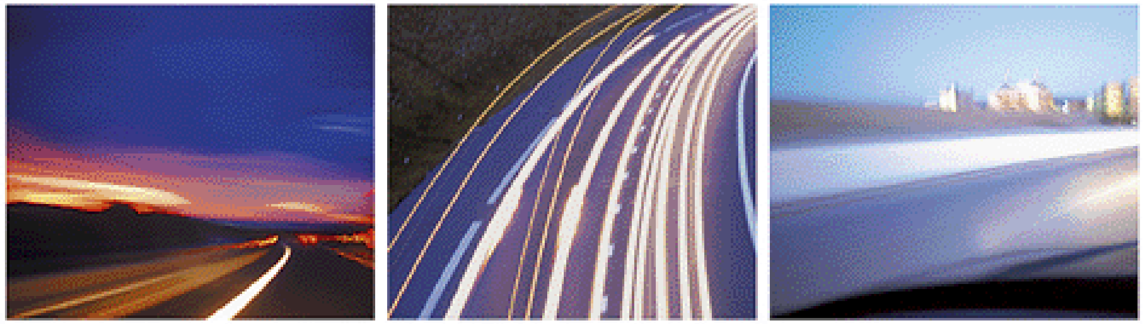
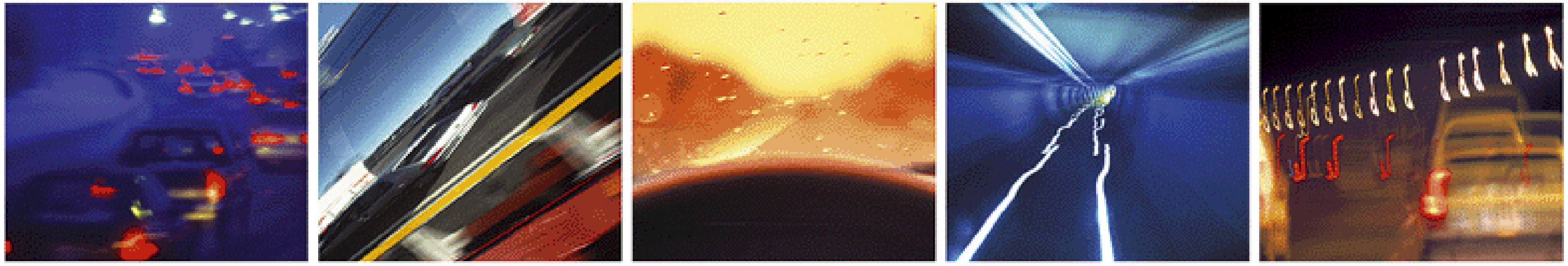
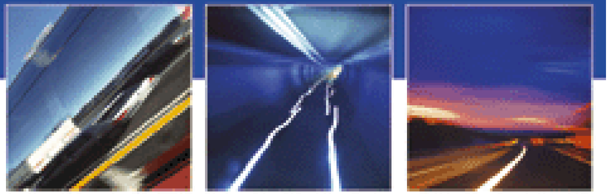


DECOMSYS



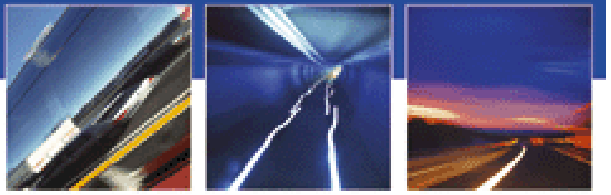
Open Source Software

Title



Experiences and considerations about open source software for standard software components in automotive environments

