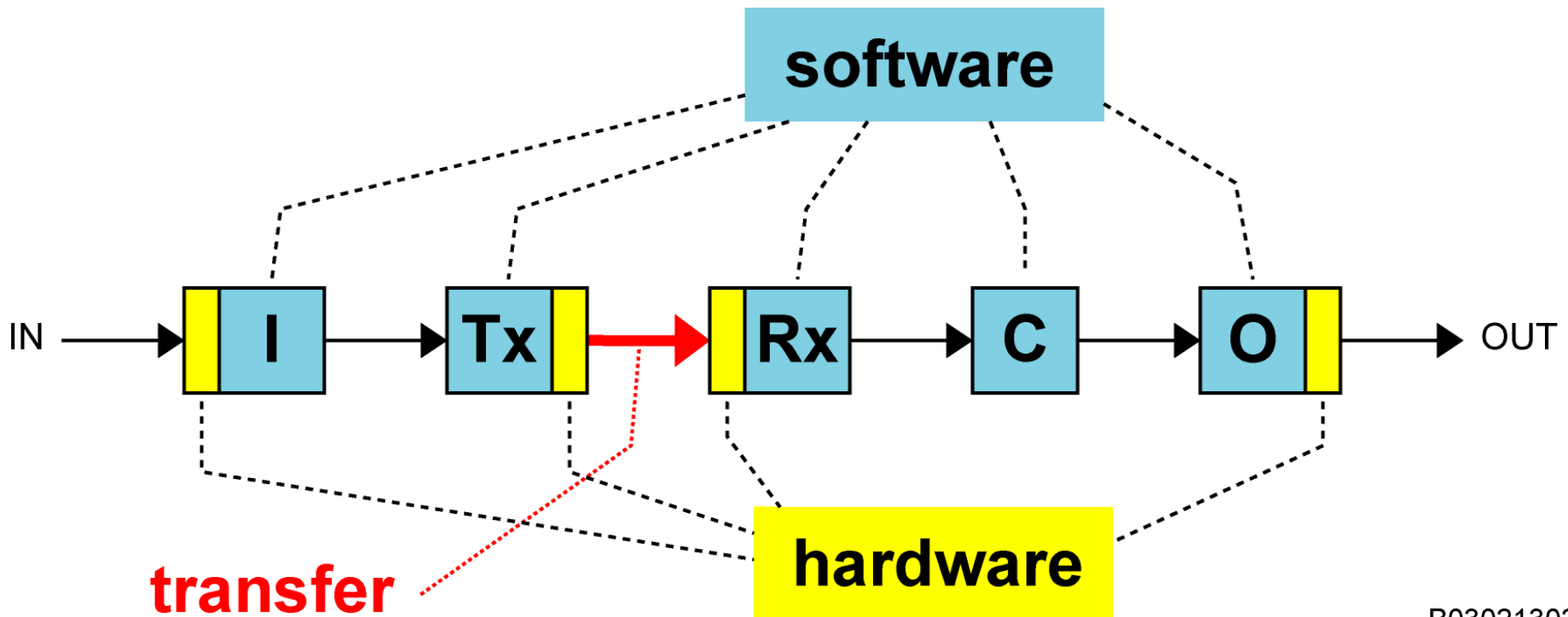


B03021301

*Note - This model uses an arbitrary partitioning boundary selected between the Input and Control blocks - other partitioning is possible.*

# Software in the distributed function

- ◆ Software establishes the speed of operation – not protocol transfer speed



B03021302

# SAE Embedded Software Task Force - GOALS

- ◆ PURPOSE -
- ◆ “to reduce distributed embedded system cost and development time through efficiency in software”
- ◆ TARGET GOALS –
- ◆ Increase the quality of delivered software
- ◆ Raise industry awareness of software design issues
- ◆ Reduce software development lifecycle cost and time
- ◆ Raise industry awareness of issues that pertain to the software development lifecycle
- ◆ Develop appropriate recommended practices or standards for distributed embedded software design
- ◆ Raise industry awareness of issues that pertain to safety-critical and mission-critical software
- ◆ Improve the general level of education across the automotive software engineering community

# SAE Embedded Software Task Force - GOALS

More TARGET GOALS...

