STARTHILFE FÜR MODELLBASIERTES TESTEN: ENTSCHEIDUNGS-UNTERSTÜTZUNG FÜR PROJEKT- UND TESTMANAGER

Modellbasiertes Testen ist eine Technik, die durch den Einsatz von abstrakten Modellen und geeigneten Algorithmen bestimmte manuelle Aktivitäten, wie z. B. das Testdesign, unterstützt. Die Einführung von modellbasiertem Testen hat das Potenzial, die Testüberdeckung durch die automatische Generierung von Testfällen zu erhöhen und somit das Vertrauen in die Software zu steigern. Den Einsparungen von manuellen Testaktivitäten steht aber der Zusatzaufwand für die Erstellung der Modelle gegenüber. Projekt- und Testmanager stehen also vor der Frage, ob modellbasiertes Testen in ihrer konkreten Testorganisation eine sinnvolle Investition darstellt. Dieser Artikel erklärt die wesentlichen Begriffe zum Thema modellbasiertes Testen und gibt Entscheidungsträgern eine heuristische Entscheidungshilfe an die Hand.

Beim Thema *Modellbasiertes Testen (MBT)* hat man manchmal den Eindruck, dass hier alter Wein in neuen Schläuchen verkauft wird (vgl. [Win09]). Bereits 1999 schrieb Robert Binder, dass Testen *immer* modellbasiert ist (vgl. [Bin99]). Warum wird also derzeit so viel über diese Testtechnik diskutiert und geschrieben? Ein Grund ist, dass Werkzeuge und modellbasierte Techniken in der Softwareentwicklung in den letzten Jahren reifer geworden sind. Auch Industriestudien belegen, dass MBT bereits in einigen Unternehmen erfolgreich eingesetzt wird (vgl. z. B. [Pre05-b]). Trotzdem dürfen sich Projekt- und Testmanager in ihrer Entscheidung „pro bzw. kontra modellbasiertes Testen" nicht auf die neueste Mode und auf Marketingversprechungen stützen, sondern müssen nüchtern eine Kosten-Nutzen-Betrachtung durchführen.

Das International Software Testing Qualifications Board (ISTQB) definiert fünf Phasen im fundamentalen Testprozess (vgl. [ISTQB]):

1. Planung und Steuerung
2. Analyse und Design
3. Realisierung und Durchführung
4. Auswertung und Bericht
5. Abschluss

Die unterschiedlichen Aktivitäten im Testprozess können manuell, semi-automatisch oder voll-automatisch durchgeführt werden. Besonders bei den Phasen (2) und (3), die wir im Folgenden als Kernphasen bezeichnen, unterscheidet man drei grundsätzliche Testtechniken:

- manuelles Testen
- skriptbasierte Testautomatisierung, bis hin zur schlüsselwortgetriebenen Ansätzen
- modellbasiertes Testen (MBT).

Alle drei Techniken können unabhängig voneinander – aber auch kombiniert – eingesetzt werden.

Manuelles Testen ist immer noch das in der Industrie am häufigsten eingesetzte Verfahren. Dabei erstellt ein Domänen-Experte die Testfälle und führt diese manuell aus. Die Testergebnisse werden mit Hilfe des Domänen-Wissens des Experten unmittelbar ausgewertet. Als besonders aufwändig und fehleranfällig gelten dabei die Aktivitäten während der oben genannten Kernphasen (vgl. [Pol02]) aufgrund des hohen manuellen Arbeitsanteils bei der Erstellung und der Durchführung der Testfälle. Auch der manuelle Vergleich der Ist- und Soll-Er-

der autor

Baris Güldali
(E-Mail: baris.gueldali@s-lab.upb.de)
ist wissenschaftlicher Mitarbeiter beim Software Quality Lab (s-lab) der Universität Paderborn. Er ist Sprecher des Arbeitskreises TOOP/MBT in der GI-FG TAV.

Dr. Stefan Jungmayr
(E-Mail: jungmayr@testbarkeit.de)
ist bei einem deutschen Automobilzulieferer in der Forschung und Vorausentwicklung tätig.

Michael Mlynarski
(E-Mail: michael.mlynarski@s-lab.upb.de)
ist wissenschaftlicher Mitarbeiter beim Software Quality Lab (s-lab) der Universität Paderborn. Er ist stellvertretender Sprecher des Arbeitskreises TOOP/MBT in der GI-FG TAV.

Stefan Neumann
(E-Mail: stefan.neumann@imbus.de)
ist Consultant für Softwarequalitätssicherung bei der imbus Rheinland GmbH.

