»Mittels Analysetechniken eine anforderungsbasierte Testfallgenerierung unterstützen«

Das Testen entwickelt sich zukünftig immer weiter in Richtung einer durchgängig automatisierten Verbindung von unter anderem Anforderungen und Testfällen. Eines unserer Ziele ist es, eine vollständigere Generierung von Testfällen zu ermöglichen, um damit eine Kostenersparnis in der System- und Softwareentwicklung zu erreichen. Der Artikel zeigt, wie Sie mithilfe von Analysetechniken hochwertige Anforderungen an den richtigen Betrachtungsgegenstand formulieren, um dadurch die Grundlage für die Automatisierung zu schaffen.

Automatisierung von unterschiedlichen Schritten innerhalb der Systementwicklung ist längst keine Zukunftsmusik! In der modernen Softwareentwicklung werden mithilfe von Continuous Integration und Continuous Delivery (CI/CD) Pipelines genutzt, die den Entwicklern vor allem Tätigkeiten zur Qualitätssicherung, zur Integration und für das Deployment abnehmen und damit die Durchlaufzeiten von Änderungen durch die Entwicklung stark verkürzen. Doch was dient eigentlich als Grundlage für die Qualitätssicherung der entwickelten Software?

Qualitätscheck bereits vor dem Test

Häufig müssen Entwickler Testfälle für deren Software per Hand aus den Anforderungen erzeugen und der CI/CD-Pipeline zur Verfügung stellen. Um auch an dieser Stelle kostbare Zeit zu sparen, liegt es nahe, die Testfallerstellung ebenfalls zu automatisieren oder zumindest teilweise zu automatisieren, um so einen weiteren Schritt in Richtung Zukunft zu gehen. Denkbar ist dabei ein ähnliches Vorgehen wie bei der modellbasierten Softwareentwicklung, bei der mithilfe von Generatoren „das Gerüst" der Software erzeugt wird und der Entwickler die Algorithmik in dieses Gerüst einfügt. Übertragen auf die Testfallerstellung bedeutet das, dass auf Basis der Anforderungen das Gerüst der Testfälle generiert wird.

Voraussetzungen für eine derartige Generierung ist, dass die Anforderungen in einer (semi-)formalen Form, das heißt in einer definierten Syntax zum Beispiel in Form von Anforderungsmodellen oder textuellen Anforderungen nach Satzschablone, vorliegen, sodass Generatoren diese Anforderungen verarbeiten können.

Neben der Syntax spielt auch die Semantik der Anforderungen eine wichtige Rolle für die Generierung der Testfälle, denn Sie wollen Ihr System ja sicher gegen das testen, was durch die Anforderungen gefordert wird. Aber wie verbessern Sie die Semantik Ihrer Anforderungen? Das heißt, wie können wir eine Basis schaffen, zukünftig die Testfallerstellung zu (teil-)automatisieren?

Durch den gezielten Einsatz einiger definierter Analysetechniken während der Anforderungsanalyse im Zusammenspiel mit der

Architekturentwicklung schaffen Sie die Grundlagen für die Erstellung der Testfälle. Die Analysetechniken fokussieren besonders die Semantik der von Ihnen formulierten Anforderungen und dienen als zusätzliches Hilfsmittel neben Satzschablonen, modellbasierter Dokumentation, dem Vorgehen vom Use-Case zum Test-Case und Glossaren.

Wie diese Analysetechniken funktionieren und welche Rolle die Betrachtung der Architektur beim Formulieren von Anforderungen spielt, können Sie in diesem Artikel lesen.

Die Wichtigkeit der Kontextabgrenzung

Einer der ersten Schritte in der Analyse der Anforderungen ist die Abgrenzung des Kontextes des zu entwickelnden Systems (im Folgenden auch Betrachtungsgegenstand genannt). Neben der Sammlung verschiedener Anforderungsquellen wie zum Beispiel Gesetzestexte, Normen und Standards untersuchen Sie, wie sich der Betrachtungsgegenstand später im Betrieb in dessen Umgebung eingliedern soll. Außerdem definieren Sie, welche Aufgaben durch das neue System abgedeckt und welche durch Nachbarsysteme erledigt werden sollen. Aus dieser Aufteilung ergeben sich bereits die ersten Schnittstellen des zu entwickelnden Systems. Bedenken Sie dabei, dass sich die Blackbox-Tests Ihres Systems und die spätere Abnahme genau auf diese Ebene beziehen. Herausforderungen bei der Festlegung des Kontextes liegen besonders in der immer stärkeren Vernetzung der Systeme oder auch der Dynamik innerhalb des Systemkontextes bei mobilen Systemen.

Diese Tätigkeiten kommen Ihnen vermutlich aus der Entwicklung einer Architektur für ein System bekannt vor, bei der Sie das Ganze in dessen Bestandteile zerlegen, diesen Bestandteilen Aufgaben zur Realisierung der Anforderungen zuweisen und sich überlegen, welche Schnittstellen aus Ihrer Zerlegung entstehen. Im Prinzip definieren Sie bei der Kontextabgrenzung Ihres zu entwickelnden Systems eine Architekturebene, die oberhalb Ihres Betrachtungsgegenstandes liegt.

