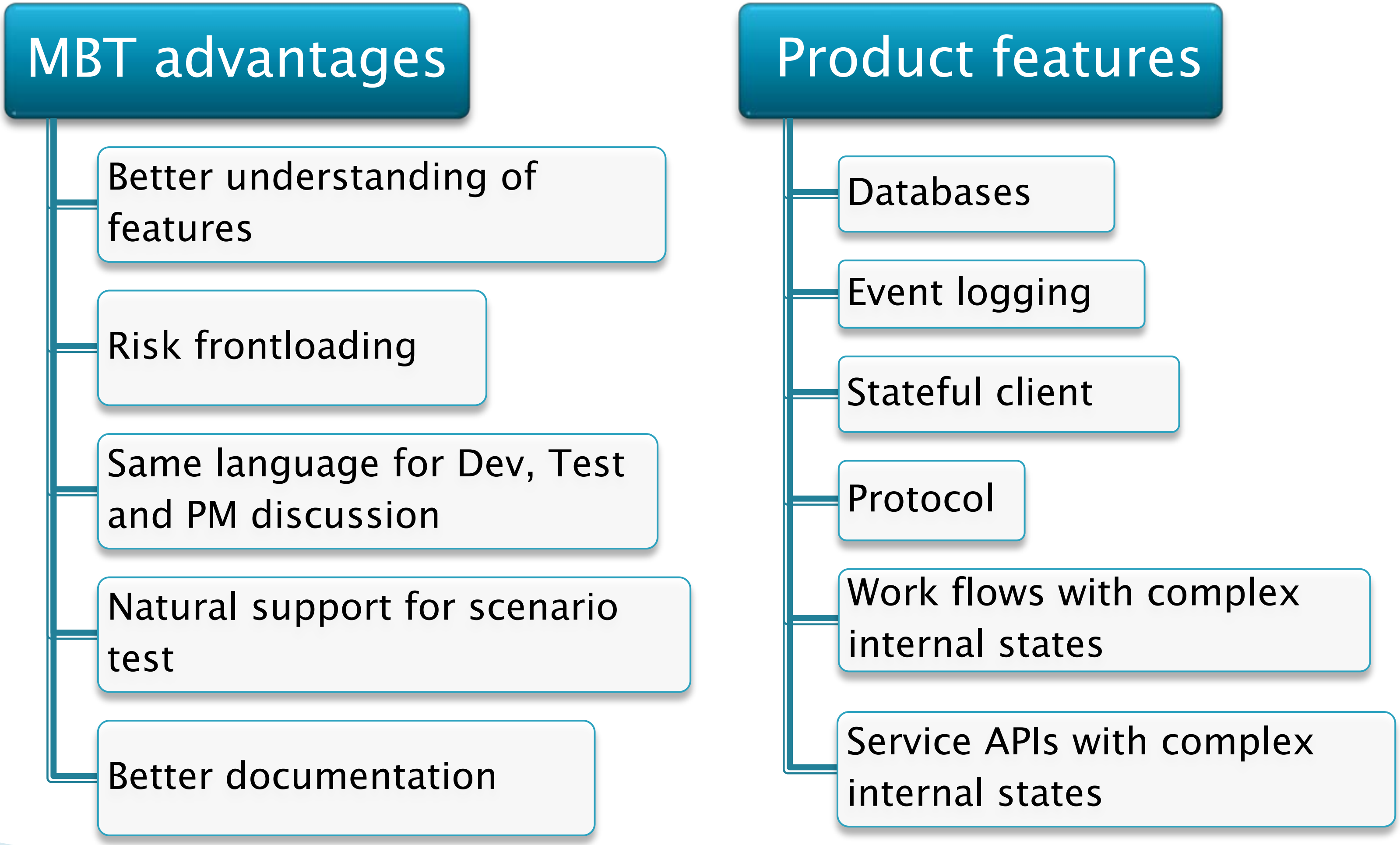
Applying Model Based Testing to the Windows Web Services Software Process

MBT User Conference 2011

Kunal Chopra
Rafael Forsbach
Huiyong Xiao

Why MBT?



