

mVerifyTM

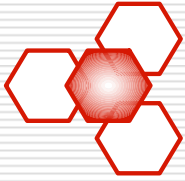
A Million Users in a Box ®

**The Advanced Mobile Application
Testing Environment: Project Report**

**Robert V. Binder
James E. Hanlon**

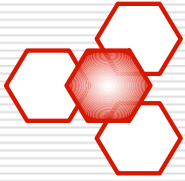
A-MOST Workshop
ICSE 25, St. Louis May 16, 2005

www.mVerify.com



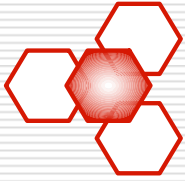
Overview

- Background and Motivation
- Models
- Demo
- Lessons Learned



Mobile App Challenges

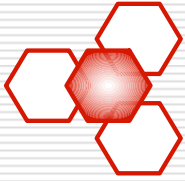
- Rubik's Cube complexity, many new failure modes
 - Connectivity
 - Mobility
 - Scalability
 - Security
 - PLUS assure functionality, performance, integration
- Adequate testing ?
- End to end, realistic testing only hope for high reliability



AMATE Project Background

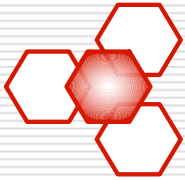
- Advanced Mobile Application Test Environment (AMATE)
- Goal: Achieve realistic end-to-end mobile testing
- Approach: *Model-based, Mobile-Centric*
 - Signal variation related to mobility
 - User behavior related to mobility
 - Traffic related to mobility
- NIST/ATP funded R&D





AMATE Technology Highlights

- Tester's Assistant generates initial model
 - Automatic inference (guess) of application semantics
- Simulator generates tests from model repository
 - Scheduled location-specific behavior
 - Scheduled location-specific airlink conditions
- Digital AirLink Emulator varies signal
 - Non-intrusive, software-defined radio
 - Consumes simulator-generated commands
 - Achieves controlled real-time airlink variation
- Distributed control, observation, evaluation
 - Scalable (1:1000 fan out, 2 levels, mobile device)
 - Tcl test object framework, Tk GUI, C++ drivers & controllers
 - Relational database
 - Web Services: XML, WSDL, SOAP, BEEP, HTTP



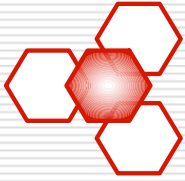
Behavior Model

- ❑ Extended Use Case
- ❑ Tester's Assistant generated and user editable

			1	2	3	4	5
Conditions							
Test Input Conditions for automatic test input generation	Variable/Object	Value/State					
	Widget 1	Query	T	T			
	Widget 2	Set Time			T		
	Widget 3	DEL					
	Host Name Pick	Valid	T	F	T		
	Host Name Enter	Host Name					
Actions							
Variable/Interface			Value/Result				
Required Actions for automatic result checking	Host Name Display	No Change	T	T	T	T	
		Deleted					T
		Added					
	Host Time Display	No Change					
		Host Time	T		T		
	CE Time Display	Last Local Time	T	T		T	
	Host Time			T			
	Error Message		F	T	F	T	F
Relative Frequency			0.35	0.20	0.30	0.10	0.05

