OTX – Der Einsparungsstandard

Standards werden generell zur Kostenoptimierung und Prozessvereinfachung definiert. Der neue Standard OTX (ISO 13209) ist hierfür ein schönes Beispiel, der deutlich zeigt, wenn Prozesse eindeutig definiert und die richtigen Tools verwendet werden, nehmen Kosten und Abläufe einen optimalen Verlauf. Macht man sich also über Randbedingungen und Einführungsszenarien von Anfang an klare Gedanken, hilft OTX bei der durchgängigen Wiederverwendbarkeit von Informationen und dies in allen Prozessphasen.

Autonomes Fahren, Connected Car, intelligente Vernetzung und E-Mobility sind gegenwärtig die Themenfelder, die in der Fahrzeugentwicklung den Takt bestimmen. Elektronik, Software und die Abstimmung von Kommunikation und Information sind dabei die Eckpfeiler, die Einfluss auf Kosten in der gesamten Prozesskette nehmen.

Schon ein flüchtiger Blick auf den Prozess zeigt, dass die Anzahl der Stakeholder beim Thema „Diagnose" schier unüberschaubar ist. Der Kundenstamm der Diagnose kommt ursprünglich aus den Bereichen Produktion und After-Sales, da Diagnose ein fester Bestandteil der heutigen Fahrzeugproduktion und -reparatur ist. Die Diagnose und der korrekte Einbau von Elektronik- und Softwarekomponenten werden heutzutage mithilfe von Steuergeräten selbstständig durchgeführt.

Auch innerhalb der Entwicklung kommt die Diagnose als Validierungswerkzeug zum Einsatz. Dies geschieht dann, wenn der Zugang zum Steuergerät über alternative Zugänge nicht mehr

Alle Bilder: Softing

© Carl Hanser Verlag, München

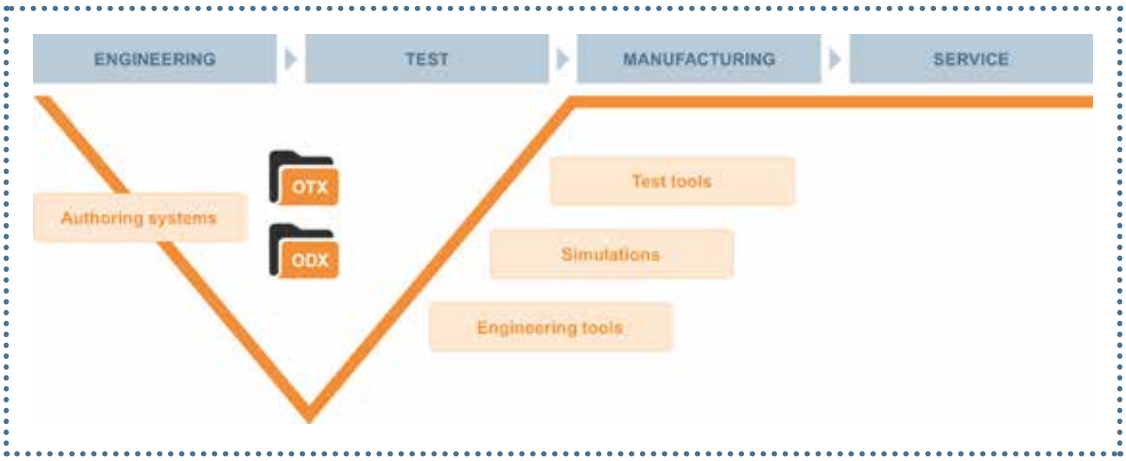

Bild 1: Der gemeinsame Berührungspunkt ist die Diagnosespezifikation: Die Diagnosekommunikation über ODX und die Beschreibung der Diagnoseabläufe über OTX.

möglich ist, sich das Steuergerät also einem Seriensteuergerät nähert und nur über die Diagnose auf Interna (wie z. B. Messgrößen) zugegriffen werden kann. Häufig wird in „Sechs-Augen-Gesprächen" beschlossen, welche Anforderungen für Produktion, After-Sales und Entwicklung zu implementieren sind. Diese Fahrzeuganforderungen werden dann auf Funktionen und Steuergeräte heruntergebrochen.

