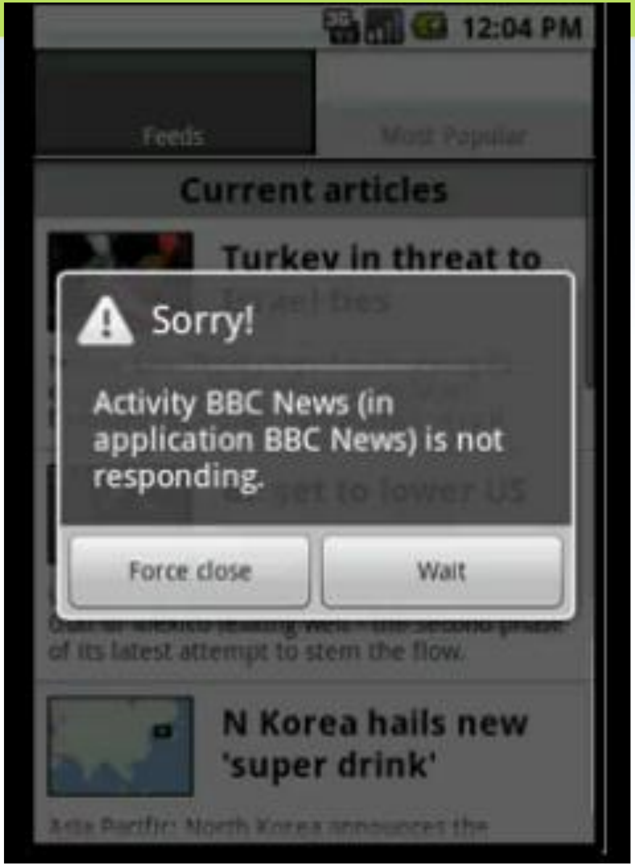
Experiences from Model-Based GUI testing of Smartphone Applications

Mika Katara
Department of Software Systems
Tampere University of Technology, Finland
first.lastname@tut.fi

Special thanks to the former TEMA team at Tampere University of Technology: Henri Heiskanen, Antti Jääskeläinen, Mika Maunumaa, Mika Mäenpää, Antti Nieminen, Tuomas Pajunen, Tommi Takala, and Heikki Virtanen

What are We Looking For?

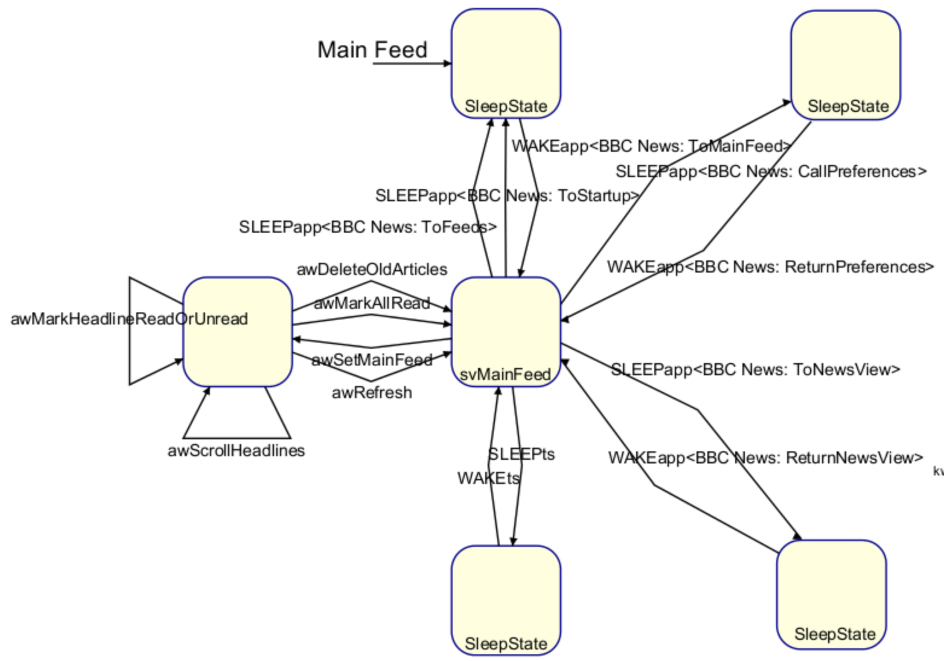


Activity BBC News (in application BBC News) is not responding.