Prof. Dr. Mario Winter
(E-Mail: mario.winter@fh-koeln.de)
lehrt und forscht am Institut für Informatik der FH Köln in den Bereichen Softwareentwicklung, Projektmanagement und Qualitätssicherung. Er ist Sprecher der GI-FG TAV und Gründungsmitglied des GTB e.V.

gebnisse während der Testauswertung ist eine aufwändige und fehleranfällige Aktivität:

- *Skriptbasierte Testautomatisierung* basiert auf der Erstellung von Testskripten und ihrer automatisierten Durchführung. Die Testskripte werden manuell erstellt oder mittels so genannter *Capture/Replay*-Werkzeuge aufgezeichnet. Durch die automatisierte Testdurchführung wird der Testprozess schneller und wiederholbar. Allerdings wird die Wartung und Anpassung der Testskripte aufwändig, wenn sich die Schnittstellen (z. B. die grafische Benutzungsschnittstelle) der Software, über die der Testtreiber mit der Software interagiert, häufig ändern.
- *MBT* ist eine Technik, die den Testprozess – insbesondere die oben genannten Kernphasen – durch den Einsatz von Modellen systematisiert und automatisiert. In Anlehnung an *Model-Driven Engineering (MDE)* werden auch beim MBT Modelle (so genannte *Testmodelle*) zur Unterstützung der Testaktivitäten genutzt. Aus den Testmodellen werden Testartefakte – beispielsweise Testeingaben, erwartete Testergebnisse und Testskripte – generiert. Dabei werden die Testmodelle entweder aus der Spezifikation der zu testenden Software abgeleitet oder aus den Kundenanforderungen für Testzwecke explizit erstellt (vgl. [Utt07], [Bak08], [Pre05-a]) [1].

MBT hat mehrere Vorteile gegenüber manuellem Testen bzw. skriptbasierter Testautomatisierung (vgl. [Rob00]):

- MBT ermöglicht es, die aufwändigen Aktivitäten des manuellen Testens durch geeignete Algorithmen zu systematisieren und durch Werkzeuge zu automatisieren. Dadurch wird der Testprozess wiederholbar und effizienter.
- Durch MBT können Testskripte für eine skriptbasierte Testautomatisierung aus den Testmodellen generiert werden. Bei Änderungen in den Anforderungen oder in der Spezifikation ist es ausreichend, die Testmodelle an die Änderungen anzupassen und die Testskripte neu zu generieren. Dabei wird angenommen, dass die Änderung der Modelle weniger aufwändig ist als die manuelle Änderung der einzelnen Testskripte (vgl. [Pre05-b]).
- Die Modelle des erwünschten Verhaltens des Prüflings (Systemmodell) oder seiner Umgebung (Umgebungsmodell) können während der Testdurchführung als Test-Orakel benutzt werden, welches das beobachtete Verhalten des Prüflings mit dem erwünschten Verhalten vergleicht und über den Erfolg des Testfalls entscheidet.

Neben Vorteilen hat MBT aber auch Nachteile:

- Zur automatisierten Generierung von Testartefakten müssen Testmodelle detaillierte Informationen über das zu testende System und seine Umgebung enthalten, was den Aufwand bei der Testmodellierung erhöht.
- Die Testmodelle müssen – im Gegensatz zu den Entwicklungsmodellen des Systems – um zusätzliche testspezifische Informationen angereichert werden (vgl. hierzu ausführlich [Pre05-a]).
- Der Wartungsaufwand bei der skriptbasierten Testautomatisierung wird in MBT auf die Wartung von Testmodellen übertragen. Wenn Änderun-

Begriff	Bedeutung im Kontext MBT
Testmodell	Ein Testmodell ist eine Abstraktion von a) dem zu testenden System b) seiner Umgebung, c) von beiden oder d) von den Testfällen oder dem Testrahmen, die von Testern explizit für Testzwecke erstellt werden.
Systemmodell	Ein Systemmodell wird von den Entwicklern im Rahmen konstruktiver Aktivitäten (wie z. B. dem technischen Systementwurf) erstellt.
Umgebungsmodell	Ein Modell des Nutzungskontextes des Systems, z. B. ein operationales Profil oder eine technische Protokollspezifikation.
Plattformunabhängiger Testfall (PIT)	Ein plattformunabhängiger Testfall beschreibt das Testverhalten auf der Abstraktionsebene der Systemfunktionalität, unabhängig von Implementierungsdetails und der eingesetzten Technologie. Ein PIT enthält somit abstrakte Informationen und ist nicht direkt ausführbar.
Plattformspezifischer Testfall (PST)	Ein plattformspezifischer Testfall beschreibt einen Testablauf, der alle relevanten Informationen über die Plattform berücksichtigt, auf der dieser ausgeführt wird. Ein PST ist direkt ausführbar.
Modellüberdeckung	Die Modellüberdeckung beschreibt, zu welchem Grad die Elemente des Testmodells (z. B. Zustände in einem Zustandsdiagramm) durch die während des Testdesigns erstellten Testfälle abgedeckt werden (vgl. [ISTQB]).
Testfallgenerator	Ein Testfallgenerator implementiert die Algorithmen zur automatisierten Erstellung von Testfällen aus Testmodellen unter Berücksichtigung der Kriterien zur Modellüberdeckung.
Testdatengenerator	Ein Testdatengenerator ist ein Testunterstützungswerkzeug, mit dem konkrete Eingabewerte und erwartete Ausgabewerte für die Testfälle generiert, bereitgestellt, verändert oder aus einer Datenbank selektiert werden können.