Ein kleines Beispiel zur Verdeutlichung

Als Unternehmen wollen wir Smarthome-Systeme am Markt anbieten, liefern dazu aber nur den Teil des Hauses, der „smart" ist. Genauer gesagt liefern wir Funktionen, die ein normales Haus smart erscheinen lassen. Das Portfolio reicht von einer intelligenten Türklingel bis hin zu automatisierter Klima- und Lichtsteuerung in einzelnen Räumen.

Im Rahmen der Kontextabgrenzung des Smarthome-Systems müssen wir nun festlegen, wie sich unser Betrachtungsgegenstand zusammen mit dem „nicht-intelligenten" Haus zu einem „intelligenten" Haus verbinden lässt – Welche Aufgaben muss das Smarthome-System bereitstellen, sodass Hausbewohner ein „intelligentes" Haus erhalten. Dabei machen wir eigentlich Architekturentwicklung (im Sinne der Systementwicklung) und definieren die Systemarchitektur des „intelligenten" Hauses. Welche Aufgaben der nicht-intelligente Teil des Hauses erfüllen kann, steht allerdings schon fest. Nun gilt es, den intelligenten Teil, das Smarthome-System, mithilfe von Anforderungen so präzise wie möglich zu beschreiben und dabei die beiden folgenden Ziele einzuhalten:

- Nur Anforderungen stellen, die der Betrachtungsgegenstand auch eigenständig erfüllen kann.
- Anforderungen möglichst als Blackbox-Anforderungen formulieren, um keine technischen Lösungen zu den Anforderungen vorwegzunehmen.

Auswirkungen beim Generieren von Testfällen bei unklarem Betrachtungsgegenstand

Betrachten wir nun die beiden formulierten Ziele etwas genauer, besonders welche Konsequenzen eine Missachtung dieser Ziele bei der Testfallableitung mit sich bringen. Das erste Ziel adressiert die Machbarkeit als ein Qualitätskriterium von Anforderungen nach IEEE29148:2018 [IEEE18]. Falls Sie mehr fordern, als Ihr zu entwickelndes System im Stande ist zu leisten, erhalten wir bei einer automatisierten Testfallgenerierung nicht zum System passende Testfälle. Sie müssen Testfälle inhaltlich umdefinieren, sodass diese die vom Betrachtungsgegenstand realisierten Aufgaben abdecken.

Soll das Smarthome-System die Eingangstür wirklich öffnen, wie in **Kasten 1** gefordert? In der technischen Realisierung sind zum Öffnen einer Tür gewisse Aktuatoren wie zum Beispiel Motoren erforderlich. Da das Smarthome-System nur den „intelligenten" Anteil zum smarten Haus beiträgt, sind Motoren nicht Teil unseres Systems. Eigentlich wollten wir mit unserer Anforderung ausdrücken, dass der Betrachtungsgegenstand die Tür lediglich entriegeln soll. Sicher können Sie sich bereits die Auswirkungen auf die Testfälle vorstellen, das Öffnen der Tür unterscheidet sich maßgeblich vom Entriegeln der Tür **(Kasten 2)**.

Auswirkungen bei der Testfallgenerierung mit unsauberen Blackbox-Anforderungen

Widmen wir uns nun dem zweiten Ziel. Welche Konsequenzen entstehen, wenn Sie keine Blackbox-Anforderungen formulieren. Blackbox bedeutet, dass Sie Ihr System nur von außen betrachten. Für die Spezifikation der Anforderungen heißt das, dass ausschließlich von außen wahrnehmbare Eigenschaften in Form von nicht-funktionalen Anforderungen oder ein nach außen wahrnehmbares Verhalten durch funktionale Anforderungen erfasst werden dürfen. Nur diese Anforderungen können Sie mit Ihrem System als Testgegenstand nach der Entwicklung überprüfen. Sollten Sie Anforderungen erheben, die das Innere der Blackbox adressieren, können diese Grey- oder

Anforderung:
„Falls das Smarthome-System eine Person als autorisierte Personen erkennt, muss das Smarthome-System die Eingangstür automatisch öffnen."

Testfall:
- Vorbedingung: *„Eingangstür ist verschlossen."*
- Ereignis: *„Das Smarthome-System erkennt eine Person als autorisiert."*
- Erwartetes Ergebnis: *„Die Eingangstür wurde geöffnet."*

Kasten 1: Smarthome-Beispiel

Anforderung:
„Falls das Smarthome-System eine Person als autorisierte Personen erkennt, muss das Smarthome-System die Eingangstür entriegeln

Testfall:
- Vorbedingung: *„Eingangstür ist verriegelt."*
- Ereignis: *„Das Smarthome-System erkennt eine Person als autorisiert."*
- Erwartetes Ergebnis: *„Die Eingangstür wurde entriegelt."*

Kasten 2: Die präzisierte Anforderung: entriegeln statt öffnen

Anforderung:
„Solange sich das Smarthome-System im Zustand ‚überwacht' befindet, muss das Smarthome-System zyklisch im Abstand von 50 ms den Status der Fenster prüfen."