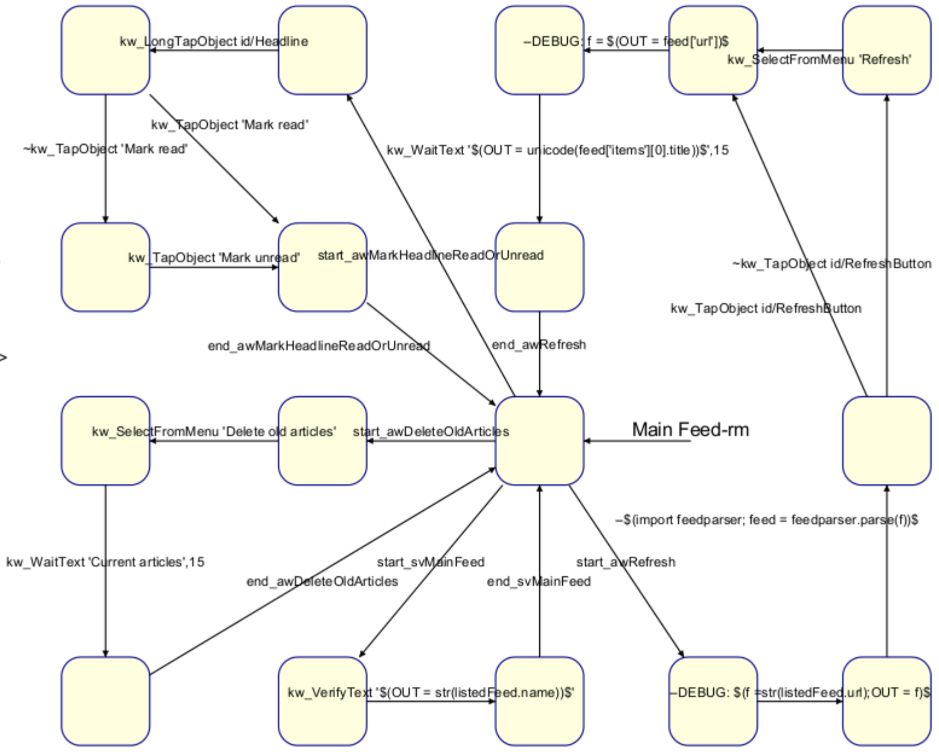
Bugs that affect smartphone users, i.e. almost everybody

How?

On-line model based testing using models describing what the user can do with the GUI and how the apps interact


Main Feed
SleepState
SleepState
WAKEapp<BBC News: ToMainFeed>
SLEEPapp<BBC News: CallPreferences>
SLEEPapp<BBC News: ToStartup>
SLEEPapp<BBC News: ToFeeds>
awDeleteOldArticles
WAKEapp<BBC News: ReturnPreferences>
awMarkAllRead
awMarkHeadlineReadOrUnread
awSetMainFeed
svMainFeed
awRefresh
SLEEPapp<BBC News: ToNewsView>
awScrollHeadlines
SLEEPts
WAKEts
WAKEapp<BBC News: ReturnNewsView>
SleepState
SleepState

(a) Action machine: main headline view.