Tabelle 1: Wichtige Begriffe von MBT.

[1]) *In diesem Artikel können wir aus Platzgründen nicht näher darauf eingehen, wie die Erstellung und Ausführung von Testfällen mittels MBT konkret aussieht. Ein ausführliches Beispiel finden Sie z. B. in [Bra08].*

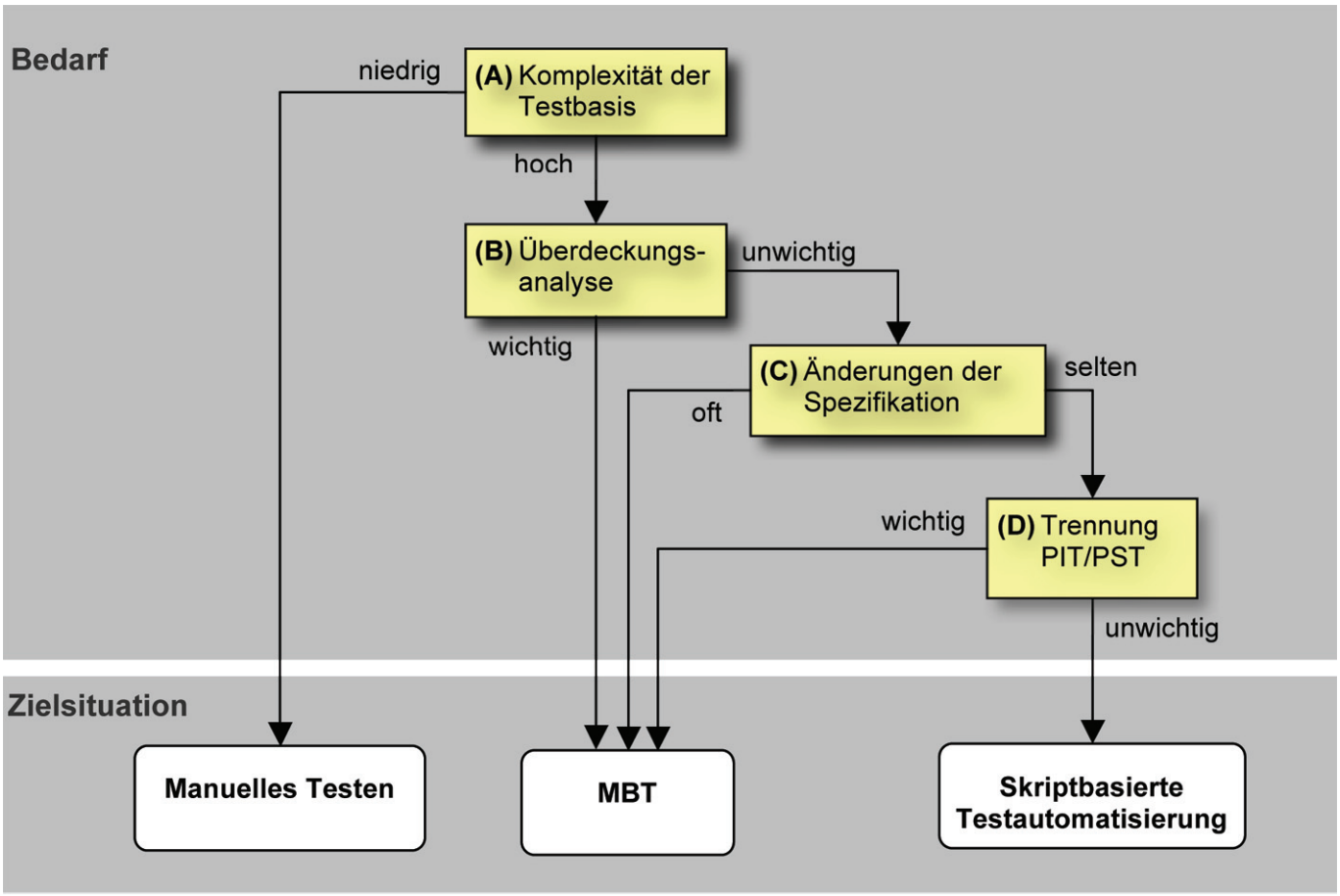

Abb. 1: Entscheidungsdiagramm zur Feststellung des Bedarfs an MBT.

gen in Kundenanforderungen oder in der Systemspezifikation vorkommen, müssen die Testmodelle an die Änderungen angepasst werden.