Overview



Experiences

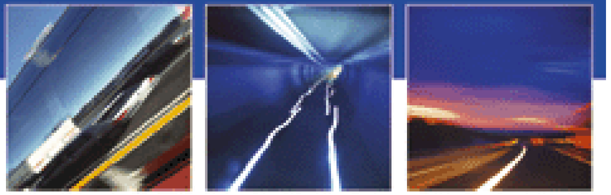
- Project
- Findings

Considerations

- X-by-wire challenges
- Relation to Open Source

Conclusion

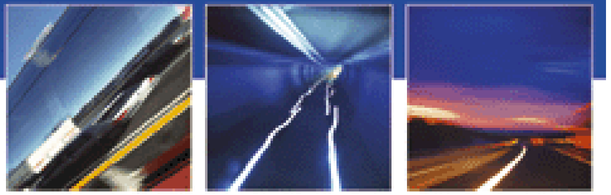
Experiences



Project

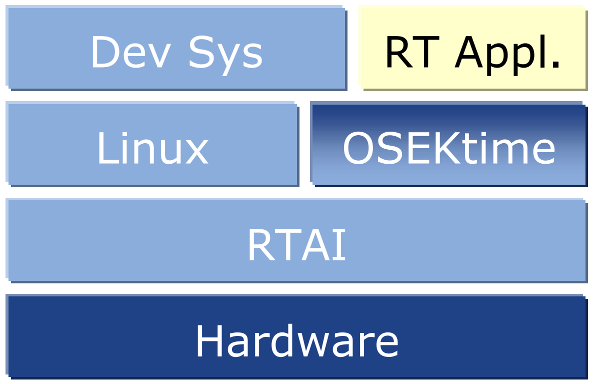
- Product including hardware and software for automotive prototyping system
- Time-triggered operating system services
- Detailed Requirements
  - OSEK/VDX OSEKtime OS 1.0 services
  - Support for multiple hardware platforms
  - Product status
  - Cost efficiency (licensing, know-how acquisition)
  - Sufficient development support for platform and developer

Experiences

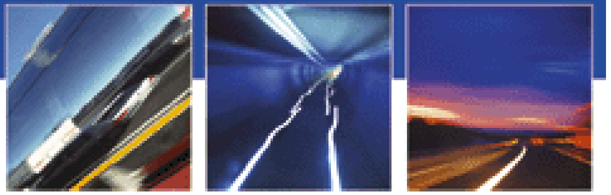


Solution

- Open source Linux Kernel
- Open source RTAI real-time extension
- Extension of RTAI in project
  - Time-driven dispatching service
  - Integration of time table interface
  - OSEKtime OS service API
  - Console support
  - Error/panic handling
  - /proc file system support (e.g. maximum measured execution time of tasks)

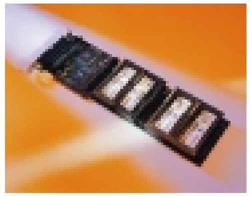


Experiences

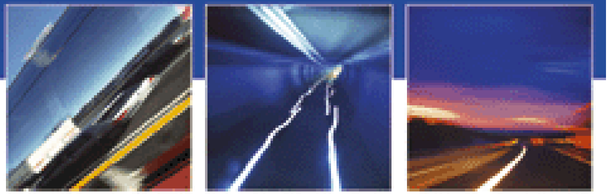


Platforms

- Industry PC
  - COTS 1,5 GHz Standard PC with PCI interface and hard disk in 19” rack
- IP860
  - COTS embedded MPC860 controller
- ARM9
  - Embedded ARM9 controller in Altera Excalibur designed by DECOMSYS



Experiences



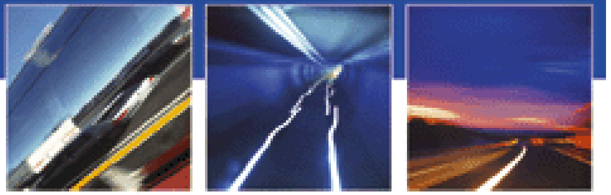
Result Preview

- All three platforms finally reached product status, but …

Detailed Observations

- Categorized by
  - Development tools
  - Runtime system
  - Support
  - Developer’s rating
  - Documentation

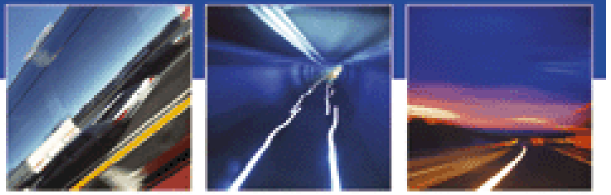
Experiences



Development Tools

- Gcc compiler and linker, make, cvs
- Found high tool quality for all platforms
  - Rating by development team on basis of test application compilation
  - One commercial compiler reached equivalent status, others were significantly worse
- No problem found in gcc code generation, linking, make and cvs for all platforms

