September, 2008

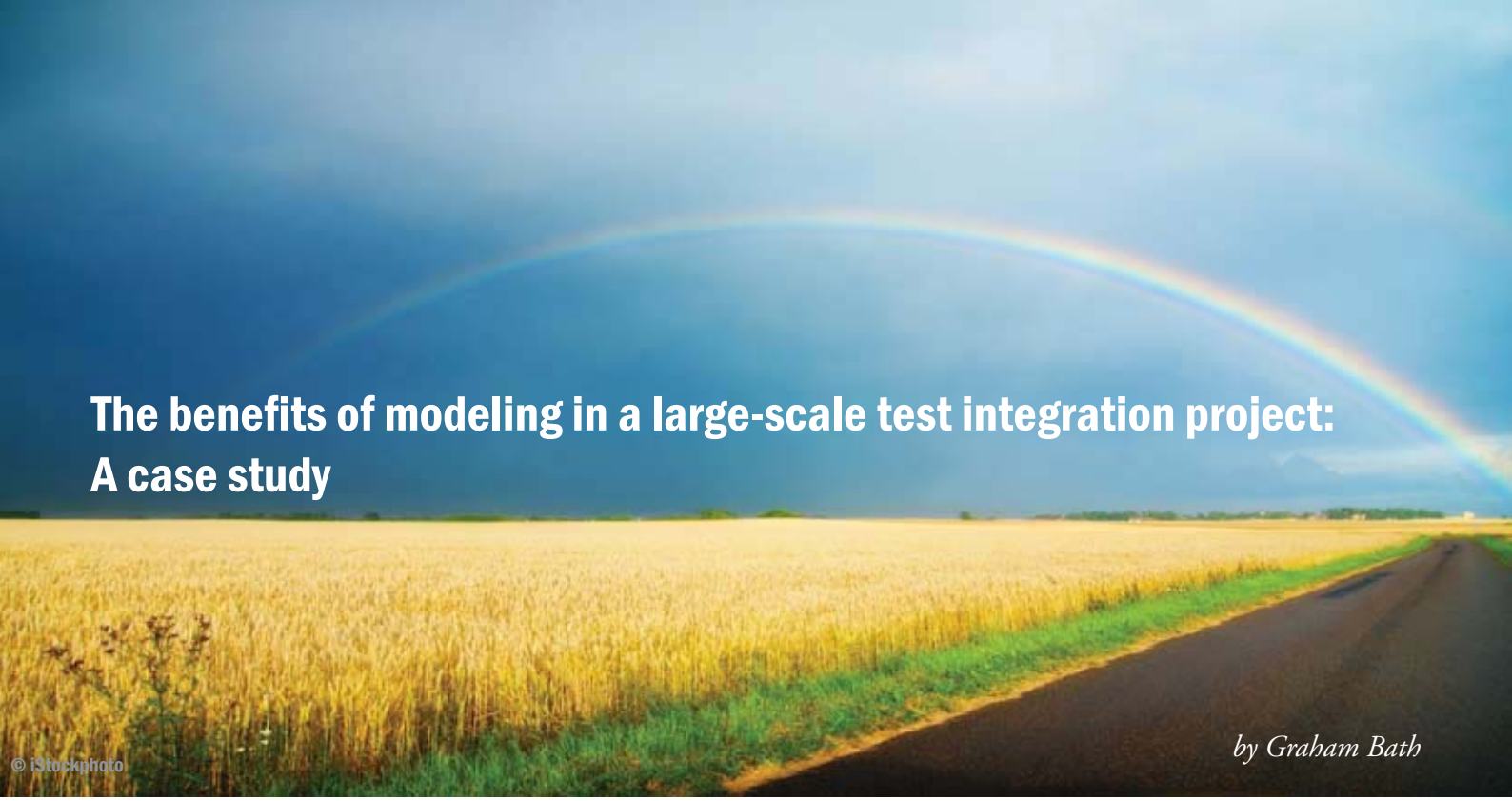
# te testing experience

The Magazine for Professional Testers

# Test Techniques in practice -

## Do they help?
## Why do we often test without them?

printed in Germany
free exemplar
www.testingexperience.com
ISSN 1866-5705



# The benefits of modeling in a large-scale test integration project: A case study

*by Graham Bath*




## Abstract

This case study describes the benefits obtained and the lessons learned from introducing a model-based approach into one of europe's largest system integration testing projects, which includes off-shore components and a staff of more than 400.