- Wie bei der skriptbasierten Testautomatisierung auch, erfordert MBT eine höhere Anfangsinvestition zur Beschaffung von Werkzeugen, zur Schulung von Projektbeteiligten und zur Erstellung von Testmodellen.

Der Projekt- bzw. Testmanager steht angesichts dieser Vor- und Nachteile vor der Aufgabe, eine geeignete Technik oder eine Kombination eben solcher auszuwählen. Leider bieten bekannte MBT-Ansätze (vgl. [Net07] [Pre05-a]), MBT-Fallstudien (vgl. [Pre05-b]) sowie MBT-Werkzeuge (vgl. [Göt09]) keine systematische Hilfestellung bei der Beantwortung der Frage, ob MBT im eigenen Testprozess eingesetzt werden kann.

Mit diesem Artikel möchten wir dem Projekt- bzw. Testmanager ein Werkzeug in die Hand geben, das den Prozess zur Entscheidungsfindung mit Hilfe einer klaren Begrifflichkeit und einem Fragenkatalog unterstützt. MBT bringt neue Begriffe in die Welt des Testens. Die Vielfalt und die uneinheitliche Nutzung der Begriffe erschweren allerdings das Verständnis. Um weiter zur Begriffsklärung beizutragen, erläutern wir basierend auf [Utt07], [Pre05-a] und [ISTQB] in Tabelle 1 zentrale Begriffe von MBT. Im Anschluss stellen wir eine heuristische Entscheidungshilfe vor, die auf acht Fragen zu den wesentlichen Einflussfaktoren für die Anwendbarkeit von MBT basiert. So kann man den Bedarf für MBT bewerten und eine Handlungsempfehlung ableiten. Am Ende des Artikels geben wir einen kleinen Überblick der auf dem Markt verfügbaren MBT-Werkzeuge.

ACHT wichtige Fragen zum Einsatz von MBT

Nicht immer sind die Testherausforderungen mit MBT am effizientesten und kostengünstigsten zu lösen. Die folgenden acht Fragen helfen dabei zu entscheiden, ob MBT oder andere Techniken für den eigenen Projektkontext geeignet sind. Die zentralen Faktoren bei dieser Entscheidung sind der *Bedarf* im Testprozess und die *Fähigkeiten* des Testteams. Die entsprechenden Fragen sind als Entscheidungsknoten jeweils in einem Diagramm (siehe Abb. 1) eingeordnet, sodass Antworten auf bestimmte Fragen weitere Fragen aus- bzw. einschließen.

Fragen zur Feststellung des Bedarfs

Das Entscheidungsdiagramm zur Feststellung des Bedarfs führt zu drei möglichen Testtechniken. Das jeweilige Resultat soll dabei nicht als alleingültige und die anderen Techniken ausschließende Empfehlung verstanden werden. In der Praxis stellt üblicherweise eine Kombination verschiedener Verfahren die beste Lösung dar. Die Empfehlung des Entscheidungsdiagramms legt hierbei lediglich einen besonderen Fokus auf die Eignung oder Nichteignung von MBT.