- ◆ Increase efficiency in the procurement of software
- ◆ Optimize the software development lifecycle shared between the OEM and supplier
- ◆ Minimize mistakes made by entry-level software engineers
- ◆ Decrease the cost of software development through reusable software building blocks and algorithms
- ◆ Develop common methods for task scheduling, I/O handling, loss of communications, etc.
- ◆ Develop generic distributed embedded software requirements
- ◆ Raise industry awareness of issues that pertain to the system - the distributed embedded system
- ◆ Develop common methodologies for vehicle network software (if OEM supported)
- ◆ Develop common software testing methods

# Major Related SAE ES Activities

- ◆ SAE Embedded Software Task Force
- ◆ J2516 – Embedded Software Development Lifecycle
- ◆ J2632 – C Coding Practice
- ◆ J2640 – Generic Requirements for Embedded Software
- ◆ Addressing Major Automotive Software & System Issues
- ◆ Distributed Embedded Systems Focus
- ◆ Upcoming X-by-Wire Industry
- ◆ Impact of Tread Act on Automotive Software
- ◆ Embedded Software Presentation Series
- ◆ Distributed Embedded Systems Presentation Series

# J2632 – C Coding Practice

---

---

# J2632 C Coding Practice

By Bob Gruszczynski  
- Visteon Corporation

# Scope

---

- ◆ Recommendation of the minimum set of best practices and lessons learned for the overall process of creating C code.
- ◆ Concentrates on specific C coding practices that the software must adhere to.
- ◆ Requires support from both the OEM and Supplier software engineering communities.

# Goals

---

- ◆ Increased data integrity
- ◆ Increased program flow integrity
- ◆ Increasing the level of determinism in embedded software
- ◆ Increasing the software engineering education level

# Overview

The relationship between the OEM and the supplier is reasonably well defined for both hardware and packaging requirements. The relationship between the OEM (or software buyer) and the supplier is not as well defined for software code requirements. As a first step to defining these requirements, a 'C' Coding standard will put in place a framework for the consistent development of error-free, maintainable code. This Recommended Practice is established to:

- ◆ Increase software quality by establishing a best practice and lessons learned document that is applied industry wide.
- ◆ Raise the experience level of the embedded software community and minimize errors made by all involved engineers.
- ◆ Provide a format and a process for the OEM to grant an exception to the Recommended Practice.
- ◆ Allow a process to change the Recommended Practice.

# MISRA

---

- ◆ Motor Industry Software Reliability Association
- ◆ Based in Europe
- ◆ Recommends NOT using C language for Embedded Software
- ◆ First to try to provide Industry-wide guidelines for C Coding
- ◆ First Document has 127 “Rules”
- ◆ Some rules are contradictory or difficult to understand and implement
- ◆ Revision is in progress, due this year

## MISRA (Cont.)

---

- ◆ MISRA Compliance can be statically checked using off-the-shelf tools
- ◆ Ford Motor Company is using compliance checking tool to check a limited set of rules
- ◆ Visteon Corporation has a phase-in plan

# J2632 additions

---

- ◆ MISRA is a good start
- ◆ Lessons learned
- ◆ Good coding practice ideas

# Authoring Code Requirements

---

- ◆ Use of include files
- ◆ Use of comparison values
- ◆ Use of conditional statements
- ◆ Use of assembly language

# Multiple Translation Unit Requirements

---

- ◆ Scope of variables/global variables
- ◆ Interface definition
- ◆ Use of duplicate identifiers
- ◆ Use of header files

# Coding for Multitasking

---

- ◆ Use of reentrancy
- ◆ Use of data coherency
- ◆ Use of hard coded "magic numbers"

# Coding for Hardware

---

- ◆ Hardware interface definition
- ◆ Use of interrupts
- ◆ Use of hardware resource sharing

# Tools

---

- ◆ Use of compilers/linkers
- ◆ Use of C Code checking tools

# Exception Process

---

- ◆ Why allow exceptions?
- ◆ How to make exceptions to the recommended practice

# Summary

---

- ◆ J2632 – recommended practice
- ◆ Combines industry-wide knowledge
- ◆ Provides educational forum

# J2516 – ES Development Lifecycle

---

- ◆ Re-uses all available resources
  - ◆ nothing new is added
  - ◆ No new invention of a process
- ◆ Embraces – “have a process”
- ◆ New focus – “attain CMM-2 or equivalent”
- ◆ Re-writing to encompass – European SPICE