An industrial-scale research project with three distinct phases was established to manage the test process modifications resulting from adopting a model-based approach and to ensure that practical and business factors remained in focus throughout. In this paper the three phases are described in terms of the concept used for achieving defined objectives, the lessons learned, and the actual benefits obtained.

The results of the case study show that the model-based approach is an effective instrument for enabling a wide range of improvements to be achieved. The customer requirements process has been integrated into the testing process and a seemless, tool-supported process from requirements through to test specification has been achieved.


## 1 Introduction: The trend to highly integrated systems

Monolithic systems are a dying breed. The trend these days is towards system architectures which are sometimes best described as "systems of systems". The reasons for this could fill a whole article, but certainly the benefits of software re-use, the availability of standard software packages, the flexibility and scalability offered by heterogenous architectures and the "mix and match" possibilities offered by web-based software services (SOA) are among the principal driving factors. But there's a price to be paid for developing architectures like this; the task of testing and providing quality-related information to decision makers is generally more complex.

It's against this background of "systems of systems" that model-based approaches was evaluated and introduced. This article describes a number of the benefits which were achieved and the strategy which was followed. Since the paper is based on industrial experience, it would come as no suprise to learn that there were problems along the way, so I'll be sharing some "lessons learned" as well.

## 2 Deciding on an approach: Somewhere over the rainbow...

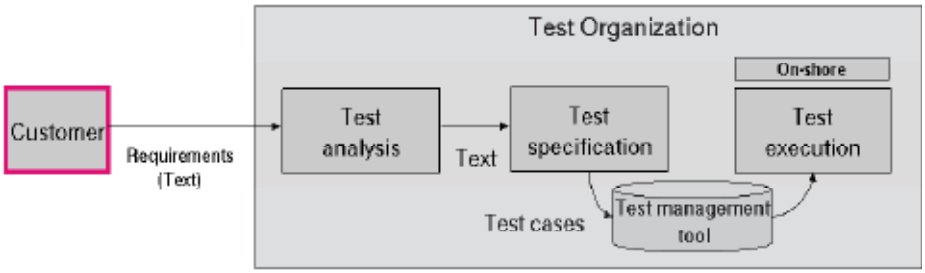
There's an old saying that if you can find the point where a rainbow ends you'll find a pot of gold. The problem is, if you set off towards your rainbow's end you're chasing a moving target, which may ultimately disappear before your very eyes. We never actually reach that mystical point where the pot of gold is buried. In IT we've all been there before haven't we? We've chased pots of gold called "tools give full test automation at the press of a button", "mainframe applications will be entirely replaced by e-commerce" and, more recently, "just send everything offshore; it's sure to be more efficient". We probably all know by now that there's more to it than that. Yes, tools can help automate certain types of testing efficiently. Yes, e-commerce is a major step forward for user-centric applications and, yes, offshore can yield substantial benefits if managed properly. But we've been around long enough to know that these new developments follow a cycle which starts with euphoria, passes through the "valley of dissolusionment" and finally reaches the a level of practical usage (which is generally somewhere between those two extremes).

At T-Systems, Test Factory we were acutely aware when we set up our study that using a model based approach for achieving our testing objectives could well end up like chasing the next pot of gold. So we need to be careful and not be too euphoric at the start and we adopted an iterative approach, where learn-ing from experience was a central element.

Oh, and we needed a realistic objective: "To achive synergies and efficiencies by introducing a model-based approach and integrating this into our established standard testing process; from planning through to execution."