MBT advantages

- Better understanding of features
- Risk frontloading
- Same language for Dev, Test and PM discussion
- Natural support for scenario test
- Better documentation

Product features

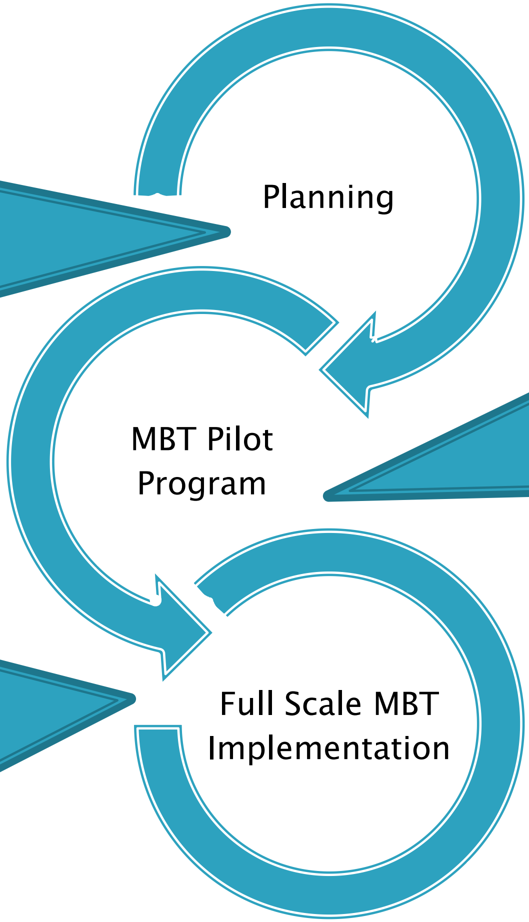
- Databases
- Event logging
- Stateful client
- Protocol
- Work flows with complex internal states
- Service APIs with complex internal states

The 3 Step Process
Plan – Pilot – Implement

Step 1 – Plan

Step 3 – Implement

1. Feedback Incorporated
2. MBT implemented as a Test Methodology
3. It is a another tool in the tester’s toolbox

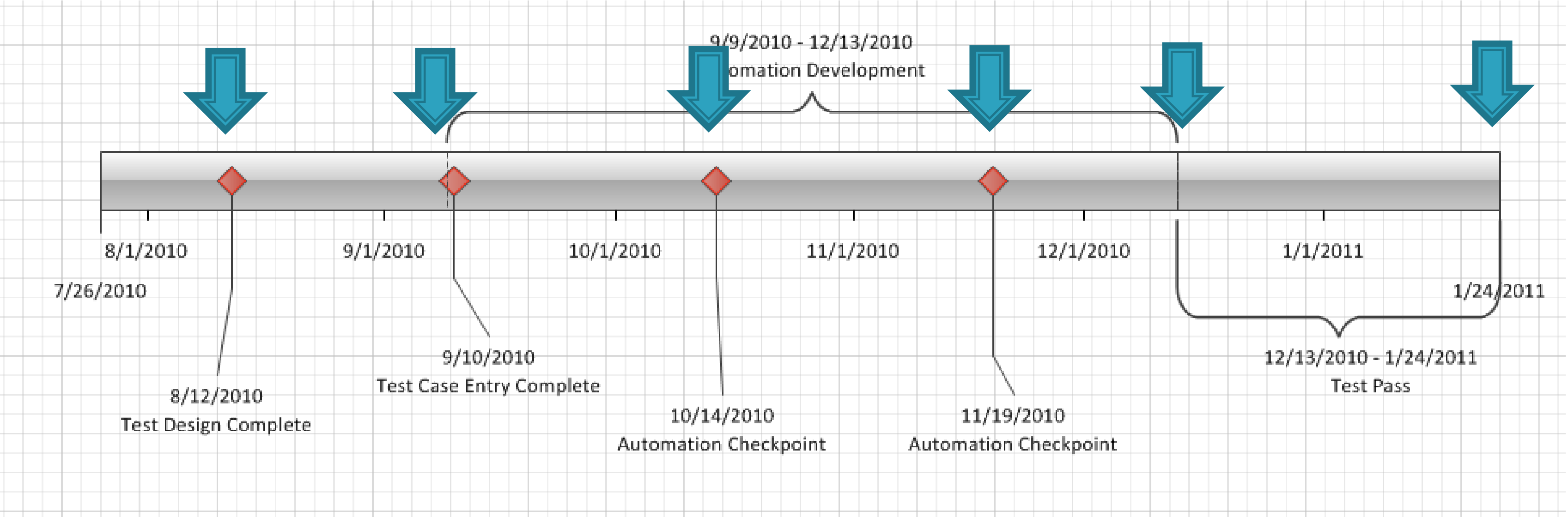


Step 2 – Pilot

1. Establish Pilot Group
2. Apply MBT to Windows Web Services Features

How did we “Fit” in?