# J2640 – Generic Requirements for ES

- ◆ Based on Ford experiences
  - ◆ New software engineers and engineers without automotive experience are creating a large number of errors that show up during module level testing at the car company
  - ◆ How can this continuing trend be stopped?
- ◆ Establish general purpose requirements
- ◆ Microcomputer resources
- ◆ Interrupt processing
- ◆ I/O processing
- ◆ EE memory use

# The Big Three – Automotive Software Activities

- ◆ Scheduler
- ◆ Application
- ◆ Network



# Some Interesting Industry Level Trends

---

- ◆ GM
- ◆ Ford
- ◆ DCA

# Today's success – is not the future

---

- ◆ Desired module performance with little to no specifications
- ◆ Countless “home runs” have been hit
- ◆ This miracle continues occurring on a regular basis

# Software Specifications

---

- ◆ Where are the software specifications?
  - ◆ System and feature descriptions are not enough
- ◆ How do we capture the automotive environment?
- ◆ What standards can we use?
- ◆ Tools will come – but what is needed?
- ◆ Can UML fit in?

# More software is coming

---

- ◆ Smart Sensor/Smart Actuators
- ◆ Air bag subsystems
- ◆ Adaptive cruise/braking subsystems
- ◆ Distributed powertrain subsystems
- ◆ Brake-by-wire subsystems
- ◆ Steer-by-wire subsystems
- ◆ More gateways will be needed

# Automotive software engineering is serious business

---

- ◆ Automotive software engineers will take the stand in the courtroom – perhaps in Washington
- ◆ Phrases like “It’s only software” or “software is free” must stop
- ◆ Automotive software must become more responsible
- ◆ Automotive software must address the important issues

# Emerging X-by-wire software

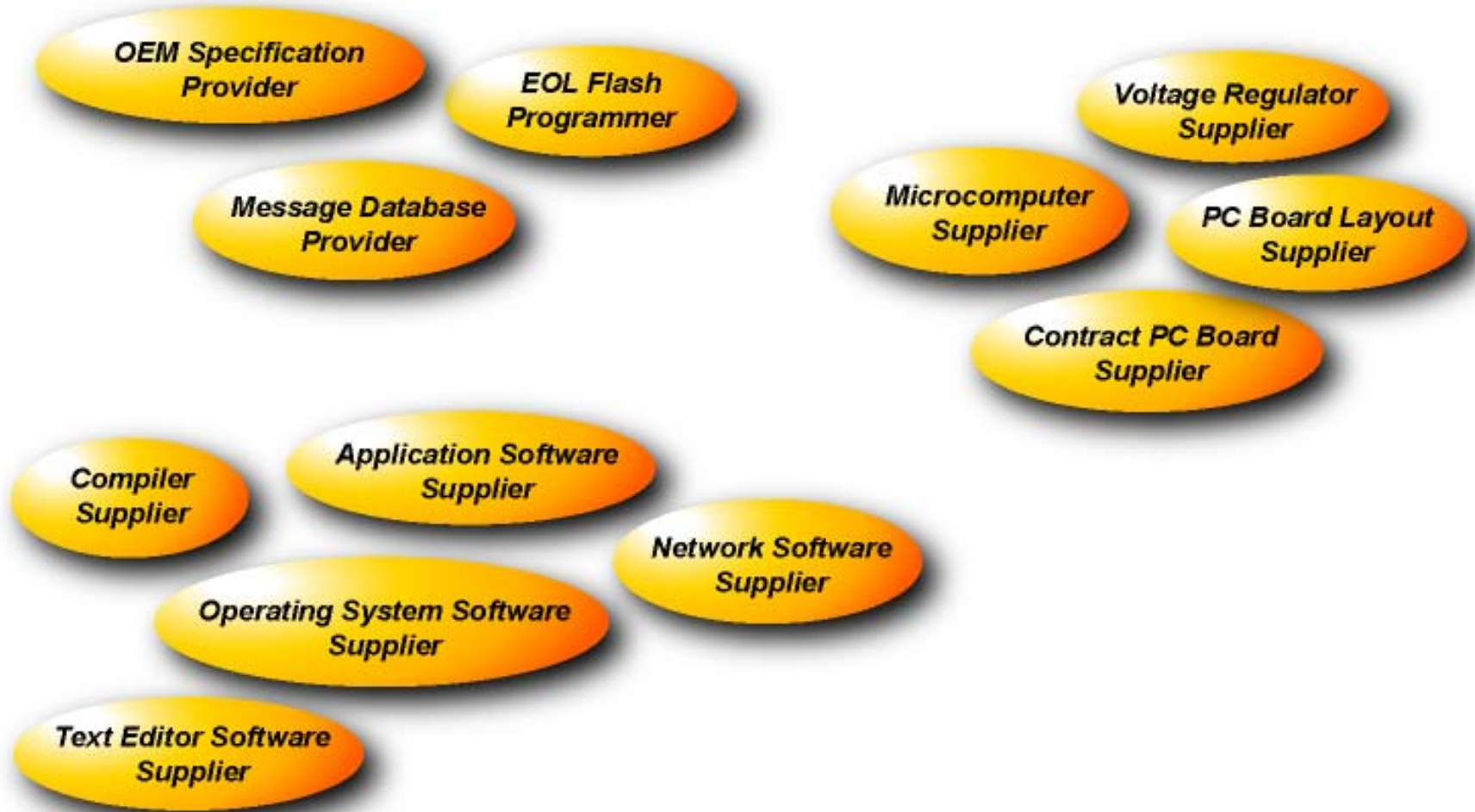
- ◆ X-by-wire requires software
- ◆ Will the software be “redundant”?
- ◆ What type of software compliance will be required?
  - ◆ DO-178B Compliant?
- ◆ What about the language choice?
  - ◆ ADA – SPARK – C
- ◆ Will some reliability number be attached to the software portion?
- ◆ Who knows how to compute software reliability?
- ◆ This effort requires some intense attention