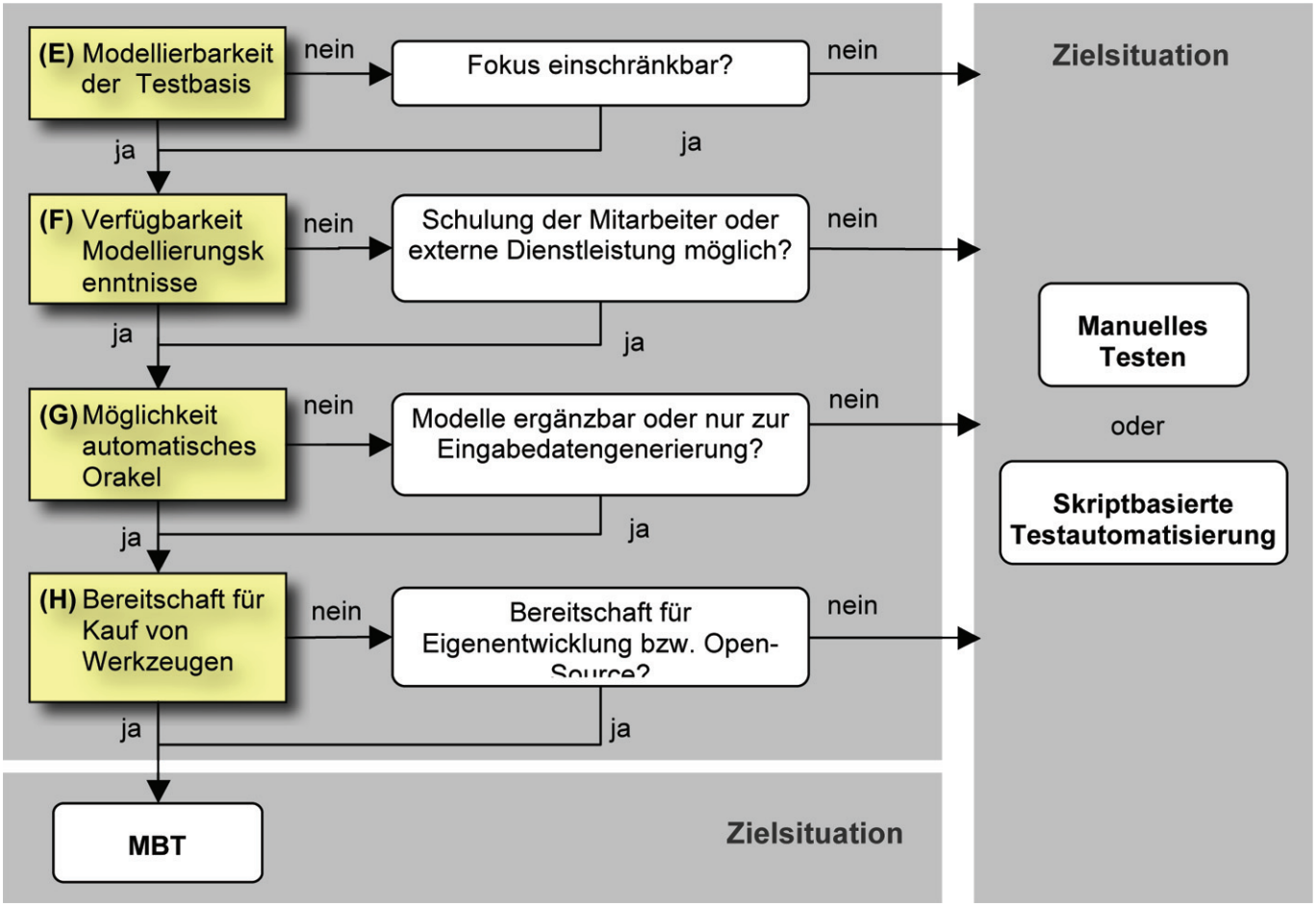

Abb. 2: Entscheidungsbaum zur Feststellung der Fähigkeiten für MBT.

1 Wie komplex sind die Testbasis und die zu testende Funktionalität?

Ist der Umfang der zu testenden Funktionalität in der Testbasis (d. h. allen Dokumenten, aus denen die Anforderungen ersichtlich werden) niedrig, kann die daraus resultierende überschaubare Menge von Testfällen manuell erstellt und ausgeführt werden. Ist die zu testende Funktionalität hingegen umfangreich oder zu komplex für manuelles Testen, können automatisierte Techniken eingesetzt werden. In diesem Fall kann MBT die Erstellung der Testfälle und eine skriptbasierte Testautomatisierung die Durchführung unterstützen.

2 Wie wichtig ist die Überdeckungsanalyse?

Während des Entwicklungsprozesses wird aus den oft informell beschriebenen Kundenanforderungen eine Spezifikation erstellt, die einen formelleren Charakter hat. Jeder Testfall im Testprozess soll sich auf einen Teil der Spezifikation beziehen. Durch mehrere Testfälle wird eine möglichst hohe Überdeckung der Spezifikation angestrebt. Bei der Testauswertung sollen die Testergebnisse und die entsprechende Überdeckung der Spezifikation an die Kundenanforderungen zurückgeführt werden, um die Erfüllung der Kundenwünsche messen und nachweisen zu können. Der Einsatz von MBT ermöglicht die zielgerichtete Erreichung einer hohen und nachweisbaren Überdeckung mithilfe der automatischen Testfallgenerierung. Die erreichte Überdekkung ist durch die Modellüberdeckung und die Verfolgbarkeit (*Traceability*) zu den Kundenanforderungen systematisch und automatisiert messbar. Dies ist ein entscheidender Vorteil von MBT gegenüber der skriptbasierten Testautomatisierung und dem schlüsselwortbasiertem Verfahren.

3 Wie häufig finden Änderungen in der Spezifikation statt?

Aufgrund von Anforderungsänderungen an ein System kann es zu Änderungen in der Spezifikation und darauffolgend zu Änderungen in der Implementierung kommen. Insbesondere für agile Projekte ist es typisch, dass bei aufeinander folgenden Entwicklungszyklen die funktionalen Anforderungen geändert oder erweitert werden. Dabei wächst der Bedarf an eine Testautomatisierung, die die Regressions- und Integrationstests der alten und neuen Funktionalität unterstützt. Für die skriptbasierte Testautomatisierung stellen die Änderungen in der Implementierung – insbesondere an den Schnittstellen – ein Problem dar, wenn dadurch die Testskripte nutzlos werden und bei jedem Entwicklungszyklus aufwändig angepasst werden müssen. Aus diesem Grund sollte bei häufigen Änderungen in der Spezifikation

der Einsatz von MBT in Betracht gezogen werden. Dabei können die Änderungen in der Spezifikation in den formalen Testmodellen abgebildet und die Testskripte nach jeder Änderung durch das automatisierte Verfahren von MBT ohne erheblichen Aufwand erneut generiert werden. Bei einer stabilen Spezifikation hingegen können Testskripte einmal manuell oder durch automatisierte Aufzeichnung erstellt und wiederholt genutzt werden.