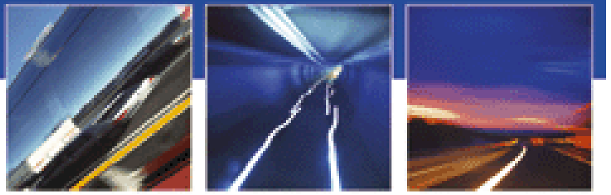
Experiences



Runtime System

- Linux and RTAI
- Problems found in each platform port of Linux and RTAI
  - All platforms: complex dependencies between versions
  - PC: interference between power management and RTAI
  - IP860: kernel service insmod faulty, problems with stack allocation in modules
  - ARM9: multiple problems

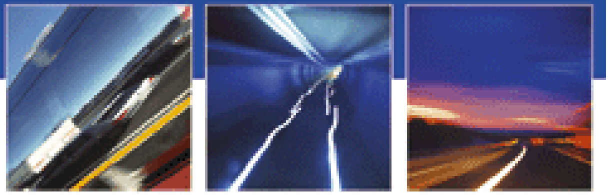
Experiences



Support

- Tools
  - Well organized and reliable
- PC platform
  - Open mailing lists, FAQs
  - Mostly competent answers, but no support guaranty
  - Took longer than planned to bring system to product status
- IP860 and ARM
  - Open mailing lists, FAQs
  - Additional porting work was necessary
  - Contracting of consulting companies for porting work and support
  - Fast and reliable support by consulting companies

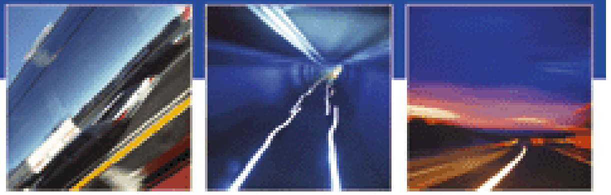
Experiences



Documentation

- Tools
  - Documentation available
  - Quality OK (user manual plus e.g. ANSI C standard)
- Runtime Systems
  - Documentation available for all platforms
  - Quality not sufficient
  - **Problem softened by availability of source code**

Experiences

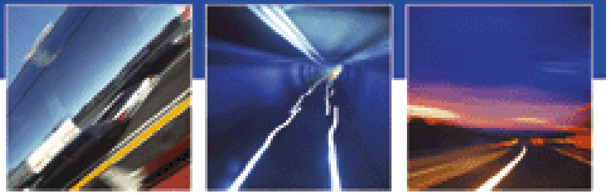


Developer's Rating

- Range [++,+,~,-,--]
- Would you use the compiler, linker, cvs in a safety related project?
  - Rating +
- Would you use Linux in a safety related project?
  - Rating ~
- Would you use RTAI in a safety related project?
  - Rating --

DECOMSYS

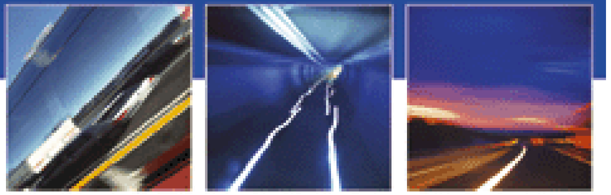
Experiences



Condensation of Experiences

- General
  - Tools have much higher standard than run time systems
  - Although support available by community, pay company for support
  - For more advanced prototyping systems we switch to commercial open source operating system
- Comments
  - Available source code helps to understand a problem (not necessarily to solve it)
  - Available source code sometimes helps to solve problems fast in time critical projects
  - gcc has a steering committee that controls development of compiler (IBM, RedHat,..)
    - This may be a key factor for high quality level

