State based Models

Ankit Goel National University of Singapore ankit@comp.nus.edu.sg Bikram Sengupta IBM Research, India bsengupt@in.ibm.com Abhik Roychoudhury National University of Singapore *abhik@comp.nus.edu.sg*

31st International Conference on Software Engineering - 2009

INTRODUCTION

In model-driven software development of distributed systems-

Scenario based models, such as High-level Message Sequence Charts (HMSC), highlight inter-process communication and are closer to system requirements

➢ State based models, such as Finite State Machines (FSMs), highlight intra-process behaviour and are suitable for code generation.

Obtaining a state based requirements model involves relatively more manual effort, as compared to the scenario based model and is hence, error prone.

OVERVIEW



We exploit the distinct strengths of the two modelling styles within a round-trip engineering validation methodology-

- 1. Test cases containing traceability information, linking test case events to requirements, are derived from a scenario based requirements model (HMSC).
- 2. Implementation code (C++) is generated automatically from a state based requirements model (Statecharts) via Rhapsody tool.
- 3. The implementation code derived in (2) is tested against the test cases obtained from (1). Execution sequences from unsuccessful test cases are traced back to the original requirements using traceability information to aid debugging.

Footprinter: TOOL ARCHITECTURE

- Comprises of four main components:
 Eclipse based graphical editor to input- (i) scenario based requirements model as an HMSC, and (ii) a testpurpose as an MSC.
- Test generation engine implemented in XSB logic programming system. Generates test cases in the form of MSCs from a HMSC model, guided by a user provided test-purpose (MSC).
 Test stub generator. Generates tester code (C++) from a test case MSC for testing an implementation.
 Test case / Test execution trace visualizer as MSC using Mscgen and Eclipse. Traceability information is also displayed.

E Topcased Modeling - Footprinter/Models/HMSC.umldi - Eclipse SDK						
File Edit Navigat	e Search Project Run Window	Help				
📑 🕶 📄 📄	Q₂ • 🔗 ½ • ₽ • ♥ ↔	╸╸。╸				
⊠ [»] 1 □ □	🐏 *HMSC.umldi 🛛 👷 *MSC1.u	umldi				
□ 🔄 🏹	/Footprinter/Models/HMSC.umldi					
Footprinter	↓ Select	state machine UMSC State Machine				
🗁 Models	门 Marquee 🗸 👻					
💫 HMSC.ı	🔁 Objects 🔹 🖈					
^ຍ ານ HMSC.ເ	🔁 Region					
💑 MSC1.u	State					
ີ MSC1.u	- Composite State					

SCREENSHOTS

🚝 Topcased Modeling - Footp	rinter/Models/MSC1.umldi - Eclipse SDK
<u>File Edit Navigate Sea</u> rch	<u>P</u> roject <u>R</u> un <u>W</u> indow <u>H</u> elp
🔁 🕶 🔛 🗁 🛛 🗛 👻 🗸	〃 월 ▼ 월 ▼ 🌣 🌣 ▼ 🔷 ▼ 🛛 🗳 🌾] 點 品 답 點 唱 恕 🛊 🛒 🚍 때 .
🔁 Nav 🧭 To 🛛 🖓 🗖	ມາເຊັ່ HMSC.umldi 🦉 Test.umldi 🖓 *MSC1.umldi 🔀
□ 🔄 🏹	/Footprinter/Models/MSC1.umldi
Footprinter	- 🔓 Select
·	Marquee
	🕞 Objects 🔹
	T Lifeline
	Discretion
	A: B:

Topcased Modeling - Footprinter/Models/TestPurpose1.umldi - Eclipse SDK				
<u>F</u> ile <u>E</u> dit <u>N</u> avigate Se <u>a</u> rch <u>P</u> roject <u>R</u> un <u>W</u> indow <u>H</u> elp				
<mark>[] ▼ [] ≙ & ▼</mark>				
🕱 🎽 🗖 🗖 🖓 🖓 🖄 🕺 🖄 🕺 🕺 🕺 🕺 🕺 🕺 🕺 🕺 🕺 🕺 🕹				
🕞 🖕 🗸 /Footprinter/Models/TestPurpose1.umldi				
Footprinter	Select			
🗁 Models	Marquee • interaction Tes	stPurpose1		
💫 HMSC.ı	🔁 Objects 🔹 🖈	/		
ໃນ HMSC.ເ	T Lifeline			
💫 MSC1.u	Discretion			
MSC1.u	🔁 Messages 🔹 🔺	<u>B:</u>		
👸 TestPur	Scall (synch.)	📥		
UMIZ TESTPUR	← Reply Message	Ready		
	Call (asynch.)			



ILLUSTRATION



REFERENCES

1. Footprinter: Roundtrip Engineering via Scenario and State based Models. Ankit Goel, Bikram Sengupta and Abhik Roychoudhury, ACM International Conference on Software Engineering (ICSE) 2009, Short paper.