4 Wie wichtig ist die Plattformunabhängigkeit der Testfälle?

Wenn technologische Änderungen (z. B. GUI-Bibliotheken) in den neuen Releases berücksichtigt werden, kann es vorkommen, dass die alten Testskripte nutzlos werden, weil sie sehr technologieabhängig sind. Aus diesem Grund sollte zwischen den fachlichen und technischen Testfällen unterschieden werden. MBT bietet die Unterscheidung zwischen *plattformunabhängigen Testfällen (PIT)* und *plattformspezifischen Testfällen (PST)*. Dadurch kann der Tester sich während des Testdesigns auf die Fachlichkeit konzentrieren. Während der Testrealisierung können aus den PITs wiederum automatisiert entsprechende PSTs abgeleitet werden. Auch wenn schlüsselwortbasierte Testautomatisierung Abstraktionstechniken zu diesem Zweck vorsieht, ist der Vorteil bei MBT, dass in den Testmodellen viel mehr testrelevante Informationen, wie z. B. Testdaten oder Testorakel, eingebettet werden können

Fragen zur Feststellung der Fähigkeiten

Ergibt sich aus dem Entscheidungsdiagramm in Abbildung 1 bzw. durch die obigen Fragen der Bedarf an MBT, sind als nächstes die Fähigkeiten des Projekts zu untersuchen, die für die Einführung von MBT notwendig sind (siehe Abb. 2). Sind diese Fähigkeiten noch nicht vorhanden, müssen verbessernde Maßnahmen eingeleitet werden. Falls keine Bereitschaft für diese Maßnahmen vorhanden ist, empfehlen wir eine Betrachtung von alternativen Techniken.

5 Ist die Testbasi modellierbar?

Um MBT einsetzen zu können, muss man die Testbasis oder Teile davon mit Hilfe einer entsprechenden Notation modellieren können. Hierfür bieten sich unterschiedliche Notationsarten an, wie z. B. zustandsbasierte oder ereignisbasierte Notationen, insbesondere zur Modellierung von funktionalen Eigenschaften. Kommt hingegen den nicht-funktionalen Eigenschaften eine wichtige Bedeutung zu, fehlen häufig geeignete Notationen und Werkzeuge, um diese einfach und gewinnbringend zu modellieren. Kann die Testbasis nicht gewinnbringend in ihrer Gesamtheit modelliert werden, muss man prüfen, inwieweit eine Modellierung für Teilbereiche der Testbasis möglich und sinnvoll ist.

6 Sind Modellierungskenntnisse im Testteam verfügbar?

Testmodelle im Kontext von MBT erheben grundsätzlich höhere Ansprüche an Vollständigkeit und Formalität als solche, die bei manuellen Testverfahren zum Einsatz kommen. Weil – je nach Art der Funktionalität und Anwendungsdomäne – sehr unterschiedliche Modelle verwendet werden können, müssen gegebenenfalls vorhandene Modellierungskenntnisse den eingesetzten Modellen entsprechen. Wenn die Modellierungskenntnisse nicht ausreichen, sollten die Teammitglieder durch entsprechende Schulungsmaßnahmen qualifiziert oder das fehlende Know-how durch externe Testexperten abgedeckt werden. Sind die Kosten für diese Maßnahmen zu hoch, verbleibt manuelles oder skriptbasiertes Testen.

7 Kann das Orakel automatisiert werden?

Im MBT werden die Testfälle aus den Testmodellen, die das mögliche Verhalten der Software für bestimmte Eingaben spezifizieren, automatisch generiert. Beispielsweise können Zustandsänderungen eines Softwaresystems zusammen mit den auslösenden Ereignissen mit Hilfe von Zustandsdiagrammen modelliert werden. Somit besteht die Möglichkeit, die erwarteten Zustände direkt bei der Testfallgenerierung zu berücksichtigen. Dabei gilt das Modell als so genanntes Testorakel.

Falls die erwarteten Systemeigenschaften in den Testmodellen nicht spezifiziert sind, sollten die Modelle um diese Information ergänzt werden. Wenn das Orakel nicht automatisierbar oder nicht mit einem vertretbaren Aufwand ergänzbar ist, müssen die Testergebnisse manuell ausgewertet werden. In diesem Fall kann MBT lediglich zur Generierung der Testeingaben verwendet werden.