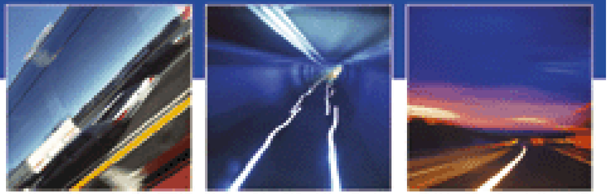
Part II



Considerations

- X-by-wire challenges
- Relations to open source
- Conclusion

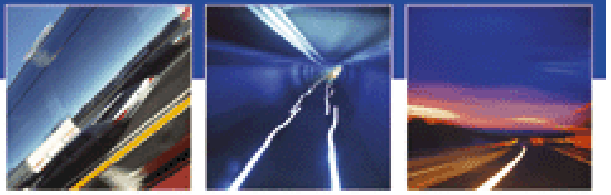
Considerations



Future Automotive by-Wire Systems

- High Performance Control Systems and high bandwidth backbones
  - Chassis control
  - Driver assistance with intelligent sensing systems
  - Architecture clean-up
- Classic X-By-Wire
  - Rear/full electronic braking
  - Steering influence
  - Full steering
  - Scope: starting from 2008

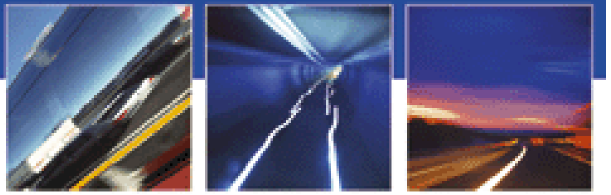
Considerations



X-By-Wire Challenges from DECOMSYS Perspective

- Integrated Design and Configuration Tools
- Standard software components
- System Reliability
- System Safety

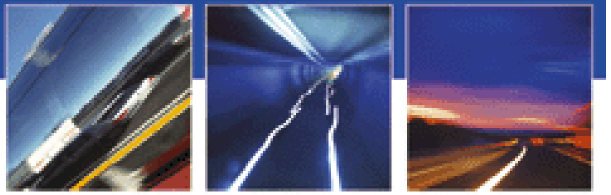
Considerations



Integrated Design and Configuration Tools

- System complexity cannot be handled without tools
  - E.g. > 1000 signals in a network
- First challenge is the seamless integration of development tools
  - No manual preparation of design data
  - Challenge is not how to do it in general, but how exactly for automotive customers
  - Tool supported collaborative design process between integrator and supplier
- Open source idea can be interesting for tools
  - At least open internal interfaces are in consideration for DECOMSYS tools
- Certification of tools is an open issue

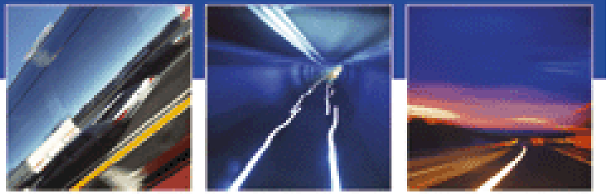
Considerations



Standard Software Components

- Operating system, communication layers, transport services, network management
- Standard software components are not an USP of a car
  - All manufacturers and supplier can use the same standard
  - Standard yet has to be found
- Benefits
  - Enable software reuse
  - Shorten development cycles
  - Create higher flexibility (e.g.: function migration)
  - Test deepness increases with every system that uses code base
- Candidate for open software solution

Considerations

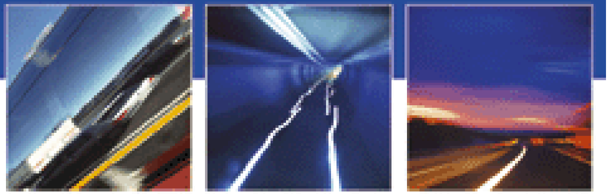


Open Standard Software Components

- Partly industry practise
    - Many components are delivered in source code with make and configuration environment
- Different opinions
    - Open software for known benefits
    - Binary components for some liability issues

DECOMSYS

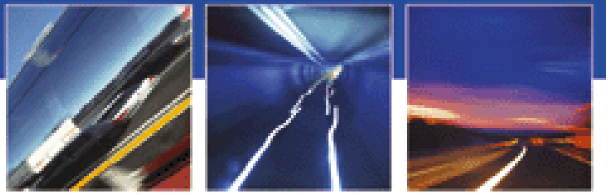
Considerations



Industry Activities for Standard Software Components

- In past numerous parallel activities
  - HIS
  - OSEK
  - ASAM
- New AUTOSAR Development Group
  - Focus on the standardization of automotive software components
  - Almost all major car manufacturer participate
  - Follows idea of open source reference implementation for components!

