Automation framework for converting legacy application to AUTOSAR System using dSPACE SystemDesk

James Joy, Anush G Nair
Tata Elxsi, India

ABSTRACT

AUTOSAR systems are getting very popular in the automotive world. The conversion of legacy application to AUTOSAR system is of great interest to automotive industry across the globe. The Python based automation framework of SystemDesk is very effective in developing time to market solutions. This paper proposes a generic automation framework for the migration of existing application to AUTOSAR architecture using a combination of Matlab scripts and SystemDesk Python APIs. The proposed automation framework uses signal information from the feature models, extracted during the conversion of feature models to SWCs, as inputs for system design. The system architecture model and the RTE are generated using SystemDesk Python interfaces. This is very effective and of having great importance for systems with large number of signals, especially when the implementation is distributed across multiple teams. This automated framework will keep the integrity and consistency of the system architecture.

INTRODUCTION

Automotive systems are becoming increasingly dependent on embedded computers. In early days, automotive systems were mainly mechanical and include very less electronic controls, but later the importance of electronics and software components started increasing. The use of embedded computers provides better information processing and control capabilities within the vehicle. The modern day cars are having multiplicity of ECUs on board. The introduction of electronic controls in vehicle started with isolated single board solutions. Later the scope changed to vehicle level solutions, using distributed control units in the network. In this distributed architecture, a feature inside the car is distributed across different ECUs connected together by different networking technologies, which makes the modern day automotive systems complex. In order to control the complexity in automotive systems, the AUTOSAR standard [1] is introduced.

The AUTOSAR standard defines architecture, application interfaces, interchange formats and an integration methodology for the automotive software. Standard application interfaces in the architecture provides the notion of the applications as a reusable component, interchange formats provides the flexibility of distributed development and the methodology supports all of these activities. The goal of AUTOSAR standard is to create an architecture which will increase the reusability of software components, improve scalability and provide seamless integration of components and services from multiple suppliers.

AUTOSAR methodology defines an outline of the system design and configuration steps to develop a system. dSPACE SystemDesk will cover the system design steps[3].

Figure 1: AUTOSAR Methodology
SystemDesk is an architecture modeling tool for the model-based development process that begins right at the functional system level. Developers working with SystemDesk can easily keep track of the planning, implementation and integration aspects that are relevant to complex system architecture and distributed software systems. SystemDesk also supports teamwork on process-oriented development, and allows OEMs and suppliers to share system models and maintain them jointly.

The overview of the migration framework is shown in Figure 2. The first step is automatic conversion of existing Matlab models to AUTOSAR application software components (SWCs). During this conversion the Matlab script will extract the signal information of each SWCs. The second step is the system architecture modeling in SystemDesk. During the system architecture modeling the SWCs are automatically modeled in SystemDesk using the Python APIs. The AUTOSAR system is modeled using the SWCs and the architecture information provided by the user and the network database files. Finally the run time environment (RTE) is generated using SystemDesk RTE generator. Since the conversion starting from the legacy Matlab models to RTE generation is automated with minimal user interaction, this automated framework will keep the integrity of the system development. The user interactions are limited to the areas of specification of the system architecture and software architecture.

### AUTOSAR SYSTEM DESIGN

The AUTOSAR system design is described by the concept of Virtual Function Bus (VFB) and RTE which really facilitates one of the key features of the AUTOSAR, relocatability of software components. VFB is the central concept of the AUTOSAR standard. The RTE abstracts the software component layer from any implementation details of the basic software and from the hardware. RTE is responsible for communication between application SWCs and service SWCs. RTE [2] also implements the OS tasks, Mode management etc.