Logic combinations
control test input data selection

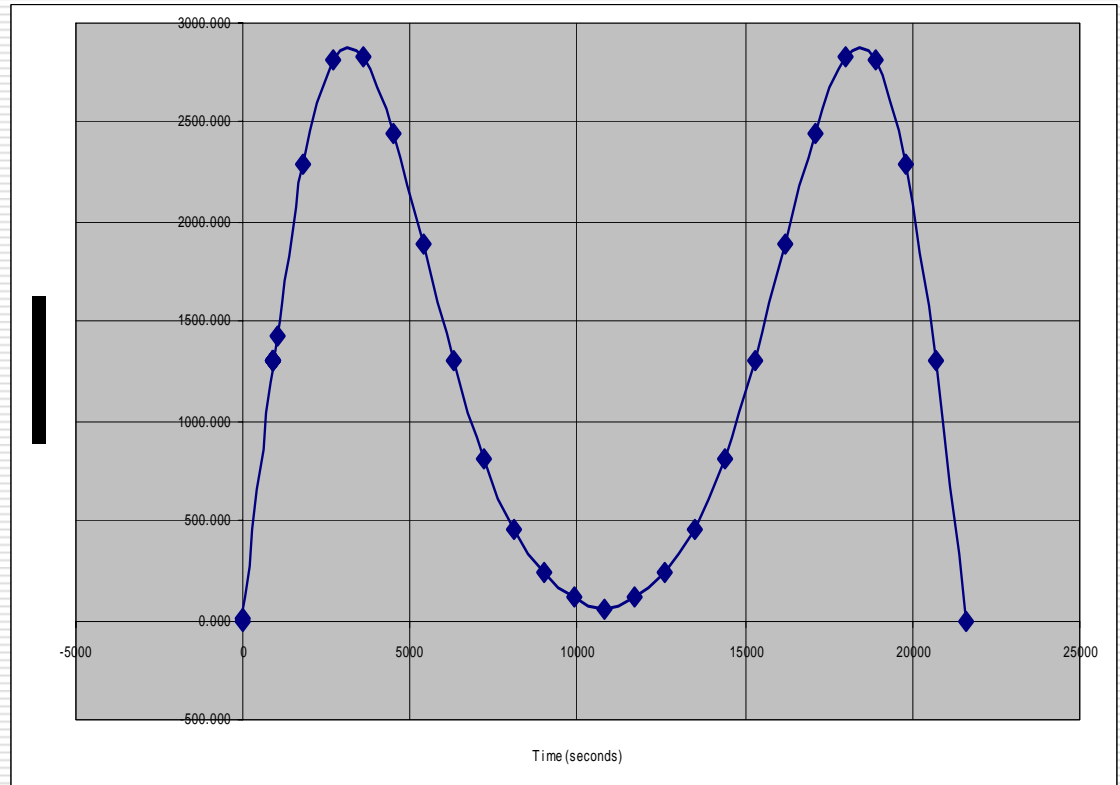
Usage Profile
controls statistical distribution of test cases



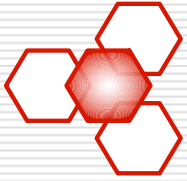
Load Model

□ Vary aggregate input rate

- Arc
- Flat
- Internet fractal
- Negative ramp
- Positive ramp
- Random
- Spikes
- Square wave
- Waves



Actual "Waves" Loading



Mobility Model

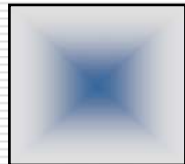
- Map generates real time itinerary for N virtual users
 - Location-specific signal strength
 - Location-specific end-user behavior
 - Controls Airlink Emulator