kw_LongTapObject id/Headline
–DEBUG: f = $(OUT = feed['url'])$
kw_SelectFromMenu 'Refresh'
kw_TapObject 'Mark read'
~kw_TapObject 'Mark read'
kw_WaitText '$(OUT = unicode(feed['items'][0].title))$',15
kw_TapObject 'Mark unread'
start_awMarkHeadlineReadOrUnread
~kw_TapObject id/RefreshButton
kw_TapObject id/RefreshButton
end_awMarkHeadlineReadOrUnread
end_awRefresh
kw_SelectFromMenu 'Delete old articles'
start_awDeleteOldArticles
Main Feed-rm
–$(import feedparser; feed = feedparser.parse(f))$
kw_WaitText 'Current articles',15
start_svMainFeed
end_awDeleteOldArticles
start_awRefresh
end_svMainFeed
kw_VerifyText '$(OUT = str(listedFeed.name))$'
–DEBUG: $(f =str(listedFeed.url); OUT = f)$

(b) Refinement machine: main headline view

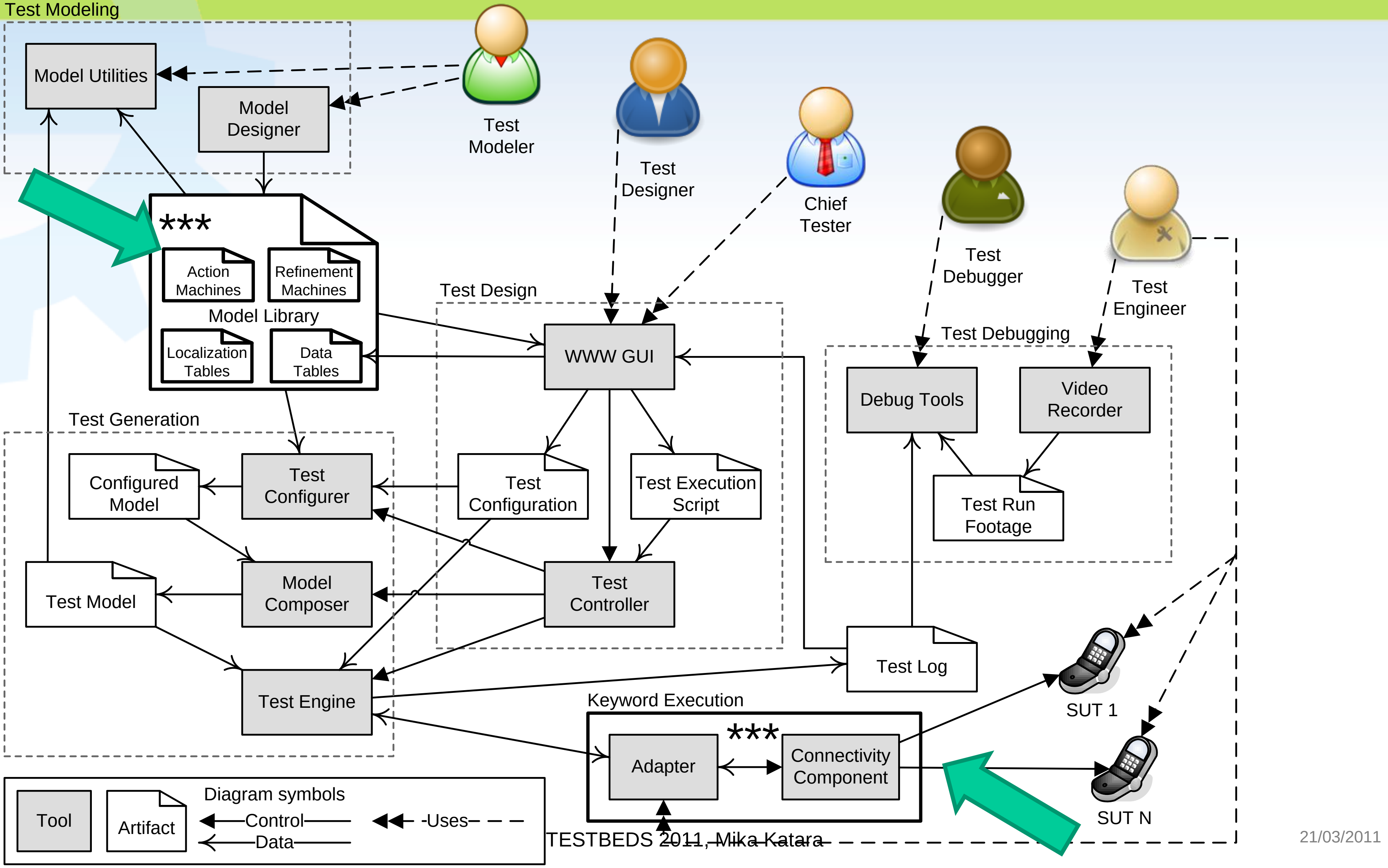
Obstacles and Opportunities for MBT

- Practitioners are willing to try out new tools that might help them
  - Wide variety of open-source testing tools already used (agile unit testing, continuous integration, etc.)
- Practitioners are not willing to invest heavily on modeling or specification in general
- When quality is not a prime consideration, conventional testing methods seem to work reasonably well
- There are areas that are very hard to test using conventional methods (static and linear test cases)
  - Many applications running concurrently and sharing resources may suggest concurrency problems
- Protecting the brand: End users who experience application hang-up/crashing problems etc. may post their bad experiences to the Internet

TEMA Toolset – Hiding Innate MBT Complexity

- Since testers don’t want to directly deal with models or test generation algorithms, we have abstracted the algorithms out in our web GUI
- TEMA web GUI is testers’ interface with the test server, used for designing and managing test configurations, running and tracking actual tests, and managing test model packages