Traditional Software Lifecycle in the
Windows Web Services Product Group



Models and System Under Test (SUT)
Integration Code Complete

[illegible]

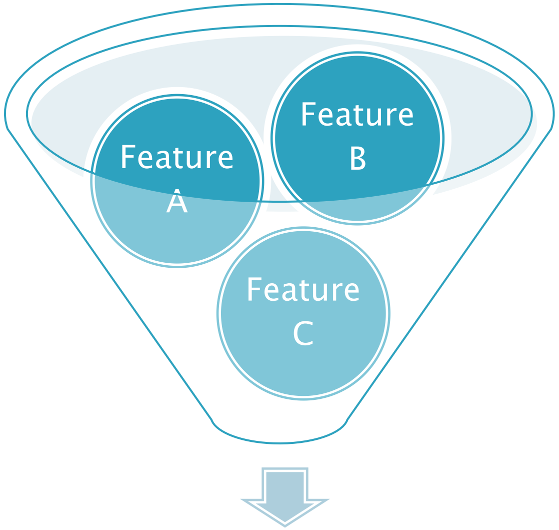
The dates shown are for illustrative purposes only

Product Cycle Phase I
Pre specification

Pilot Results

Reviewed Processes

Commit to it



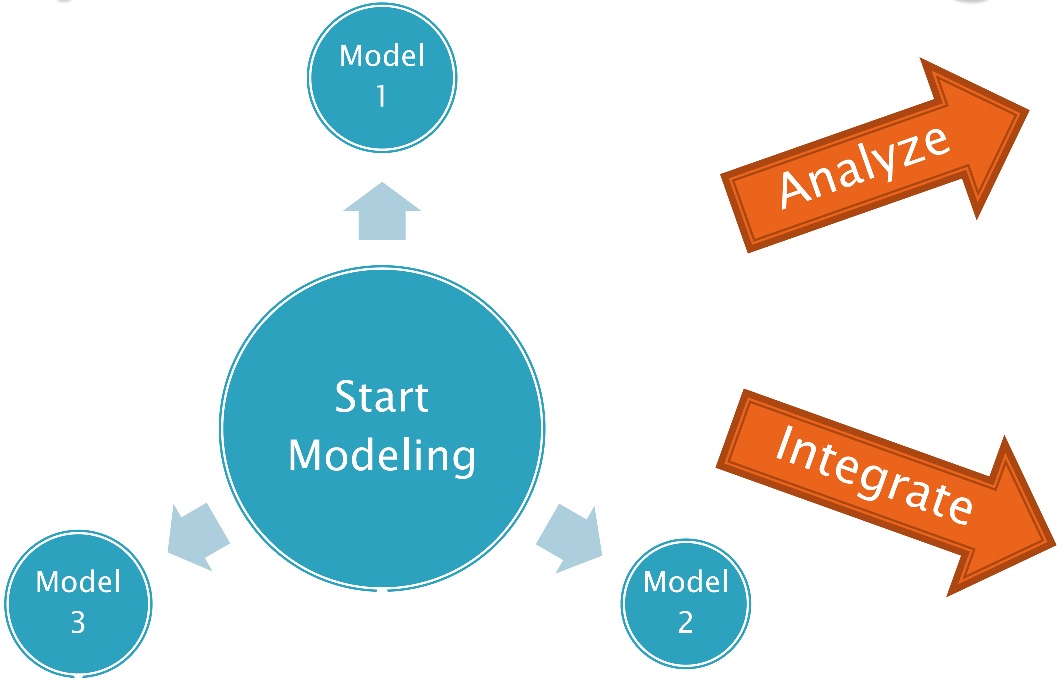
Feature A

Feature B

Feature C

Best MBT Candidates

Product Cycle Phase II Specification & Design

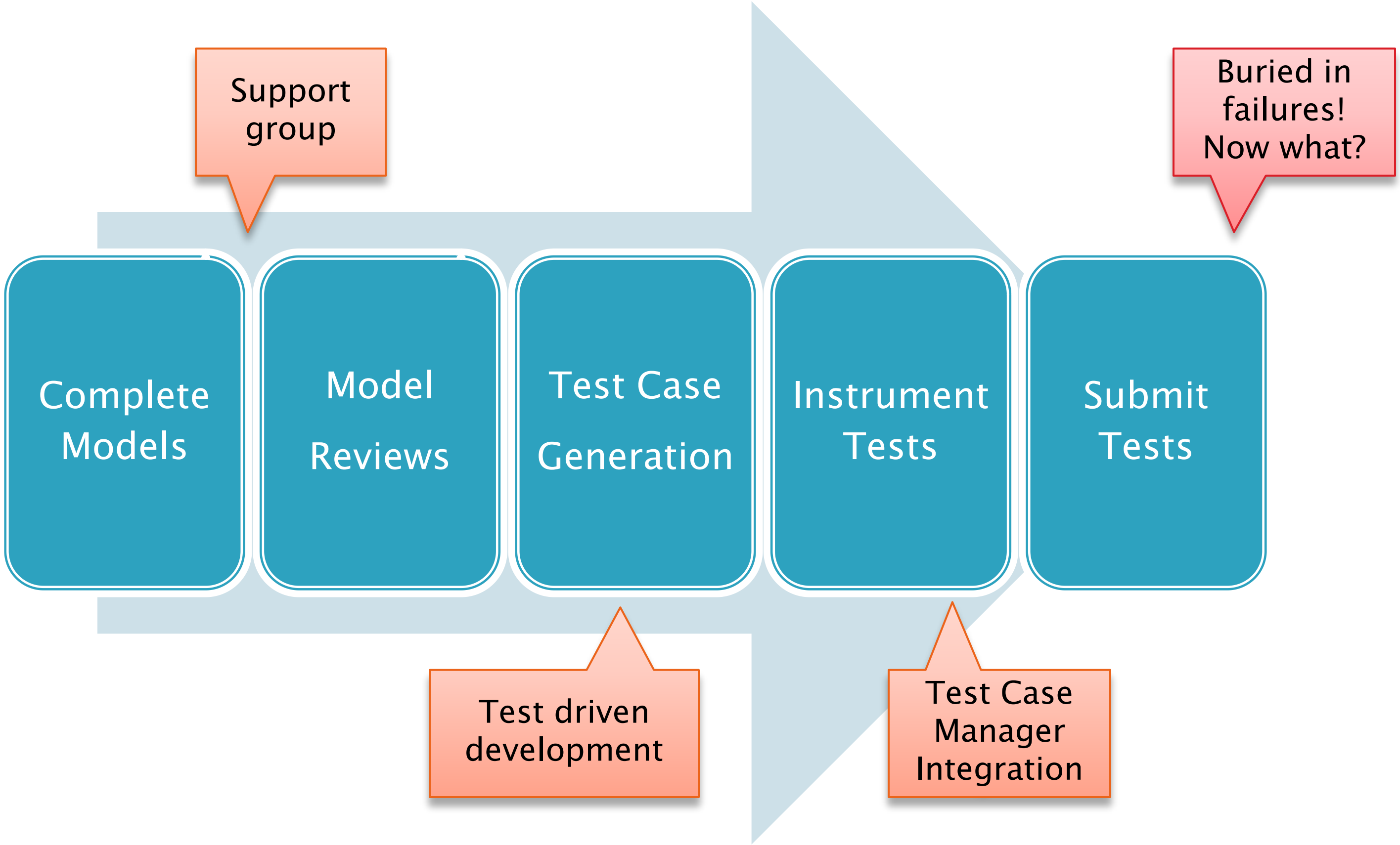