# Legal Issues

---

- ◆ Does distributed software & systems mean distributed liability?
- ◆ Will upcoming TREAD ACT impact automotive software?

# Who is responsible?



B00021310

# ADD - the systems perspective

---

- ◆ Look at the systems side of software
- ◆ Learn the systems side
- ◆ Understand the business & legal consequences of technical actions

# Network strategy continues to become competitively irrelevant

---

- ◆ Who really cares about one strategy over another?
- ◆ Who really can make the comparison of one strategy to another?
- ◆ What business advantage is there to having your own strategy?
- ◆ Ford and GM continue work toward a common strategy

# One Common Automotive Network Solution

---

- ◆ Move away from proprietary
- ◆ Networking strategy will become more off-the-shelf
- ◆ Create a common terminology
- ◆ Create a set of common methods
- ◆ Provide an appropriate amount of variety
- ◆ Selectable, scalable, and configurable
- ◆ Application-to-Network API will narrow and become more efficient

# What the Auto Industry Wants from Software

---

- ◆ More emphasis on industry wide solutions
- ◆ More emphasis on business rather than technical
- ◆ More emphasis on increasing quality and lowering cost
- ◆ More emphasis on time to market reduction
- ◆ More emphasis on standards and recommended practices

# How we get there – guiding principles

---

- ◆ Adequate - Perfect
- ◆ Industry importance – local importance
- ◆ Emotional – Non-emotional
- ◆ Business first, technical second
- ◆ Thinking “our” solution, not “my” solution
- ◆ Participate

# Embedded Software – (draft) Bill of Rights

- ◆ WHY does software need something like this?
  - ◆ HELPS TO CHANGE THE MINDSET
- ◆ KEY POINTS...
- ◆ No person, company, or organization shall discriminate against embedded software
- ◆ Software must be treated the same as hardware
- ◆ Software is not free – just like hardware is not free
- ◆ Software has quality processes – just like hardware
- ◆ Software is as reliable as hardware or any of the materials near the software

# GLSPIN – SAE Involvement

- ◆ Where is the overall CMM effort moving?
- ◆ SAE Embedded Software needs to help many people to understand this agenda at two key levels
  - ◆ Introductory Level
  - ◆ Detailed Level
- ◆ Could someone from GLSPIN speak once a year at our “Presentation Series” meetings
- ◆ SAE Embedded Software would encourage GLSPIN to create an invitation email for automotive software engineers to join GLSPIN
  - ◆ We could send this out once a year.

# Get involved...

---

- ◆ If you are interested in having your name added to the mail list for either the SAE Embedded Software effort or the SAE Distributed Embedded Systems Engineering effort, please send an email to [bruce.emaus@vector-cantech.com](mailto:bruce.emaus@vector-cantech.com)

# Email List

---

- ◆ To be added to the SAE Embedded Software Task Force Email list –
  - ◆ Send email request to **bruce.emaus@vector-cantech.com**