- This all boils down to allowing testers to just choose what they want to test and what physical device they want to run their tests on
- Organizational impact:
  - Need for test design has diminished, only test configurations (that may involve use cases) have to be created
  - Modeling is imperative
    - High-level models can be reused, but domain-specific refinements must be created case by case for each domain

TEMA Tool Architecture



Test Suite Maintenance

- A major problem with conventional test automation, especially in the GUI context, is the maintenance of the test suites
  - In the worst case, you have to modify each test in your suite whenever something changes in the SUT (System Under Test)
  - Using models, test suites are generated automatically, and you only have to change your model
    - Or few of the component models

Keywords and Action Words

- Action words describe the user’s actions at a high level of abstraction
  - Send an SMS, answer a call, add a new contact etc.
  - Used in high-level models (action machines)
- An action word is translated to a sequence of keywords (keystrokes) for menu navigation, text inputting etc.
  - Some action words can have multiple keyword sequences implementing them
  - Keywords are used in low-level models (refinement machines)

To achieve a good separation of concerns, we use action words and keywords in separate models at different levels of abstraction

Action machines containing action words are composed with refinement machines containing key words

The resulting composite model is input to the tools executing the model i.e., generating the test cases

To avoid state space explosion, this has been implemented using an on-the-fly algorithm