Die Zuverlässigkeit des Codes und der Kontrollalgorithmen ist extrem wichtig, da man sich in den nachgelagerten Prozessen von Produktion und After-Sales zu 100% darauf verlassen können muss. Ohne die Unterstützung der im Fahrzeug integrierten Diagnose- und Testfunktionen kann der Kfz-Mechatroniker in der Werkstatt Fehlerfälle in den seltensten Fällen diagnostizieren und beheben. Bei der Implementierung der Testfunktionalitäten im Steuergerät geht man heute davon aus, dass je nach Komplexität des Steuergeräts zwischen 30% und 70% des Codes aus Diagnose bestehen.

Darüber hinaus gibt es noch viele weitere Stakeholder für Diagnose. Ein Beispiel sind die Systemhersteller, die vom OEM mit Informationen zur Diagnose versorgt werden und entsprechende Dokumentationen zurückliefern. Aber auch Toolhersteller sind häufig „an Bord", insbesondere wenn sie die Testwerkzeuge vor Ort beim OEM betreiben oder neue Funktionen in den Tools integrieren.

Einsparungsmöglichkeiten – wohin man schaut

Der gemeinsame Berührungspunkt der verschiedenen Beteiligten ist die Diagnosespezifikation. Ein Teil dieser Spezifikation, die Diagnosekommunikation, ist heute über ODX standardisiert und wird in dieser standardisierten Form von zahlreichen OEMs in Deutschland und international verwendet. In der Praxis sind zahlreiche Abläufe zu beachten, die zu spezifizieren, zu implementieren und anschließend zu verwenden sind (Bild 1). Die Spezifikation erfolgt verbal oder graphisch, es kommen Werkzeuge wie Doors oder Winword zum Einsatz. Die Implementierung erfolgt mithilfe von Werkzeugen wie VisualStudio, Python oder LabView. Die Anwendungsbereiche sind ausgesprochen vielseitig, das damit verbundene Einsparungspotenzial entsprechend groß.

Im Bereich Diagnose lassen sich Einsparungen vor allem durch die Wiederverwendung von einmal erstellten Informationen, sowie durch die Prozessautomatisierung erreichen. Bei der Wiederverwendung ist zu beach- »

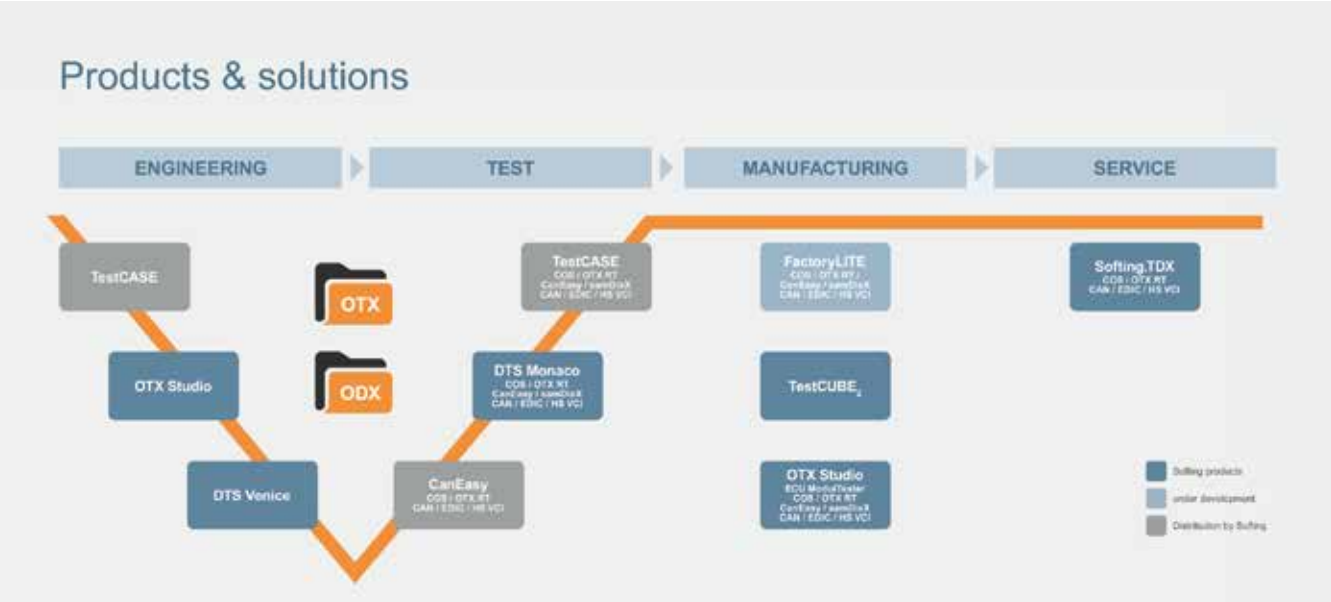

Bild 2: Eine durchgängige Lösung von Softing vom Engineering bis hin zum Service.