### 2.1 The starting point

The existing test process is based on the description provided by the Interna-tional Software Testing Qualifications Board (ISTQB). It was not an objective to replace this test process, but we certainly wanted to make it operationally more efficient. The diagram below illustrates the aspects of the test process which were identified as candidates for efficiency improvements (note that this diagram does not show all of the steps in the test process).



The primary areas for potential improvement were identified as:
- the customer requirements interface
- the analysis and specification of test cases
- the use of off-shore organizational components

### 2.2 The concept

To investigate the benefits of using models in testing a research project was set up which works closely with both operational projects and the local university, (in this case the Technical University of Munich). The clear intention was to obtain buy-in from the projects right from the start, avoid the problems of developing new concepts in "ivory towers" and to make use of existing academic expertise in the field of modeling.
The project plan we developed had three phases, each with approximately six month's duration. The objectives for each phase were only set after learning from the previous phase.
The three phases we identified are shown in the table below:

| Project phase | Name of phase |
|---|---|
| 1 | "Let's get modeling" |
| 2 | "Integrate, integrate" |
| 3 | "Automate, where it makes sense" |

Details of the project phases will be discussed in later chapters. Before we started with phase 1 though, it was important to choose a suitable pilot project.

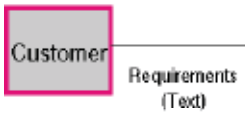
### 2.3 Choosing a pilot project

The objectives we set ourselves could only be achieved if a representative project was used as a pilot. Of course, it's possible to reduced risks by using a relatively simple project for piloting, but what value are the results afterwards? Will they scale to the types of large, complex project we have to test?
What we needed from our pilot project were a number of attributes so we could be sure the approach was going to provide benefits when rolled out. We selected a major testing project as our pilot because it gave us good coverage of these attributes.

| Required attributes for pilot | Characteristics of the chosen project |
|---|---|
| Technical complexity: Large numbers of applications and interfaces. | Over 70 inteconnected applications |
| Business processes complexity: Several applications communicate with each other to implement business processes | Mostly complex business processes running over multiple applications. Many "end-to-end" tests are performed. |
| Organizational complexity: Geographically distributed organization, including off-shore elements | The project is performed at several locations in Germany and in India. |
| Real business case: There has to be a reason for doing this. We're not interested conducting a purely academic exercise. | With over 400 testing professionals involved in the project, any improvements to efficiency will lead to substantial savings. |

### 2.4 The chosen pilot project

The fundamental task of the pilot project is to provide functional and operational acceptance testing of the entire system stack prior to entering production. Regular releases of software are tested and quality-related information provided on which production decisions are taken. For each release the testing organization receives a number of text-based "solution documents" from the business owners which are used as the test basis.




## 3 Phase 1: "Let's get modeling"

In this first phase we looked carefully at the issue of modeling in general and established an overall concept for introducing model-based approaches to our test process.