Example Test Models

S60 Camera application, action machine

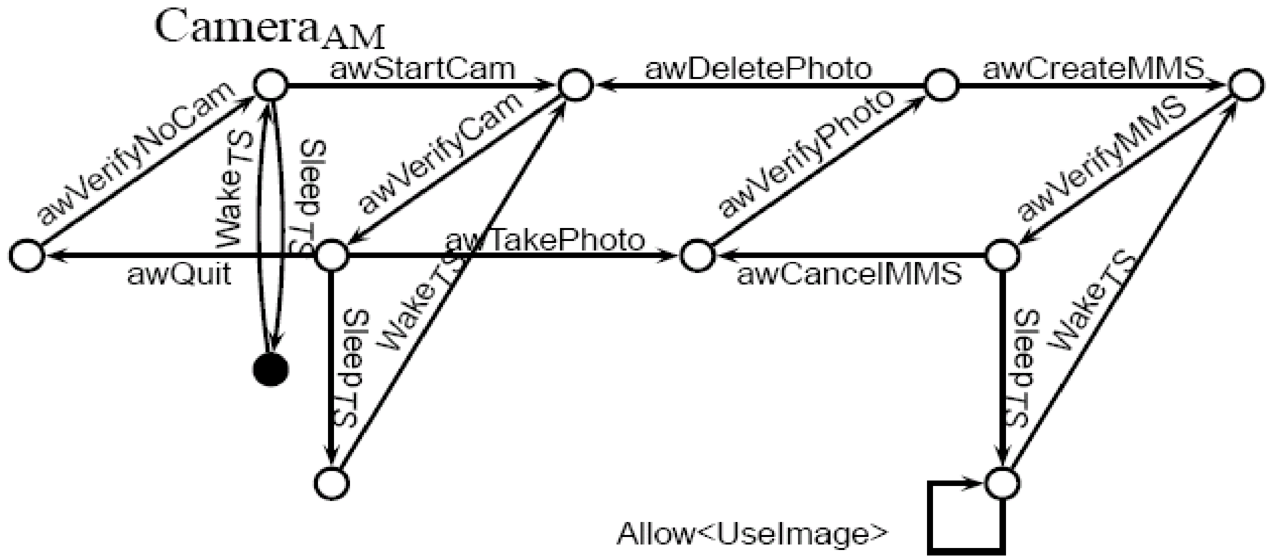


Illustration: Antti Kervinen/TUT

S60 Camera application, refinement machine

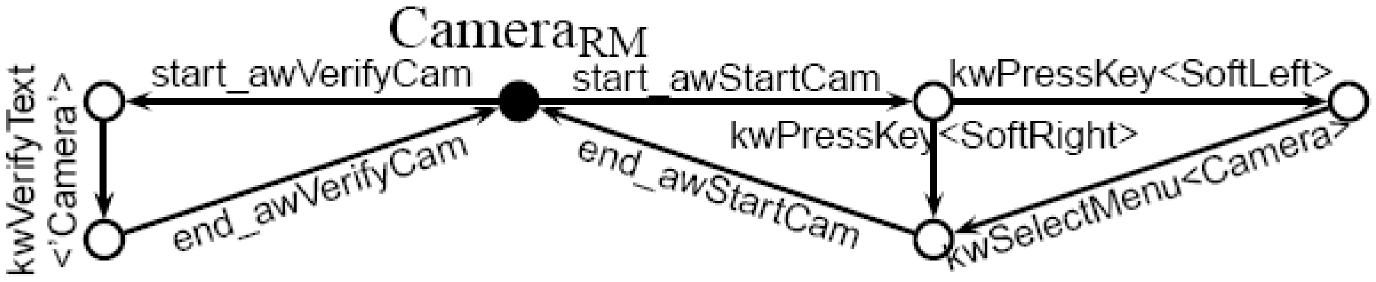


Illustration: Antti Kervinen/TUT

Debugging Long Error Traces

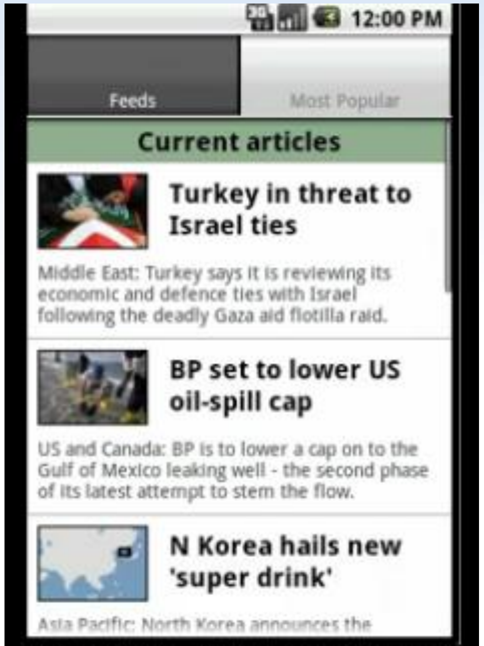
Debugging is a major practical problem

This is emphasized in online MBT, since error traces can be very long

Solutions:

1) Test Run Video Synchronization with Log Data

2) Trace Incrementation
Based on the concept of gradually executing a failed test run in subsets


12:00 PM
Feeds
Most Popular
Current articles
Turkey in threat to Israel ties
Middle East: Turkey says it is reviewing its economic and defence ties with Israel following the deadly Gaza aid flotilla raid.
BP set to lower US oil-spill cap
US and Canada: BP is to lower a cap on to the Gulf of Mexico leaking well - the second phase of its latest attempt to stem the flow.
N Korea hails new 'super drink'
Asia Pacific: North Korea announces the

```
Emulator/BBC%20News%20-%20Startup: start_awStartNews
0604150017.028: kw_LaunchApp 'net.jimblackler.newswidget.MainTabActivity'
0604150017.031: kw_IsTrue False...?
0604150017.048: kw_UpdateScreen ...?
```

Domains Conquered

- We have been primarily focusing on mobile GUI testing, but our approach is also suited to other domains
- Our model library presently holds models for the following domains:
  - S60
    - Over 100 action machines, almost 1000 action words
    - Estimated number of states if we would compose all the models in parallel: at least 10^19 states
  - Mobile Linux
  - Android
  - Java Swing
- Action machines (high-level models) have been reused for different domains

Conclusions

- Modeling typically uncovers more bugs and quirks than test execution itself
  - Reverse-engineering vs. use of system models
  - Lack of precise enough GUI specifications
    - Agile trend hasn't made matters easier….
- It is possible to find bugs in already well-tested applications
  - Mostly minor or cosmetic, but also serious (system errors, etc.)
- A talented student was able to create the first version of the S60 model library in 2 months (+1 month for debugging & maintenance)
- Automatic GUI testing requires mature test automation support from the domain

Case: S60 (Project Starting Point)

- Built-in applications in S60 smartphones, such as Gallery, Music Player, Flash Player, Notes, Voice Recorder, Contacts and Messaging
- Keyword execution using proprietary and commercial test automation tools
  - Optical character recognition was used for verifications, which caused some reliability and maintenance issues
- 21 defects of different severities and priorities were found
  - Some of these defects existed in more than one smartphone model
  - The most severe of the defects caused the phone to hang with “System error” message on the display
  - About two thirds of the defects were discovered while modeling (reverse engineering), and the remaining third by execution (dynamic testing)
  - Most of the defects had already been previously found in traditional testing (both manual and automatic test execution), but they had not been fixed for some reason
    - However, there were also some that were totally new
  - Many of the defects were related to **concurrency issues**: performing some multimedia-related functionality in one application and then switching to another application caused unexpected behavior in some circumstances
  - In addition to defects found in applications, some were found in test automation tools, which was considered rather surprising, as these tools were quite mature

Case: Mobile Linux

- Media player application by Ixonos
  - New modeling challenge: real-time requirements
    - Playing videos, fast-forwarding, rewinding, pausing…
    - Although it was difficult, real-time support was eventually accomplished at model level
- Keyword execution using Linux accessibility features
  - API access to GUI components
  - Easier and more reliable than in S60 case
- Some minor bugs were found (both during modeling and execution)