ten, dass besonders Diagnosedaten einer ständigen Änderung unterliegen. Je häufiger die Daten verwendet werden, desto höher ist das damit verbundene Einsparpotenzial. Nicht zuletzt können Einsparungen durch die Fehlervermeidung realisiert werden. Insbesondere Missverständnisse, die sich aus der Fehlinterpretation geschriebener Spezifikationen ergeben, summieren sich zu erheblicher Bedeutung auf.

OTX ist als Standard für die Beschreibung von Diagnoseabläufen in allen genannten Einsparungsfeldern von der Spezifikation bis zum Test im Einsatz. Dadurch können Missverständnisse und Parallelaufwände vermieden werden. Schließlich stellt es eine klassische „Single Source" dar, sodass die durchgängige Wiederverwendbarkeit in allen Prozessphasen gegeben ist.

Unterschiedlichste Anwendergruppen

OTX ist sowohl für Diagnoseanwender, als auch für das Erstellen komplexer Diagnoseabläufe geeignet. Die Spanne der Anwendergruppen reicht vom Techniker, der zwar die Diagnosedienste zur Erstellung von Reparaturanleitungen kennt, aber keine weiterführenden Programmierkenntnisse besitzt, bis hin zum Entwickler, für den das Programmieren das tägliche Handwerk ist.

Als Diagnoseanwender bekommt man mit OTX das für den entsprechenden Anwendungsbereich passende Werkzeug an die Hand. Für diese Anwendergruppe spielt es in erster Linie keine Rolle, auf welche Art und Weise die Informationen in das Werkzeug integriert werden und ob dabei ein Standard oder proprietäre Daten als Quelle dienen.

Deutlich komplexer ist die Betrachtung auf Seiten der Erstellsysteme. In der Regel beschreibt OTX zwei unterschiedliche Arten von Abläufen. Einerseits sind das reine Diagnoseabläufe, wie sie in Diagnoselastenheften zu finden sind. Dazu gehören Routinen wie die Flash-Programmierung oder das Auslesen des Fehlerspeichers mit allen Umgebungsbedingungen. Andererseits werden Abläufe beschrieben, die der Geschäftslogik zuzuordnen sind, wie etwa die Routine zur Inbetriebnahme von Steuergeräten.

Anforderungen an Werkzeuge

Aus den verschiedenen Anwendergruppen ergibt sich eine Reihe von Anforderungen an Werkzeuge. Auf Seiten der Testsysteme findet man sehr unter-