8 Ist die Bereitschaft für den Kauf und Einführung eines MBT-Werkzeugs vorhanden?

Auf dem Markt gibt es zahlreiche kommerzielle MBT-Werkzeuge (vgl. [Göt09]). Die Anschaffung solcher Werkzeuge ist meistens mit Kosten verbunden, wie etwa Lizenzkosten und Kosten für die Schulung der Mitarbeiter. Falls keine kommerziellen Werkzeuge erwünscht sind, können Open-Source-Werkzeuge oder eine Eigenentwicklung in Betracht gezogen werden. Allerdings können die Alternativen auch mit hohen Kosten verbunden sein, z. B. Kosten für die Anpassung und die Entwicklung der Werkzeuge. Vor der Anschaffung eines Werkzeugs sollte man daher prüfen, ob dieses sich in die existierende Werkzeugumgebung integrieren lässt. Auch die Skalierbarkeit und Flexibilität des Werkzeugs muss bewertet werden.

Hinweise zur Anwendung des Entscheidungsdiagramms

Das hier vorgestellte Entscheidungsdiagramm kann natürlich nicht alle in der Praxis auftretenden Spezialfälle abdecken. Es beschreibt aber eine Heuristik, die den Entscheidungsprozess sinnvoll unterstützt. Falls das Entscheidungsdiagramm die Nutzung von MBT nahelegt, können die folgenden Maßnahmen die Entscheidung weiter absichern:

- Externe Experten einbeziehen, insbesondere bei der Auswahl der Modellierungsnotation sowie der Algorithmen zur Testfallerstellung.
- Ein Pilotprojekt durchführen und dabei Entscheidungen, Aktivitäten, Problemen und Aufwände dokumentieren.

Erste Erfahrungen aus der Industrie

Zur Validierung haben wir unseren Ansatz in einzelnen Interviews mit Testverantwortlichen der Firmen Axini Testautomation, Capgemini sd&m AG und Wincor-Nixdorf International AG erprobt. Diese Firmen besitzen Testteams oder dedizierte Testabteilungen, die aus 40 bis 60 Mitar-

Testfall-editor	Unterstützter Modelltyp	Hersteller	Web
Rational Tau	UML	IBM	www.ibm.com
Spec Explorer	Spec#, AsmL	Microsoft	research.microsoft.com/specexplorer
TGV	IOLTS	Verimag	www-verimag.imag.fr
TTModeller	UML, UTP	TestingTechnologies	www.testingtech.com
Expecco	UML	Exept	www.exept.de

Tabelle 2: Modellbasierte Testfalleditoren.

beitern bestehen. In allen Interviews hat das Entscheidungsdiagramm geholfen, die Eignung von MBT im Testprozess festzustellen. Die Entscheidung für oder gegen MBT hing stark vom Projektkontext ab, da die Rahmenbedingungen der Projekte sehr unterschiedlich sind. Von den Interviewpartnern wurde bestätigt, dass MBT in einem Projekt meist in Kombination mit anderen Techniken eingesetzt wird. Ein Interviewpartner hatte bereits vor dem Interview Erfahrungen mit MBT und wurde durch die Anwendung des Entscheidungsdiagramms in seiner Entscheidung für MBT bestätigt.

Kommerzielle MBT-Werkzeuge

Auf dem Markt sind bereits zahlreiche MBT-Werkzeuge verfügbar, sodass es für den MBT-Einsteiger nicht einfach ist, das für ihn geeignete Werkzeug auszuwählen. Daher stellen wir hier die wichtigsten zwei Kategorien von MBT-Werkzeugen vor (vgl. auch [Göt09]) und siehe Tabellen 1 und 2 für beispielhafte Werkzeuge).

- *Modellbasierte Testfalleditoren:* Diese Editoren erzeugen – basierend auf einem abstrakten Modell eines Testfalls – ein oder mehrere Testfälle bzw. Testskripte zur manuellen bzw. zur automatischen Testdurchführung.
- *Modellbasierte Testfallgeneratoren:* Diese Generatoren erzeugen – basierend auf einem Modell des Systemverhaltens, der Systemumgebung oder der Tests sowie bestimmter Steuerinformationen – Testfälle, Testskripte bzw. Test-Suiten für konfigurierbare Modellüberdeckungskriterien. Funktionen zur Testdaten-Generierung werden ebenfalls unterstützt. Häufig können die modellbasierten Testfallgeneratoren über eigene Schnittstellen oder *Plug-Ins* an Anforderungsmanagement-Werkzeuge angebunden werden.

Ein erfolgreicher Einsatz modellbasierter Testwerkzeuge hängt von der Reife der Prozess- und Methodenlandschaft der Organisation ab, von einem ausreichenden Bekenntnis des Managements zur Bereitstellung der notwendigen Ressourcen, von definierten Zielen, realistischen Erwartungen und der Bereitschaft, sich auf ein neues Thema einzulassen.