Case: Android Phone

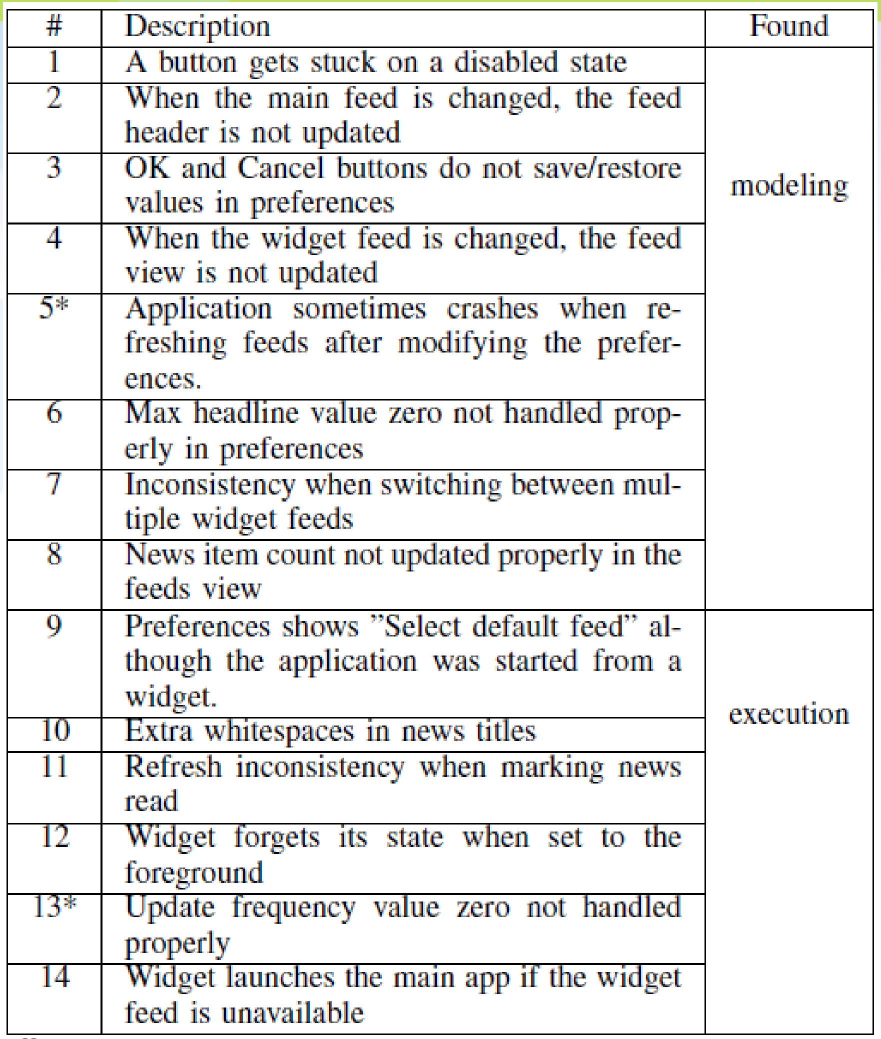
- Messaging, Contacts and Calendar applications
  - Action machines created for S60 were reused
  - Calendar was modeled with ATS4 AppModel and converted to TEMA models with an automatic converter
- Keyword execution was based on A-Tool by Symbio
- Optical character recognition was implemented with MS Office Imaging, which could have been more reliable
- Some bugs were found (both during modeling and execution)

Case: Android Revisited

- BBC News application
  - RSS reader optimized for BBC news feeds
- Self-made test automation for the emulator
  - Based on API access -> improved reliability
- Modeling the BBC News (16 state machines) took a few days time
- With usability improvements in the modeling tool this could be made even faster
- During the case, the application was updated thrice and the platform once (2.1 → 2.2)
- Maintaining the models was fast – no need to update a huge set of test cases

- Random mode was used in the test generation
  - Setting up the test run takes practically no time at all
  - 240 separate test runs lasting over 115 hours in total
  - 27 000 action words – 50 000 keywords
  - The longest run lasted over three hours
  - Average duration was 30 minutes
  - Emulator lost Internet connection in every couple of hours

Bugs Found



#	Description	Found
1	A button gets stuck on a disabled state	modeling
2	When the main feed is changed, the feed header is not updated	
3	OK and Cancel buttons do not save/restore values in preferences	
4	When the widget feed is changed, the feed view is not updated	
5*	Application sometimes crashes when refreshing feeds after modifying the preferences.	
6	Max headline value zero not handled properly in preferences	
7	Inconsistency when switching between multiple widget feeds	
8	News item count not updated properly in the feeds view	
9	Preferences shows ”Select default feed” although the application was started from a widget.	execution
10	Extra whitespaces in news titles	
11	Refresh inconsistency when marking news read	
12	Widget forgets its state when set to the foreground	
13*	Update frequency value zero not handled properly	
14	Widget launches the main app if the widget feed is unavailable	