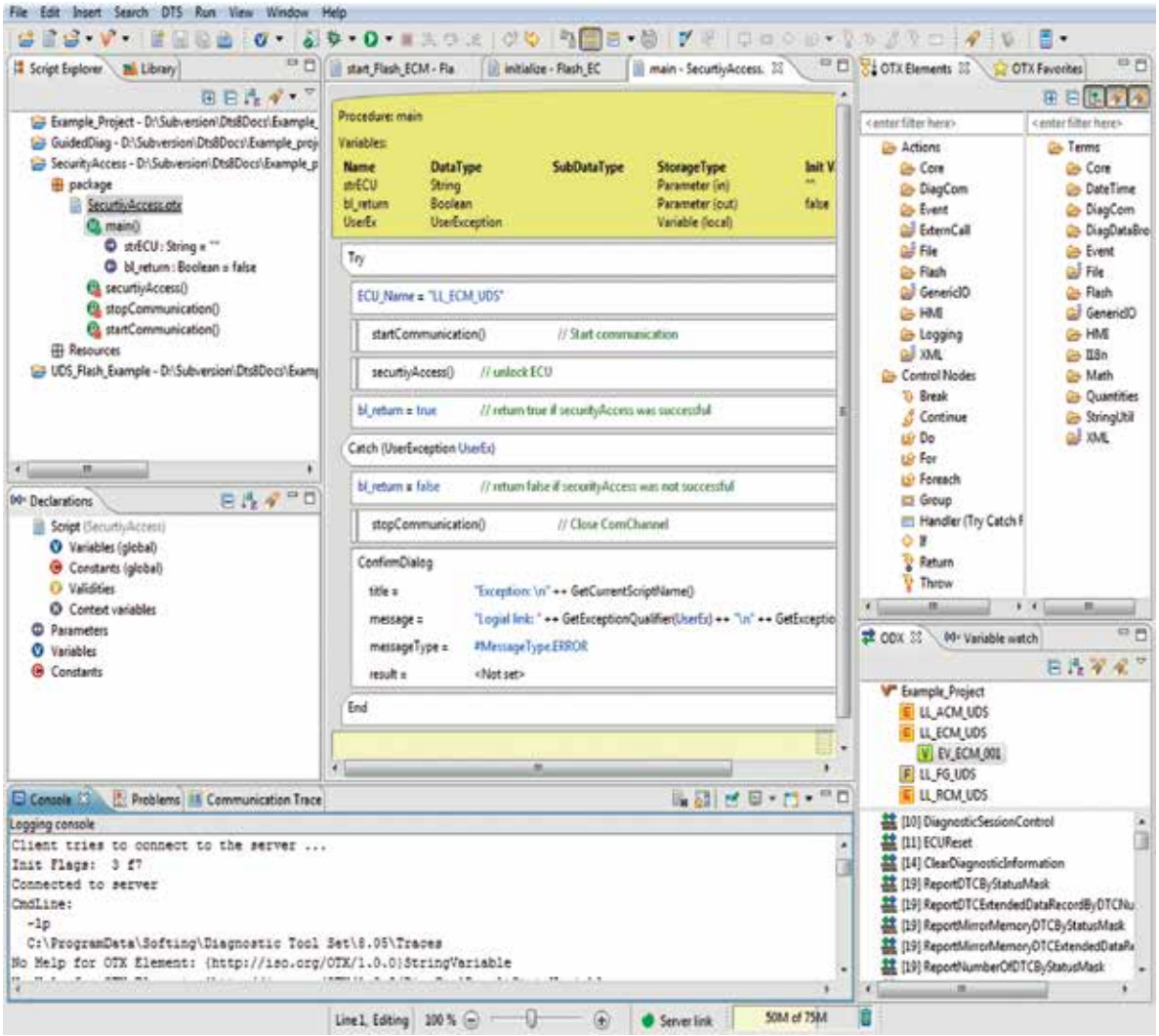

Bild 3: OTX Studio Expertenmodus: Der Programmierer hat umfassenden Zugriff auf alle OTX-Sprachelemente.

© Carl Hanser Verlag, München

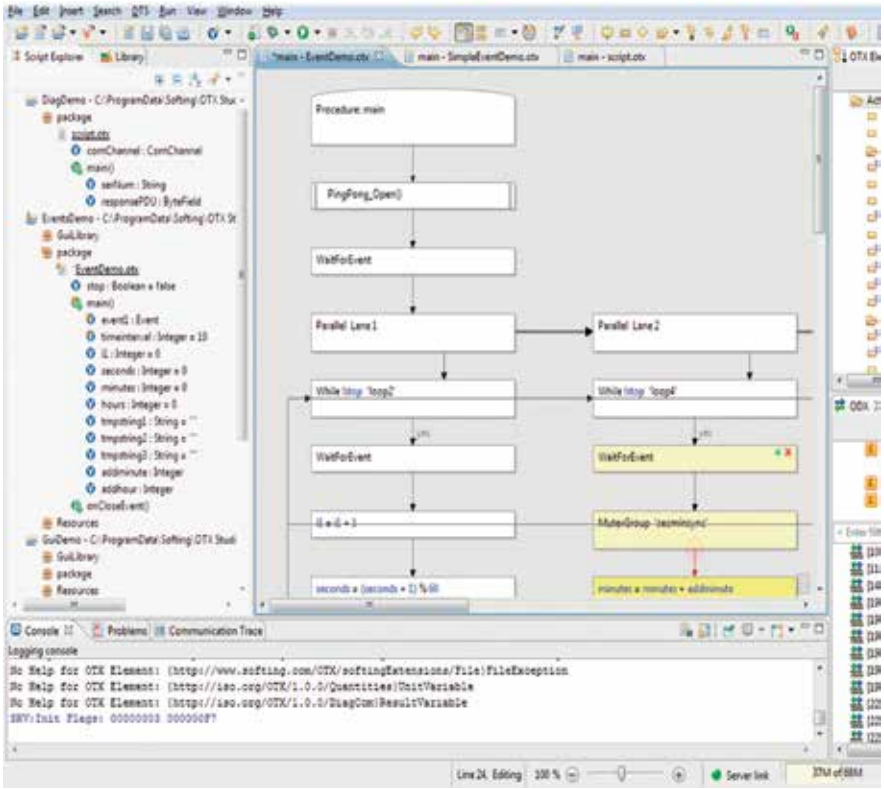

Bild 4: Der Komfortmodus von OTX Studio: Durch reine Konfigurationsdialoge erfolgt die Umsetzung ohne Programmieraufwand oder -kenntnisse.