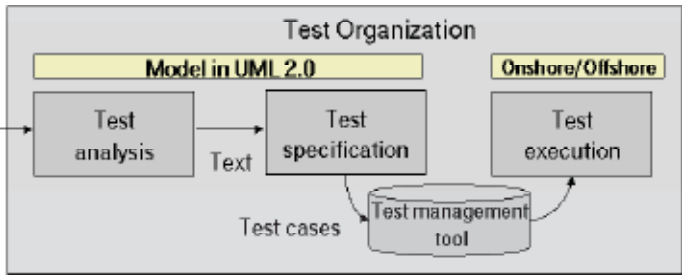
### 3.1 The overall concept

Central elements of the concept were:
- Use of UML 2.0 to model the test basis. In this step (which we called "design for test" ) the business requirements provided in the customer's solution documents were captured as UML2.0 Sequence Diagrams and Class Diagrams.
- Use of the UML-2 Testing Profile (U2TP) to take over the UML diagrams and extend them for test purposes. The result is a Test Model (equivalent to a test specification) which contains information about the System under Test (SUT) and each test case.
- Use the Test Model as a basis for test execution. This could be performed either manually or (potentially) used by tools for the generation of automatically executable tests.
- For each of these steps the roles and tasks for on-shore and off-shore staff are clearly identified.
- Identification of business processes for implemention in UML. These needed to be stable, well understood processes and the responsible test designers needed to be willing to take on the piloting task in a "live" project.
- Training in UML 2.0 provided by the Technical University of Munich. A total of 12 experienced staff were selected to perform the phase 1 modeling
- Selection of the Enterprise Achitect (AE) tool for the UML 2.0 modeling. The selection decision was based primarily on ease of use and low license costs.
- Modeling of 30 selected business processes as UML 2.0 activity diagrams.
- Modeling of the messaging and communications between individual systems using UML 2.0 sequence diagrams.
- Getting feedback from industry: The concept was presented at the International TTCN-3 conference in Berlin in 2006 and at the System Engineering Conference "SE08" in Munich in 2008.
- The basic concept for phase 1 is shown in the diagram below.

### 3.2 Strategies for off-shore compared

Before we could really settle on the concept outlined above we first needed to consider the issue of off-shoring, which is an essential element of our testing organization. What would be the best way to capture the customer's requirements as a test specification so that our off-shore teams could use it efficiently and effectively? We looked at three different options:
1. **Text-based**: This is the existing baseline. Textual test cases were generated onsite, translated along with other specification documents and then sent to offshore for execution.
2. **Model-based on-shore**: Test design and specification both conducted on-site and then executed off-shore.
3. **Model-based partial off-shore**: Test design conducted on-site. The design is then refined and the test specification created off-shore. After a review with the on-shore team the test are then executed off-shore.

The text-based approach (option 1 above) in which test cases are translated from German into English was known to be inefficient (that's one of the primary reasons for looking at modeling). Here are a few reasons why:
- Lower proportion of work carried out off-shore.
- High levels of communication due to queries regarding the test



specifications and incorrectly raised defect reports.
- Lower than expected levels of productivity.
- Insufficient knowledge transfer.
- The translations were often a major source of error and off-shore teams often found themselves (incorrectly) interpreting their meaning.
- Insufficient knowledge transfer.
- The off-shore staff were not involved in the testing process early on, which lowered motivation.

### 3.3 Benefits of the model-based approaches

A structured questionnaire was used to support the results obtained from the first phase. Even though this evaluation was primarily subjective in nature, it enabled all those participating in the innovation project to express their views on the benefits obtained. Did they "feel" that there had been improvements made? Could they perform their work better than before? Was the effectivenss of the testing now better or worse than before? Questions like these were important for giving us the confidence for taking further steps.

Here are the primary benefits which emerged from the survey:

- **Increased efficiency**. In some areas, efficiencies of over 50% were achieved compared to a testing approach without modeling. These efficiencies arose from a number of different sources (see below).
- **Better communications**. Perhaps one of the most significant benefits was in better communications within distributed testing teams. Testing staff in Germany and India could communicate via UML diagrams instead of pure text. There were fewer misunderstandings; fewer telephone conferences and a general improvement in the process of off-shore governance.
- **Higher motivation**. Levels of motivation for off-shore staff increased because they became far more involved in the test process as a whole.
- **Better knowledge capture**. The capture of requirements documents as models relieved the problem of "tacit" knowledge where particular individuals are in possession of key business process know-how, which is often not fully documented.
- **Stimulus for offshore**. In general a model-based approach proved to be a significant "enabler" for the off-shore testing process. Whilst many papers and presentations have highlighted the automation-based benefits of a model-based approach to testing, the benefits we obtained pointed to significant improvements in other aspects of the software development process such as in knowledge management, team motivation and off-shore governance.