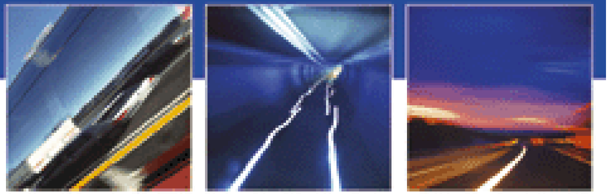
Considerations



One open Implementation of Standard Software Components

- One implementation of code base
- Advantages
  - Developers know-how focuses
  - Effort for conformance testing can be reduced
  - Generic certification (or preparation for certification) of core code base
- Disadvantage
  - Decreases market dynamics
  - Business model difficult
- Alternatively
  - One reference implementation
  - Strict conformance tests

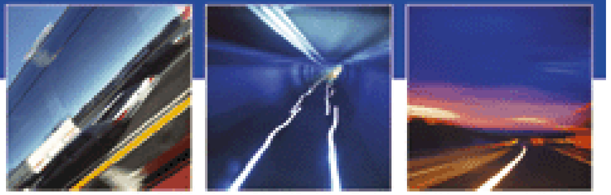
Considerations



One open Implementation of Standard Software Components

- Requires clear responsibilities and processes for
  - Software development (all elements of V-model)
  - Change management
  - Configuration management
  - Conformance testing
    - Still required since platform adaptations have to be checked

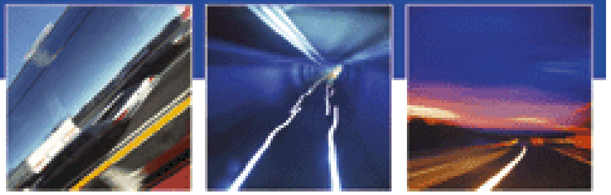
Considerations



System Reliability Aspects

- Software layers for fault tolerant communication and task execution
  - Standardization is prerequisite
  - But solutions depend strongly on fault models for underlying hardware/software which can vary (e.g.: cost factor of physical layers)
- Advantages would be
  - Test deepness
  - Increasing experience
- Disadvantages would be
  - Standard would have to cover many situations (probably high complexity)
- Standardized open source solutions may not be suitable
- Open design templates may help to reduce number of problems

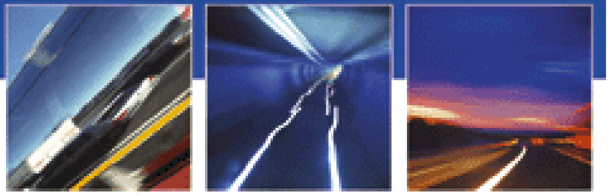
Considerations



System Safety

- The “automotive way” to build safe systems is not yet defined
  - Although it will not be very different to other industries (maybe cheaper)
- Software architectures for safety-relevant systems
  - E.g.: software solution for car state management (German: Fahrzeugzustandsmanagement)
- Solutions tend to be very system specific
- Standardized open source solutions may be unsuitable
- Open design templates may help to reduce number of problems
  - E.g., templates for distributed synchronized state machines or atomic broadcast

Considerations



Conclusion

- Potential use of open software idea for
  - Standard software components
  - Design templates for fault tolerance and safety functions
- Only under a strict regime of an organization responsible for
  - Specifications
  - Conformance specification and testing
  - Change management
  - Configuration management
  - Certification
  - Support
  - Documentation

DECOMSYS

Thank you for your attention!

DECOMSYS 2004 for 45th IFIP Workshop