14 bugs found in total

8 during modeling

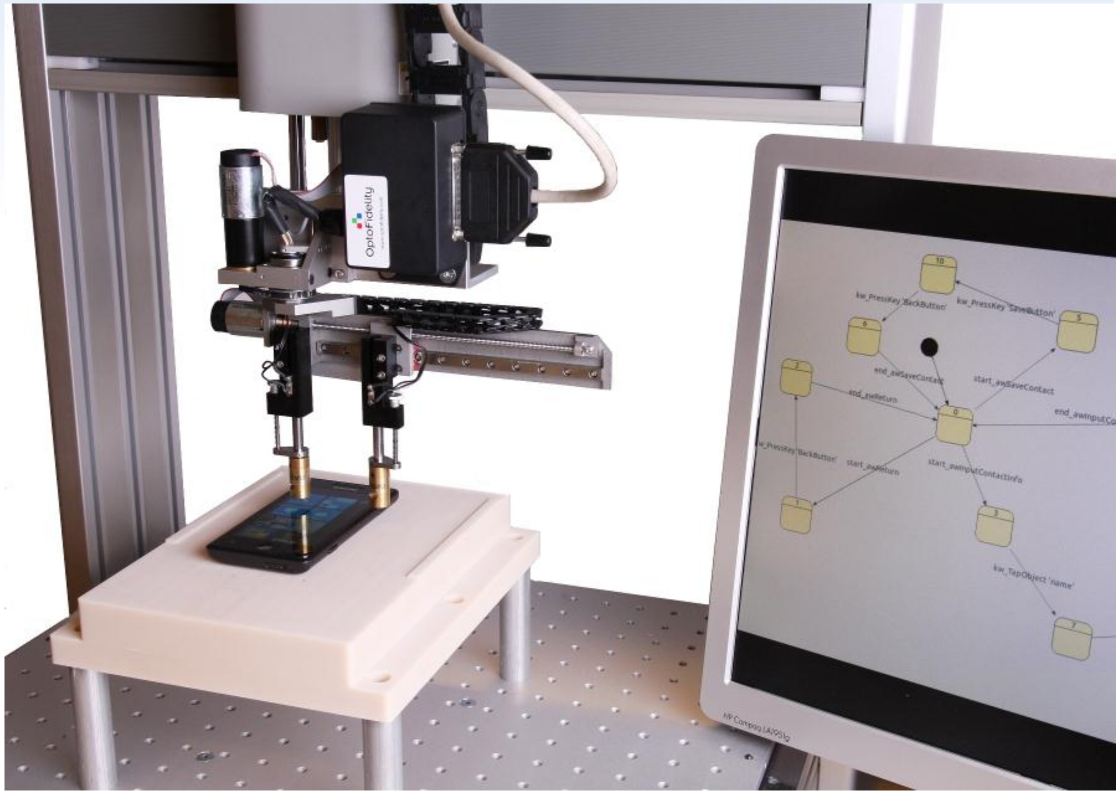
6 in random test execution from the model

Two of the bugs caused the application to crash

Even quite small inconsistencies were found – easily missed by a manual tester

Keyword Execution with a Robot

- Solution for the automated testing of touch display devices
- Simulates real human user interaction with SUT
- The applications are tested in actual devices
- Different sets of robot fingers for device actuation
- Visual verification of the results with a camera and OCV (Optical Character Verification)
- Easy integration with TEMA Toolset



For more information, visit
http://www.optofidelity.com

Thank You!

TEMA TOOLSET NOW AVAILABLE!

Acknowledgements for financiers:
Tekes, Nokia, Ixonos, Symbio, Cybercom Plenware, F-Secure,
Qentinel, Prove Expertise, Academy of Finland (grant #121012)