Find design flaws
Functional/
Development
Specifications

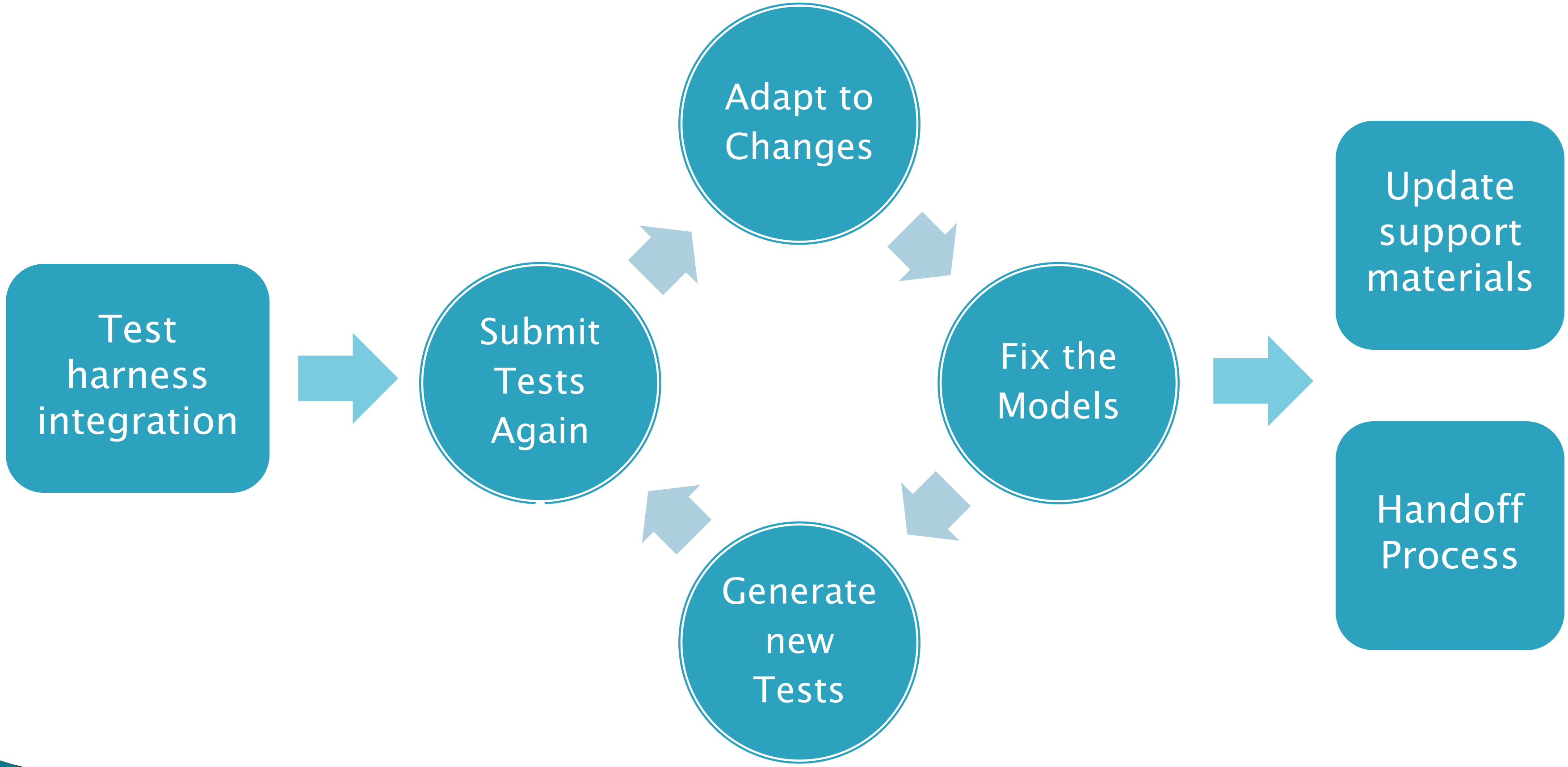
Test Design Specification
Standardize MBT specs
Review with stakeholders

10M states! -> Slice, refactor, rethink model
Support group -> help new MBT testers