The main steps involved in the AUTOSAR system design is the architecture definition, design of the software component, this involves the design of structure and interfaces of the functional software of your ECUs and the communication modeling between the different application and service SWCs. The AUTOSAR architecture will ensure the seamless integration of the functional software with the ECU’s basic software. The AUTOSAR system design is described in Figure 3. In the overall ECU development, the application and the RTE part is designed and generated using the SystemDesk tool.
The basic software stack (BSW) configuration is another part of the ECU development process handled by tools provided by the BSW stack vendors. The BSW stack contains all the standard automotive modules in a layered architecture. The BSW architecture contains a layer of Microcontroller Abstraction drivers, a layer of ECU Abstraction modules and a layer of Service components. AUTOSAR OS and application specific Complex drivers are additional modules provided as part of BSW stack for flexible development of automotive ECUs based on AUTOSAR architecture.

![System Design](image)

**Figure 3: System Design**

**MATLAB MODELS**

A normal legacy body control application is distributed over a number of Matlab/Simulink models that communicates among themselves as well as the lower layer software. A major part of the AUTOSAR conversion lies in converting these Simulink models to AUTOSAR compliant SWCs [1] [4] [5]. Since the number of Simulink models as well as the interfaces can be considerably large in number, an automated approach towards SWC creation is necessary from the point of view of development effort as well as human error reduction.

Matlab scripting (m-scripting) is a simple and effective way to quickly achieve automation in the various stages of AUTOSAR conversion. The different stages in the AUTOSAR conversion are legacy model analysis and generation of I/O signal list, creating AUTOSAR interface, port and data elements corresponding to the I/O signal list of the legacy model, creating wrapper model, runnable and SWC and finally auto code generation.

The different stages of AUTOSAR conversion through m-script based automation is represented in Figure 4. The inputs to this process are the legacy Simulink model(s) and the list of AUTOSAR standard key words used to form signal names. The outputs of this process are the SWC and generated code for the SWC corresponding to the legacy model.
An SWC consists of one or more runnables, which can be triggered independently. The individual runnables contain the functionality which exists in the form of legacy Simulink models. Since the legacy models have non AUTOSAR I/Os, conversion of these signals to AUTOSAR standard signals are mandatory. Input and output wrapper Simulink models do the conversion of signals. This conversion can be a simple name change to AUTOSAR standard name, signal multiplexing, de-multiplexing or logical conversions.

![Diagram of Core Function SWC](image)

**Figure 4: Modeling Approach**

**SYSTEM MODEL AND AUTOMATION FRAMEWORK**

System architecture modeling is done using the SystemDesk. SystemDesk is having Python APIs which will enable automation of the architecture modeling and RTE generation process. The software component details for the architecture modeling is directly extracted from the Matlab models and the user will add system architecture details and the ECU details in the same excel sheet. The ExcelDataProcessing class in the python script will read all the necessary information and convert it into a defined standard format that information will be used by other classes to model the architecture and configure the ECU details for the RTE generation. The class diagram for the Python script is given in figure 5.

SwComponentModeling class convert the information read from excel sheet to a software component in the system desk library of software components. The rest of the python classes uses this software components for the system architecture modeling. SystemArchitecture class reads the information from the excel sheet and the placements of software components to ECU is handled here. Once the software components are mapped to an ECU then the SwArchitecture class configures the operating system tasks based on the architecture details. The Networking class reads the CAN Database (DBC) files and LIN Description Files (LDF) files to configure the network and map the network signals to the software component signals. The Services class read the information from the excel sheet and create the service components in the system desk library and configure the service component interactions. All these processes will be
monitored by the ExceptionReports class and check the integrity of all the data and the exceptions are reported. This class will provide a statistical report at the end of successful completion of RTE generation.

CONCLUSION

This paper presents an automation framework for the conversion of existing applications to AUTOSAR architecture. The automation framework consists of automation in Matlab for the conversion of application models to SWCs and the automation in SystemDesk for the system modeling and RTE generation. The same framework can be used for the AUTOSAR system modeling and RTE generation for new applications as well. In a system having large number of signals and lot of interactions, it is really difficult to model manually without human errors; the automation can handle large number of signals without any errors. The automation includes the data consistency checks and will ensure the integrity of the system. This automation framework was successfully used in one of our body system ECU migration project having large number of applications features implemented as Matlab models.

REFERENCES

[4] AUTOSAR_SimulinkStyleguide.pdf Ver 1.0.5