Virtual Users



1 Bar Signal



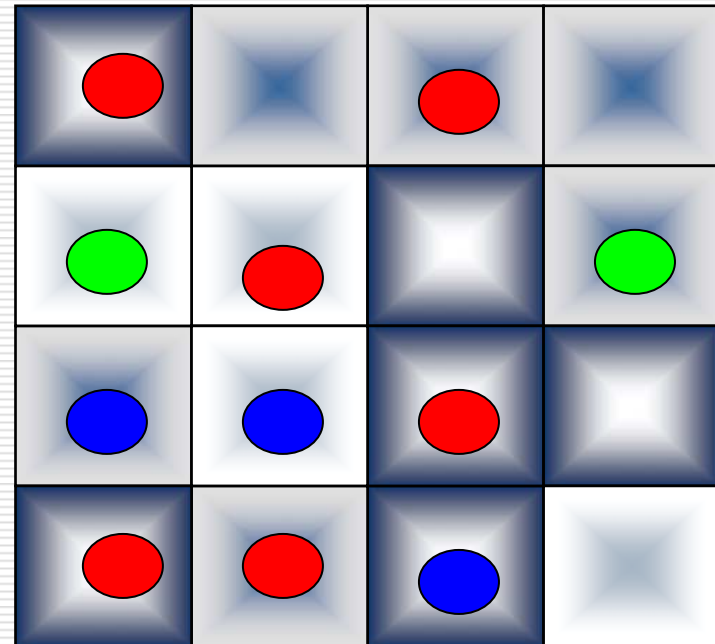
2 Bar Signal

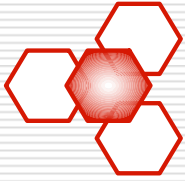


3 Bar Signal

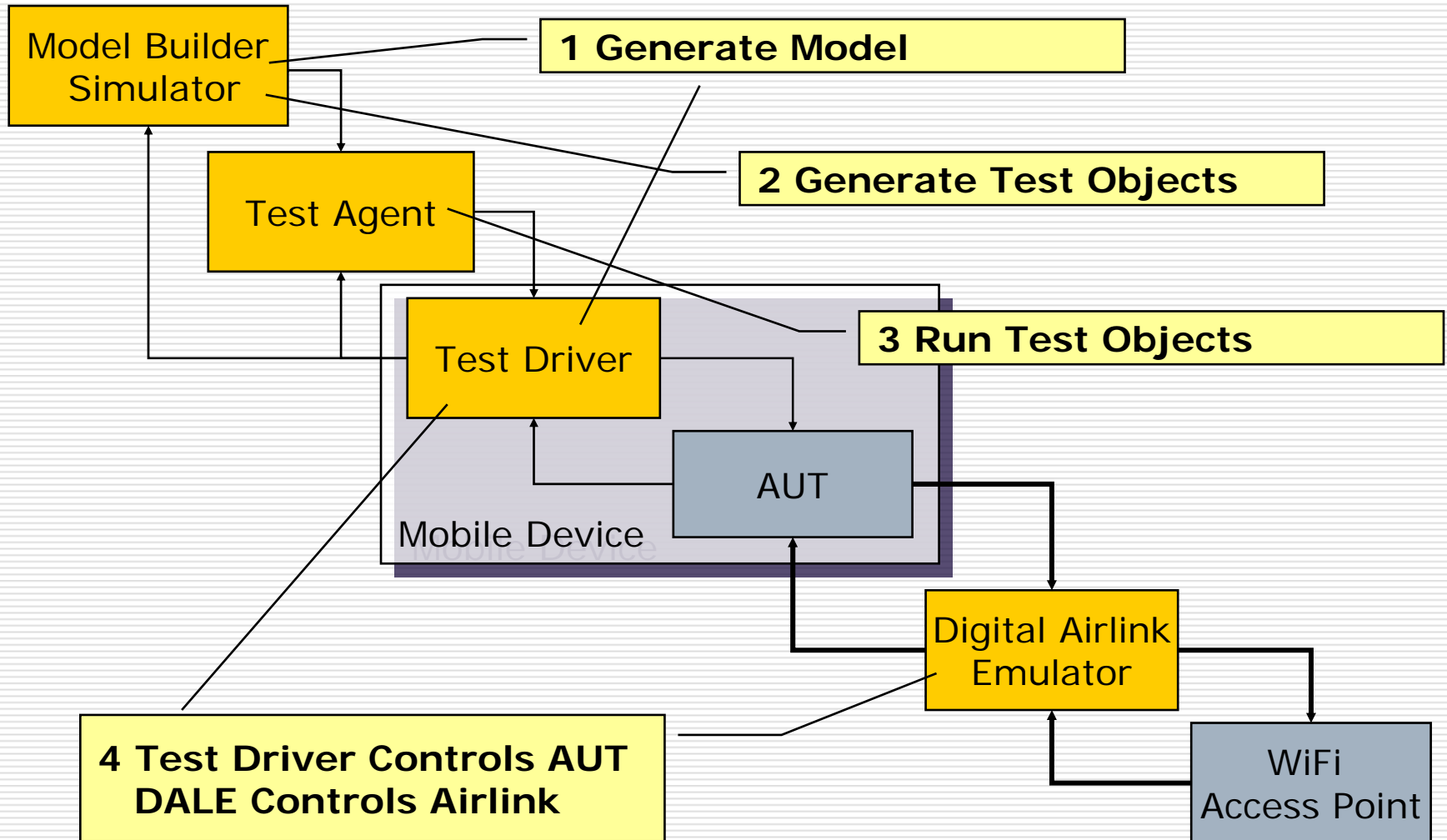


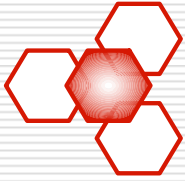
Signal Propagation Map





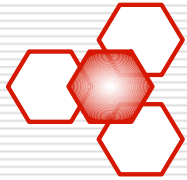
An AMATE Session





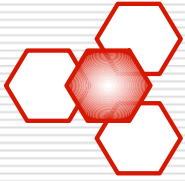
Lessons: Test Automation

- Goal: try to use/adapt standards, open source
- Eclipse/Hyades interesting, but too Java-centric
- FIPA agent model
 - Some concepts useful, but didn't need movement
- Postgres + Win32 NRFPT
 - MySQL stable, fully featured
 - Relational-OO integration challenges
- Extreme Programming x-Unit framework not scalable
- Plumbing critical and expensive
 - CORBA too heavyweight, not web-friendly
 - Own message system abandoned – limited, buggy, expensive
 - BEEP + SOAP + transport (TCP, USB, named pipes ...)



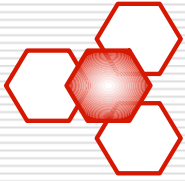
Lessons: Model-Based Testing

- Deterministic end to end model not feasible
 - Exogenous simulation strategy effective
- RF Propagation
 - Modified ray-casting, Quad Trees, DXF import
 - UMTS Interference Spec
- XML limited as meta-data representation
- TTCN complete & well-structured, but standalone
- UML Test Profile still a muddle