schiedliche Werkzeuge: Interaktive Werkzeuge, bei denen der Anwender eine große Bandbreite an Einstellmöglichkeiten hat (z. B. Entwicklungstester) oder Testautomatisierungen wie etwa in der Produktion, die gar keine Anwenderinteraktion beinhalten. Alternativ können Testautomatisierungen einzelne automatische Abläufe aufweisen, die einer manuellen Kontrolle bedürfen. Alle genannten Werkzeuge müssen mit den vorgegebenen OTX-Abläufen umgehen können; die Darstellung für den Anwender muss jedoch von der zugrundeliegenden Technologie unabhängig sein.

Auf Seiten der Erstellsysteme stellen sich die Anforderungen komplexer dar: Neben den unterschiedlichen Anwenderklassen spielt die Trennung zwischen Diagnoseabläufen und der Geschäftslogik eine bedeutende Rolle. Im Detail sollte man sich auch mit Freigabeprozessen und deren Arbeitsweisen beschäftigen.

Dazu ein Beispiel aus der Flash-Programmierung: die Aufgabe heißt „Daten an ein Steuergerät senden". Heutige Diagnoseprotokolle sehen dazu je einen Start- und Beendigungsdienst vor, sowie einen weiteren Dienst zur eigentlichen Datenübertragung. Außerdem werden zusätzliche Sicherheitsmechanismen benötigt, die vom Steuergeräteentwickler tendenziell spät integriert werden. Gleiches gilt für Mechanismen, die in der Fahrzeugprogrammierung möglichst viel Bandbreite zur Verfügung stellen. Folglich sollten kleine sequenzielle Abläufe erstellt und in Bibliotheken freigegeben werden.

Erfolgreiche Erstellsysteme müssen den Spagat zwischen effizienter Programmierung und Funktionalität bewältigen. Einerseits wollen die Bibliothekersteller möglichst effizient programmieren. Dafür benötigen sie Programmierumgebung mit komfortablem Editor, Debugger und Administrationswerkzeugen wie Differ, Checker und Versionsverwaltung. Andererseits soll die Programmierung für die Entwicklung der Geschäftslogik keine Rolle mehr spielen. Hier kommt es den Anwendern entgegen, wenn sie nur noch fertige Funktionsblöcke zusammenstellen müssen. Deshalb werden dem Anwender in der Spezifikationsphase die Grundstrukturen vorgegeben, die beispielsweise mit Flussdiagrammen »

allgemein verständlich dargestellt werden können. Im späteren Verlauf spielen Autoren der geführten Fehlersuche eine wichtige Rolle, da sie neben der Geschäftslogik auch die Darstellung, Reporting-Mechanismen und weiterführende Dokumente, wie Reparaturanleitungen oder Explosionszeichnungen, integrieren müssen.

i OTX-Standard

OTX (Open diagnostic Test sequence eXchange; ISO 13209) spezifiziert einen Standard zum Austausch von Diagnoseabläufen. Ziel der Standardisierung war es, eine domänenspezifische Programmiersprache für die Automobilelektronik zu definieren, die über einen langen Zeitraum zu Verfügung steht. Als Ablageformat wurde – ähnlich wie beim zugrundeliegenden Standard ODX – XML gewählt. XML erlaubt eine einfache Validierung, ermöglicht aber auch verschiedene Formen der Darstellung von Abläufen auf eine einfache Weise. Darüber hinaus können auch Administrationswerkzeuge wie Checker, Differ, etc. leicht realisiert werden. OTX besteht grundsätzlich aus zwei Teilen. Der Sprachkern (Core) beschreibt Sprachelemente wie Anweisungen (Schleifen, Verzweigungen, etc.) und Datentypen, sowie spezielle Methoden, die z. B. das Variantenhandling erleichtern. Darüber hinaus gibt es einen Erweiterungsmechanismus. Dieser wird für standardisierte Erweiterungen verwendet (z. B. Diagnoseaufrufe oder String-Verarbeitung), kann aber auch für proprietäre Erweiterungen verwendet werden. Folgende standardisierte Extensions existieren:

- DiagCom: Diagnose-Dienste senden (in der Regel über D-Server),
- Job/Flash: Flash-Programmierung,
- Measure: Datenerfassung ,
- HMI: Darstellungselemente (Anwenderinteraktion),
- DateTime: Systemeigenschaften,
- Logging: Datenaufzeichnung,
- EventHandling: Ereignissteuerung, z. B. über Timer oder Maus,
- I18N: Internationalisierung,
- Quantities: Automatische Größenumrechnung,
- Math: Mathematische Funktionen,
- StringUtil: String-Verarbeitung.

