EARS-CTRL: The Demonstration

Levi Lúcio[1], Salman Rahman[1], Saad Bin Abid[1], and Alistair Mavin[2]

[1] fortiss GmbH
Guerickestraße 25, 80805 München
{lucio,abid}@fortiss.org, salman.rahman@tum.de
[2] Rolls-Royce, PO Box 31, Derby, UK
alistair.mavin@rolls-royce.com

Abstract. In this part of the paper, we demonstrate our tool in practice with the example case study of a sliding door controller. We build controller specification in natural language, automatically generate the controller and then perform verification of the generated controller by utilizing testing and simulation techniques. The demo video can be found at: [URL: youtubevideo]

1 Introduction

In this demonstration, we focus on providing a guidence to the users of our tool. Our EARS-CTRL tool is a github project [1]. An overview of the demonstration steps is as follows,

- Progressively build a set of requirements for the controller written in english like sentences using EARS lanaguage [2]
- Automatic realisation of the controller by analyzing if the specification is complete (our one click approach)
- Simulation and test cases for performing conformance analysis between the EARS-based specification and the generated controller.

2 The Running Example: Automatic Sliding Door

Our running example for this demo is a sliding door system of PLC based controller. The behavior of the sliding door controller is supposed to be as follows: the sliding door opens if somebody enters by sensing the object using the infrared sensor. The sliding door opens until the opening limits are reached (also detected by the sensor). Upon reaching the opening limits the count down timer starts. and the controller then closes the door when the timer expires until the closing limit is reached detected by sensor.

3 Tool Demonstration

Discuss the main steps of process that needs to be followed for synthezing and performing the conformance analysis,that are,

3.1 Glossary building and terms definition

The first step towards writing the controller specifications as a natural language in EARS-CTRL is to define the glossary terms. The user is provided with a projectional editor (figure 1) to define glossary terms for the following, 1) components that interface with the controller, 2) sensors and actuators those components make available to the controller and 3) rules between signals definitions.

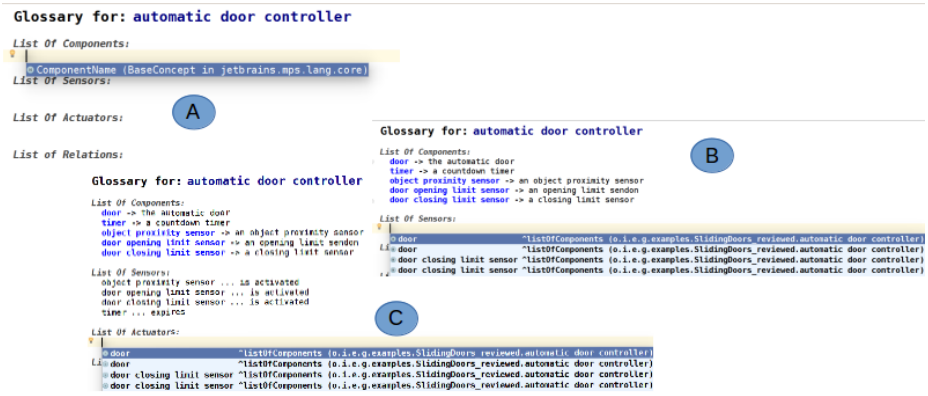

Fig. 1. step-by-step glossary building for sliding door controller: (A) Components definition, (B) Sensors definition and (C) Actuators definition

3.2 EARs-based requirements building for the controller

```
Requirements for: automatic door controller
Glossary:        automatic door controller
Temporary path:  solution_root/models

Req1 : When object proximity sensor is activated occurs , the automatic door controller shall open door .
Req2 : When door opening limit sensor is activated occurs , the automatic door controller shall
    stop door and start countdown timer .
Req3 : When countdown timer expires occurs , the automatic door controller shall close door .
Req4 : When door closing limit sensor is activated occurs , the automatic door controller shall stop door .
```

Fig. 2. EARS requirements for sliding door

3.3 Synthesizing the EARs-based requirements

3.4 Simulation and Test Cases

interfacing with Simulink

setting up test case parameters

obtaining results and conformance analysis

4 Discussion

Discussion will go here...

Notes: Point can be arised that if you are generating code from the model why do we need tests? we can sell our idea by stating that we don't necessarly want to generate the code but write the controller requirements as EARS and somebody else can write the code. Some engineers don't rely on automatic code generation and want to develop/build controllers explicitly. In such a situation, test case generation would help to perform conformance between the specified controller and its respective implementation. More points: 4. Code generation and synthesizer for EARS-based requirements 5. Test case generation for conformance checking of the generated controller 6. Interfacing with Matlab Simulink 5. Viewing the Results 7. Lessons learned

8. Start discussing the steps as flow model and follow exactly the same steps!!!! 8.1 Expression of Controller Requirements as EARS and related Models (e.g., Glossary) 8.2 MPS constraints to ensure completeness of the EARS-based requirements (hint: discuss some examples if something breaks) 8.3 Test case generation process (step-by-step) that includes interfacing with simulink, test case generation sequences, showing the results of the results as inputs and output as the SimulinkResult model

Notes: Point can be arised that if you are generating code from the model why do we need tests? we can sell our idea by stating that we don't necessarly want to generate the code but write the controller requirements in EARS and somebody else can write the code. Some engineers don't rely on automatic code generation and want to develop/build controllers explicitly. In such a situation, test case generation would help to perform conformance between the specified controller and its respective implementation.

Acknowledgements

This work presented in this paper was developed in the context of the "IETS3" research project, funded by the German Federal Ministry of Education and Research under code 01IS15037A/B.

References

1. Ears-ctrl github project. `https://github.com/levilucio/EARS-CTRL.git/`.
2. A. Mavin, P. Wilkinson, A. Harwood, and M. Novak. Easy approach to requirements syntax (ears). In *2009 17th IEEE International Requirements Engineering Conference*, pages 317–322, Aug 2009.