Fazit und Dank

„Ist das modellbasierte Testen für mein Projekt einsetzbar?" Das hier vorgestellte Entscheidungsdiagramm gibt Projekt- und Testmanagern eine Hilfestellung bei der Beantwortung dieser Frage. Das Diagramm stellt natürlich nur eine einfache Heuristik dar, sodass eine Kosten-Nutzen-Analyse sowie das Einbeziehen externer Experten eine weitere sinnvolle Maßnahme zur Absicherung der Entscheidung pro bzw. contra MBT ist.

Die Methoden und Werkzeuge für den Einsatz des modellbasierten Testens erfordern einen stabilen Testprozess und entsprechendes Expertenwissen. Insbesondere die Modellierungskenntnisse der Testdesigner sind bei der Einführung von MBT entscheidend.

Wenn Sie sich für die Nutzung des Entscheidungsdiagramms oder weiterführende Aspekte des modellbasierten Testens interessieren, laden wir Sie herzlich ein, über die Homepage des Arbeitskreises TOOP/MBT (vgl. http://toop.gi-ev.de) Kontakt mit uns aufzunehmen.

Testfall-generator	Unterstützter Modelltyp	Hersteller	Web
TestBench	UML	imbus	www.imbus.de
TestDesigner	UML	SmartTesting	www.smartesting.com
Qtronic	UML + Java/C#	Conformiq	www.conformiq.com
Reactis	Simulink/Stateflow	Reactive Systems	www.reactive-systems.com
Rational Statemate/ Rhapsody	UML, SysML	IBM	www.ibm.com
.getmore	UML	sepp.med	www.seppmed.de
UnITeD	UML	Afra	www.afra.de

Tabelle 3: Modellbasierte Testfallgeneratoren.

Wir bedanken uns sehr herzlich bei Melanie Spät, Hubert Segin, Machiel van der Bijl, Dr. Christof J. Budnik, Rob Kuijt, Thomas Roßner, Prof. Dr. Andreas Spillner, Dr. Armin Metzger, Dr. Eike Hagen Riedemann, Lars Frantzen, Dr. Matthias Hamburg und den Teilnehmern des Arbeitskreises TOOP/MBT der GI für ihre ausführlichen Reviews und Anregungen zu diesem Artikel. ■

Literatur & Links

[Bak08] P. Baker, Z. Ru Dai, J. Grabowski, O. Haugen, I. Schieferdecker, C. Williams, Model-Driven Testing, Springer Verlag, 2008

[Bin99] R. Binder, Testing Object-Oriented Systems: Models, Patterns, and Tools Addison-Wesley, 1999

[Bra08] C. Brandes, A. Ditze, C. Kollee, D. Kreische: Modellbasiertes Testen praxisgerecht realisiert: Vom UML2-Profil zum Testmanagementwerkzeug, in: OBJEKTspektrum, 3/2008

[Fra04] F. Fraikin, M. Hamburg, S. Jungmayr, T. Leonhardt, A. Schönknecht, A. Spillner, M. Winter, Die trügerische Sicherheit des grünen Balkens, in: OBJEKTspektrum, 1/2004

[Göt09] H. Götz, M. Nickolaus, T. Roßner, K. Salomon, Modellbasiertes Testen, in: iX Studie 1/2009

[ISTQB] ISTQB/GTB Standard Glossar der Testbegriffe Ver. 2.0, siehe: www.german-testing-board.info/downloads/pdf/ISTQBGTB_Glossary_EN_ DE_V20.pdf

[Net07] A.D. Neto, R. Subramanyan, M. Vieira, G. Tracassos, A Survey on Model-based Testing Approaches: A Systematic Review, Siemens Corporate Research, 2007

[Pol02] M. Pol, T. Koomen, A. Spillner: Management und Optimierung des Testprozesses, dpunkt.verlag, 2002

[Pre05-a] A. Pretschner, J. Philips, Methodological Issues in Model-Based Testing, in: M. Broy, et.al. (Eds.), Model-Based Testing of Reactive Systems, Springer-Verlag, 2005

[Pre05-b] A. Pretschner, W. Prenninger, S. Wagner, C. Kühnel, M. Baumgartner, B. Sostawa, R. Zölch, T. Stauner, One evaluation of model-based testing and its automation, in: Proc. of the 27th Int. Conf. on Software Engineering, ACM, 2005

[Rob00] H. Robinson, Intelligent Test Automation, Software Testing & Quality Engineering, 2000, siehe: www.geocities.com/model_based_testing/intelligent.pdf

[Spi05] A. Spillner, T. Linz, Basiswissen Softwaretest (3. überarbeitete Auf.), dpunkt.verlag 2005

[Utt07] M. Utting, B. Legeard, Practical Model-Based Testing: A Tools Approach, Morgan Kaufmann, 2007

[Win09] M. Winter, Modellbasierter Test – Alter Wein in neuen Schläuchen? In: Proc. of 10. SQC-Kongress, Düsseldorf, 2009