- Usability critical, hard to get right
 - Incremental composition, minimize dependencies
 - “Tester’s Assistant”



Lessons: Mobile Technology

- Software Defined Radio price/performance 100x
- Wavelength matters
- Proprietary islands; HW/SW Adapter framework
 - *Win Mobile, J2ME, Symbian, BREW, ...*
 - *WiFi, CDMA, GSM, WCDMA, WiMax, Ultra WB, Mesh, RFID, ...*
- Fundamental limitations of lab testing
 - Scalability bandwidth limited
 - Virtualization doesn't scale over 1000s
 - *Share fielded configuration*



Thank You, Open Source

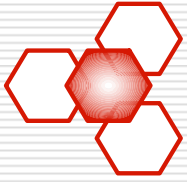
Embedded

- Tcl 8.4
- Image Magic
- MySQL
- Linux (Red Hat)
- Universal Software Radio Peripheral
- gnu Radio (DSP interface)

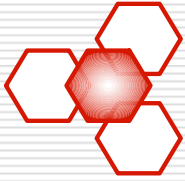
Free, but not cheap ...

Tools

- gSOAP
- SWIG
- Subversion
- Boost/Jam
- RoboDoc
- Tcl Wiki
- Bugzilla
- cppUnit
- gEDA
- PCB



Q & A



Lessons: Process

- Systems engineering + iterative development
 - Six development increments
 - Architecture matters, patterns useful
 - Plan throw *several* away, you will anyhow
 - Result: family of OO Frameworks
- Full time sys/tool admin necessary
- XP-style testing, but not “Test First”
- Customer dialog
 - Real problems, avoid duplication
- Build shared vision through communication