

IBM Podcast

[MUSIC]

MATHENY: Welcome to this IBM podcast, Model-based Testing and Smart Product Development. I'm Angelique Matheny with IBM. As products and systems get ever smarter, both business and customer expectations for those products continue to rise. Innovative functionality coupled with high quality is the price of entry to the market for many products. At the same time, business objectives demand that new products are delivered to market in the shortest time with maximum predictability.

This has driven development organizations to adopt best practices work flows and automation, including techniques such as requirements-driven model-based design. Such approaches can dramatically raise design productivity which in turn can lead to test and verification activities struggling to keep up to deliver the required quality.

Model-based testing is a technique that can help to balance testing and design productivity providing an effective way to help meet cost, quality and time to market objectives. So we welcome special guest Dr. Udo Brockmeyer, Chief Executive, BTC Embedded Systems AG and IBM Business Partner.

And today's podcast looks at how model-based testing delivered success for a smart power metered development project. Hi, Dr. Brockmeyer. Welcome to this podcast. Thanks for joining us today.

BROCKMEYER: Yes, hello.

MATHENY: Udo, BTC developed the Trio smart box to provide consumers with full visualization of energy usage habits in real time and allow them to easily understand the impact of their energy habits and leverage off-peak hours to save money and reduce peak loads.

It sounds like it would be very challenging to deliver to consumers high-quality systems that integrate with an energy company's back-end systems. You decided to use model-based testing. What is model-based testing?

BROCKMEYER: You are right. It's a big challenge to deliver high-quality systems. And so far, using a model-based testing technique, was a bit hard to achieve these expectations and goals. And so essentially a model-based testing means is that you're testing very early in your process. In fact, you develop models and start your testing based on models instead of immediately going to develop your software and doing your software testing. So you start with developing models and you verify and test your models.

And the objective is to find problems early, because if you find a problem early you can fix it, and finding and fixing problems early in the process makes it cheaper to fix them, much cheaper than finding them during software coding and software testing and fixing or even later in the [product] systems testing.

If you test your models early, it will lead to much more stable models and stable requirements. And this in fact is a very good starting point for your [real] software development based on the [stable] model.

So several benefits using models, models-based testing, and in addition to that it also enables you much more automation than what you have today in the more traditional development process.

So there's several key aspects in using model-based testing.

The bottom line is for me model-based testing provides you an integrated framework to describe and relate all relevant aspects in your development process, which in our case, model-based testing cases linking together the requirement, model element and the overall view of all the artifacts in the construct of your project.

MATHENY: So, Udo, how was model-based testing applied

for the development of the smart box?

BROCKMEYER: That's a good question. As always, there are many, many possible ways to apply a technique in the project. And of course this holds for model-based testing.

What we did was the following. We started to develop a system view of the smart box using [moderate].

And the system view needs to describe the top level view with the interface of the smart box. So it's the input and output interface of the system which are then later used to satisfy the test scenarios and to execute the tests against the model and the final system.

Based on this system specification and its interfaces, it is possible to automatically generate so-called test architecture, also known as test harnesses, which are needed to do a real testing of your model and your [INAUDIBLE] system. And so this is an essential base test architecture.

And these are generated automatically.

Then we started to write down our, let's say, test scenarios or test cases by using UML sequence diagrams which are very powerful and easy to use means to visually satisfy test scenarios. And they are based on the view of the system and the interfaces and the functions and the events which are going in and out of your system.

These test cases or sequence diagram test cases were then linked to the initial requirements, the user requirements, and these requirements were also part of the model. So in the model we have the requirements, the system view with the interface and the test cases and everything was linked together.

And later, when we had completed the model and much later when we had completed the system, all these sequence diagram test cases were fully automated, executed and in order to perform verification validation of model and system and later in iterative phases, the regression testing.

And last but not least, as always in testing is very important, we put the generated automatically based on the model and the requirements and so on in order to get every day an overview about the status of the project and the [INAUDIBLE] available during the project.

MATHENY: You mentioned benefits earlier. What were the benefits of using model-based testing on the project?

BROCKMEYER: If I think through this, there are a few key benefits when using model-based testing. So one very essential effect is that you can do a visual specification of your test cases using sequence diagrams, for instance,

and a visual specification of your test architecture, test harness, giving you explicit view on your test system.

And this in turn means it's much easier to maintain and extend your test systems compared with the traditional techniques and methods applied on software testing, on the code centric testing, yes. And these test cases could really be highly automated and it could be generated anytime, fully automatically, which that's really the time and cost.

And then I think more it enables much better communication between the system engineers and the software developers and the test engineers and the tester, which in turn leads to much higher quality and much better efficiency.

And the whole process, as I mentioned in my first sentences, if you do this early specification of testing analysis, early testing, earlier, that's quite normal and always the case, and this gives you a big benefit in terms of time, the key benefits from my point of view.

MATHENY: And Udo, our last question today. If someone is developing software for an embedded system today, is it difficult for them to start using a model-based testing approach?

BROCKMEYER: No. I think it is not difficult, because if the engineer's already using a model-based development approach, it's really easy to use a model-based testing. If the engineer is new to model-based development at all, we have of course a learning curve to be able to deal with the model-based testing, but many, many people are able to do this and the learning curve is not too high.

In fact, if you use the right tool for the right job, it's even easier. And we decided to work with IBM Rational Rhapsody and IBM Rational Rhapsody Test Conductor, to work in this project. These tools are mature products and available for this job.

And for us it was the perfect tool set to specify and execute and report the test cases throughout the whole project. It will immediately help us to deal with the challenges -- cost, time and quality for this important project for us.

MATHENY: Udo, thank you so much for sharing your time today. This was very informative, and we really appreciate it.

BROCKMEYER: Thank you very much.

MATHENY: That was special guest Dr. Udo Brockmeyer talking about model-based testing and smart product

development, validating a smart meter device using model-based testing.

To share this podcast with your colleagues, or if you're interested in more podcasts like this one, check out the Rational Talks To You podcast page at www.ibm.com/rational/podcasts.

We'll post a link to the case study titled, Uses Model-Driven Testing for Real-time View of Energy Usage. Be sure to check it out today. This has been an IBM podcast. I'm Angelique Matheny. Thanks for listening. Keep tuning in as Rational Talks To You.

IBM Podcast

[MUSIC] [END OF SEGMENT]