### 3.4 Lessons Learned from Phase 1

Not everything worked out absolutely as expected in this first phase. In particular, the following problems arose and a number a lessons were learned:

- The modelling task was actually more time-intensive than predicted, despite the training courses and the availability of expert support. We realized quite quickly that modelling guidelines are absolutely essential, especially regarding the levels of abstraction required.
- The use of Enterprise Architect (EA) as a modelling tool proved to be a double-edged sword. EA is easy to learn and use and allows the modeller complete freedom to capture requirements as models. This freedom, however, can cause problems, even when modelling guidelines are available. First of all, the modeller has to keep referring to those guideleines and, secondly, control mechanisms have to be in place to ensure that they are being correctly applied. One particular problem area was fixing the level of abstraction required in the model. This started to erode some of the efficiency gains.
- The levels of defects detected using model based test specifications with off-shore testing teams remained relatively constant.

### 3.5 Just a minute: why is the test organization doing this?

Phase 1 of the project was initiated in order to improve the efficiency of our testing organization's process. One of the principal tasks performed to achieve this was arguably, however, not a testing task at all. The modeling of requirements in UML was a necessary step to enable other benefits to be gained for the testing organization. In principal it's a desirable skill for a testing organization to be able to model requirements, but we were still left at the end of phase 1 with this basic testing overhead.

## 4 Phase 2: "Integrate, integrate"

During phase 1 of the project we became aware of similar kinds of project being conducted by our customer. Their idea was to model their requirements so that development organizations could use them. For us this was welcome news; it would be a relatively small conceptual step to also include the testing organization in their plans.

The contours of the next phase in the project began to take shape when we learned that a modelling tool had already been selected by the customer. The tool, MID-Innovator, would not only provided the "bridge" between our customer and the testing organization, it would also help to strengthen our modeling process and make it more precise compared to our earlier approach of using Enterprise Architect with modeling guidelines.

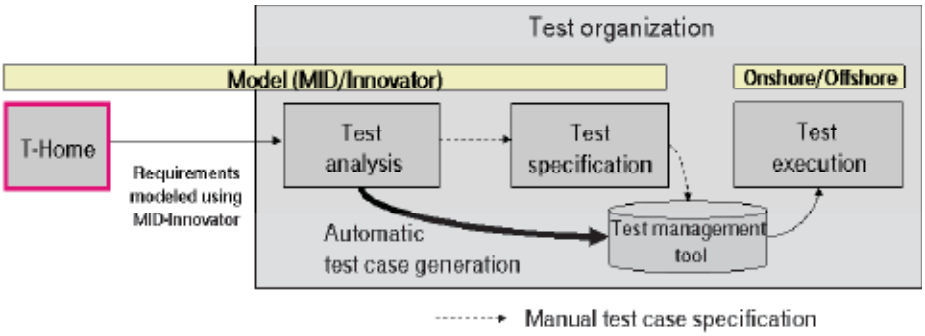
After discussions with our customer the potential benefits of project phase 2 were clear; if we could integrate the MID-Innovator modelling tool into our testing process we would have a seemless, model-based interface with our customer. The test organization would no longer need to model the customer's text-based requirements and significant gains in efficiency could be achieved. In addition, the use of this particular tool held the prospect of being able to automatically generate test cases from the models.

### 4.1 Concept for Phase 2

Introducing a new tool like MID-Innovator into the testing process triggered a number of integration tasks (hence the name given to this phase!):

- Integrate test-specific aspects into the requirements specification process. To do this we would implement a Domain Specific Language (DSL) so that both customer and test organization could agree on a common domain-specific language for modelling. The test organization would assist in creating this DSL so that our standard test process could be taken into account.
- Integrate the MID-Innovator tool into our own tool set. This would mean creating a new XML-based interface between MID-Innovator and our standard test management tool.
- Integrate the modelling tool into our existing testing process. This would mean altering a number of steps in the process, including test planning and test specification.
- Integrate the modified test process into off-shore components of our organization
- Establish a knowledge management concept which would ensure that best practices and lessons learned from the innovation project could be shared across the entire testing organization. To achieve this a Wiki would be developed using the Confluence product.
- Model the same test scenarios used in phase 1 to allow comparisons to be made.