Product Cycle Phase III Development



Product Cycle Phase IV
Testing & Stabilization



Test harness integration

Submit Tests Again

Adapt to Changes

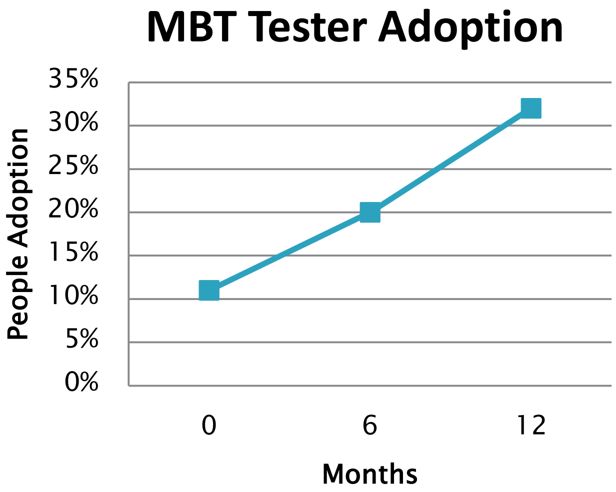
Fix the Models

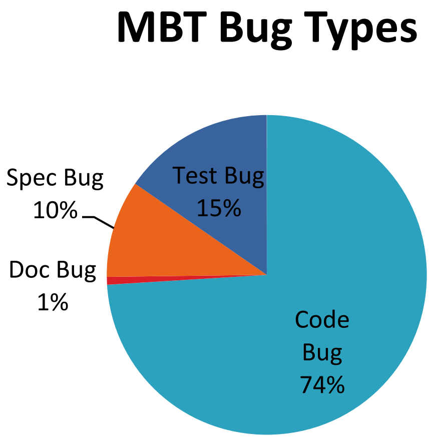
Generate new Tests

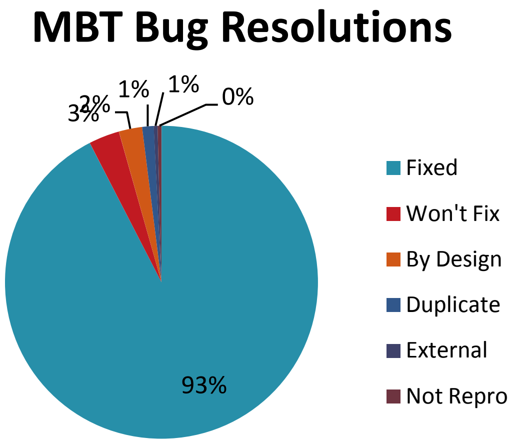
Update support materials

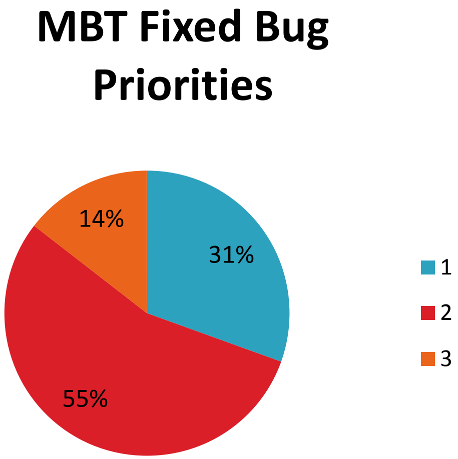
Handoff Process

MBT Adoption Results

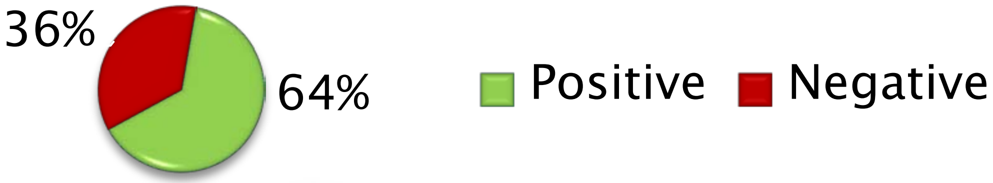

MBT Tester Adoption


MBT Bug Types


MBT Bug Resolutions


MBT Fixed Bug Priorities

MBT Survey and Takeaways



Modeling really helped us review our feature specs with the feature crew using a diagrammatic representation.

There was a lot of enthusiasm from the people involved in Modeling.

Modeling helped me find additional bugs around the UI as a result of generating models.

There were a huge number of spec issues found when modeling

The TDS was very detailed because of modeling and very understandable by the concerned parties.

The model blew up because of capturing all the functionality in one model.

Needed to redo the model a couple of times, so spent some time modeling.

Once you have too many states, it's hard to keep track of them all in your actions.

If you don't build your model piece by piece there is no assurance that you did it correctly

Maintaining the model takes quite a bit of work.

Think small, think scenarios, break down the model to small user interaction pieces.

Enhancing data sets with real world data to populate strings.

It would be great to validate the model quickly e.g. at every level you can find how many nodes are generated.

Would like to see a good process to track code coverage for model test cases.

Would like to see more efforts to see MBT integrated

Questions

More Resources

- Windows 8 Engineering Blog
  - http://blogs.msdn.com/b/b8

- Information on the //BUILD conference
  - http://www.buildwindows.com