OTX Studio – ein Werkzeug für Alle

Mit dem OTX Studio stellt Softing ein OTX-Werkzeug zur Verfügung, das unterschiedlichste Anwendungsbereiche abdeckt. Es dient sowohl der Erstellung von Bibliotheken als auch der Geschäftslogik von der Spezifikation über das Testfeld bis hin zu Produktion und Service. Dafür stehen zwei schaltbare Modi zur Verfügung: Der Expertenmodus und der Komfortmodus.

Im Expertenmodus (Bild 3) ermöglicht OTX Studio eine State-of-the-art-Programmierung von Diagnoseabläufen. Der Programmierer hat umfassenden Zugriff auf alle OTX-Sprachelemente. Eingaben sind durch die integrierte Vorschlagsfunktion sehr leicht zu bewerkstelligen. Der erstellte Code kann sofort mit dem Debugger überprüft werden, wobei eine Änderung des Programmablaufs während des Debuggens jederzeit möglich ist. Die Darstellung kann zeilenorientiert oder als Flowchart erfolgen. Ein Wechsel zwischen den Ansichten ist jederzeit möglich. Durch den integrierten Differ und die implementierte Versionsverwaltung bleiben auch verschiedene Stände des Codes jederzeit nachvollziehbar.

Der Komfortmodus ermöglicht es dem Diagnoseexperten, Abläufe nach dem Bausteinprinzip zu erstellen (Bild 4). Die Bausteine werden von OTX-Experten in Bibliotheken bereitgestellt. Der Anwender kann das Laufzeitverhalten einstellen oder das Reporting beeinflussen. Der benötigte OTX-Code wird im Hintergrund automatisch erzeugt. Mithilfe des Vorlagenmodus können wiederkehrende Funktionen schnell erstellt werden. Dazu werden Templates erzeugt, die dann vom Diagnoseexperten mit den tatsächlichen Kommunikationsmechanismen verknüpft werden.

Darüber hinaus kann OTX Studio als einfacher Diagnose-Tester verwendet werden. Dies hat mehrere Vorteile: Zum einen wird sichergestellt, dass sich Diagnosedienste wie vorgesehen verhalten. Darüber hinaus ist es möglich, bereits ausgeführte Diagnosedienste in einer Tracedatei zu speichern und auf Knopfdruck in einen OTX-Ablauf umzuwandeln. Ergebnissollwerte können problemlos in der Konfiguration ergänzt werden. Dadurch kann in der Steuergeräteentwicklung der Entwicklertest – parallel zur Entwicklung – als Regressionstest wachsen und mit jedem Programmstand ohne zusätzlichen Aufwand ausgeführt werden.

Standard + Tool = Einsparung

Heutige „Tool-Zoos" erfordern einen hohen Aufwand für Anschaffung, Schulung und Einarbeitung. Dieser Aufwand erweist sich als entscheidender Kostenfaktor, der zusätzlich prozessseitig steigt, wenn redundante Vorgänge an verschiedenen Stellen wiederholt umgesetzt werden. Daher gilt es bei einer Kosten-/Nutzenbetrachtung auch die Zeit für notwendige Änderungen (inklusive Korrekturschleifen) und die Beseitigung von Missverständnissen zu berücksichtigen. OTX ermöglicht eine von Anfang durchgängige Methodik, bei der die Spezifikation in Testabläufe umgesetzt wird. Mit der Wahl des richtigen Werkzeugs, das zusätzlich unterschiedliche Anwendersichten unterstützt, erfolgt die Implementierung von OTXs verlustfrei und mit geringstem Lernaufwand. Den entsprechenden Investitionsschutz stellt die standardisierte Basis kostenlos bereit, sodass Kosten zielgerichtet gesenkt und Prozesse optimiert umgesetzt werden. ■ (oe)

» **www.softing.com**

Markus Steffelbauer ist Director Product Management bei Softing.

© Carl Hanser Verlag, München