The basic concept and objectives for phase 2 are summarized in the diagram below:



Note that the option to generate test cases automatically from the MID-Innovator-based models does not mean that the option to define test cases manually, using UML (as in phase 1) or indeed with any other valuable approach is eliminated.

### 4.2 Benefits and Objectives of Phase 2

At the time of writing this paper the second phase of the innovation project was under way. The benefits we expect from this phase include:

- Efficiency gains resulting from the use of a common DSL
- Efficiency gains resulting from the use of a common modelling tool
- Fewer defects found as a result of requirements problems
- More efficient generation and administration of test cases
- Easier monitoring and verification of achieved test coverage levels
- Better sharing of knowledge across the testing organization

In addition to the benefits noted above, we want to provide answers to a number of questions:

- How easy is it to create a common DSL and what are the primary areas to pay particular attention to.
- What proportion of the lessons learned are applicable specifically to the pilot project itself and what can be defined generically so that other projects can benefit?
- Will the modelling tool be able to generate a high-value set of test cases based on risk, or will expected efficiencies be eroded by an "explosion" of automatically generated test cases.
- Does our modified testing process handle requirements changes efficiently?
- How does this approach compare to the results of phase 1, where models were created in UML using Enterprise Architect.
- Is our customer satisfied with the seamless model-based approach?

The results of phase 2 will be documented as a business case which will enable management to decide on a general roll-out and the objectives of phase 3.

## 5 Phase 3: "Automate where it makes sense"

Assuming that we are satisfied with the results from phase 2, the next phase planned will look at automation. As the name for this phase suggests, we will be performing evaluations of automation concepts which integrate to our testing process.
Modular test automation concepts will be evaluated. These may be based on keywords defined in the DSL "meta-model", or may make use of existing concepts. Where a clear business case can be made for automation this will be implemented, potentially using our off-shore organization.
If we have been able to a well-defined DSL in phase 2, the opportunity may also exist for using tool-support to generate automatically executable test cases. There are already some products available to do this, but for the moment we'd like to let those products mature a little.
One of the principal objectives to be answered in this phase will be to define modeling frameworks at a high enough level of abstraction to make the benefits generally available in other projects and for other customers.

## 6 Summary and conclusion

We're experienced enough to know that "magic solutions" rarely become reality and that the day is unlikely to come where we can represent requirements in model form, press button "B" and then watch all the tests run automatically. Recalling the rainbow anecdote at the start of this paper, we're a long way off from this particular "pot of gold". However, we have shown in this industrial scale pilot project that some major benefits can be obtained from modeling in a complex large-scale integration project.
Here are the main benefits in summary.

- Modeling enables the benefits of off-shoring to be realized.
- Models can build bridges between customer and test organization, especially if Domain Specific Languages and common modeling tools can be utilized. Major efficiency gains can be achieved as a result.
- Modeling is an effective instrument in achieving effective knowledge management.
- Modeling can help introduce rigour into testing, especially if tools can be properly integrated into the test process.

I look forward to informing you of progress. Who knows, we may find the occasional "pot of gold" along the way!

## Biography

Graham's experience in testing spans over 25 years and has covered a wide range of domains and technologies. As a test manager, he has been responsible for the testing of mission-critical systems in spaceflight, telecommunications and police incident-control. Graham has designed tests to the highest level of rigor within real-time aeropsace systems such as the Tornado and Eurofighter military aircraft. As a principal consultant for the T-Systems Test Factory he has mastered the Quality Improvement Programs of several major German companies, primarily in the financial and government sectors. In his current position, Graham is responsible for the training programme and for introducing innovative testing solutions to his company's large staff of testing professionals.

Graham is a member of the authoring team for the new ISTQB Advanced Level Certified Tester and is a long-standing member the German Testing Board, where he chairs the Advanced Level and Expert Level Working Parties.

Together with Judy McKay, Graham has co-authored the recently published book "The Software Test Engineer's Handbook", which is a study guide for the ISTQB Test Analyst and Technical Test Analyst Advanced Level certificates.