Testfall:
- Vorbedingung: *„Das Smarthome-System ist nicht im Zustand ‚überwacht'."*
- Ereignis: *„Das Smarthome-System wechselt in den Zustand ‚überwacht'."*
- Erwartetes Ergebnis: *„Das Smarthome-System prüft zyklisch im Abstand von 50 ms den Status der Fenster."*

Kasten 3: Beispiel für Fensterüberwachung

Anforderung: *„Solange sich das Smarthome-System im Zustand ‚überwacht' befindet, muss das Smarthome-System zyklisch im Abstand von 50 ms den Status der Fenster prüfen."*

Anforderung: *„Solange sich das Smarthome-System im Zustand ‚überwacht' befindet und falls sich ein Fensterstatus zu ‚geöffnet' ändert, muss das Smarthome-System einen Sicherheitsalarm auslösen."*

Testfall:
- Vorbedingung: *„Das Smarthome-System ist im Zustand ‚überwacht'".*
- Ereignis: *„Der Fensterstatus ändert sich in ‚geöffnet'."*
- Erwartetes Ergebnis: *„Das Smarthome-System löst den Sicherheitsalarm aus."*

Kasten 4: Verbund zweier Anforderungen

Whitebox-Anforderung nur in Verbindung mit einem anderen Testgegenstand aus der Whitebox automatisiert getestet werden.

Bedenken Sie dabei, dass die vollständige innere Struktur eines Systems erst im Rahmen der Architekturentwicklung entsteht und dass erst dort die Allokation der notwendigen Aufgaben zur Realisierung der Systemanforderungen zu Systembestandteilen erfolgt. Erst mit dieser Tätigkeit ergibt sich der Testgegenstand für das Testen der Whitebox-Anforderungen an das System. Als Konsequenz für die automatisierte Erzeugung von Testfällen aus Ihren Anforderungen ergibt sich, dass Sie neben der zu testenden Anforderung auch Systembestandteile aus der Architektur benötigen. Außerdem muss die Anforderung genau durch einen Bestandteil realisiert werden, da sonst kein Testgegenstand automatisiert identifiziert werden kann. Betrachten wir auch hierfür ein Beispiel, siehe **Kasten 3**.

Im ersten Moment mögen diese Anforderung und der zugehörige Testfall als sinnvoll und für den Betrachtungsgegenstand korrekt formuliert erscheinen. Doch wie kann diese Anforderung mit dem Smarthome-System als Testgegenstand getestet werden? Richtig. Diese Anforderung können Sie lediglich im Verbund mit anderen Anforderungen auf der Ebene des Smarthome-Systems testen, die ein von außen wahrnehmbares Verhalten erzeugen, oder Sie kennen aus der Architektur den Systembestandteil, der diese Anforderung eigenständig realisieren kann, und nutzen dieses als Testgegenstand.

Um die oben formulierte Anforderung zu testen, haben Sie zwei Optionen:

- Sie überlegen sich, in welchem Verbund die oben genannte Anforderung getestet werden kann.
- Sie betrachten die Architektur und nutzen den Bestandteil des Smarthome-Systems, der diese Anforderung realisiert.

Für einen Test verwenden wir den Verbund der in **Kasten 4** angegebenen beiden Anforderungen.

Damit ist eine weitere Anforderung entstanden, die ein von außen wahrnehmbares Verhalten anfordert und die mit dem Smarthome-System als Testgegenstand getestet werden kann. Die oben aufgeführte Anforderung zur Überprüfung der Fensterstatus lässt sich im Verbund mit der neuen Anforderung testen.

Für die zweite Option wird die Architektur des Smarthome-Systems benötigt. Genauer gesagt brauchen wir den Systembestandteil, der die Überprüfung der Fensterstatus realisieren soll. Im Beispiel werden dafür Sensoren in den Fensterrahmen verbaut, die den Status des jeweiligen Fensters der Steuerung des Smarthome-Systems bereitstellen. Mit den Kenntnissen der gewählten Architektur lassen sich diese Sensoren als Testgegenstand für die oben formulierte Anforderung identifizieren – bei einer automatisierten Testfallgenerierung jedoch nicht.

Die Qualität von Einzelanforderungen

In der bisherigen Betrachtung haben wir die einzelne Anforderung zum Großteil vernachlässigt und die Konsequenzen des richtigen oder falschen Betrachtungsgegenstandes erläutert. Zusätzlich hat die Qualität der formulierten Anforderung direkte Auswirkungen auf die Testspezifikation. Beginnen wir mit einem Wunschkonzert – oder was sind aus Testsicht unsere Anforderungen an die Anforderungen?

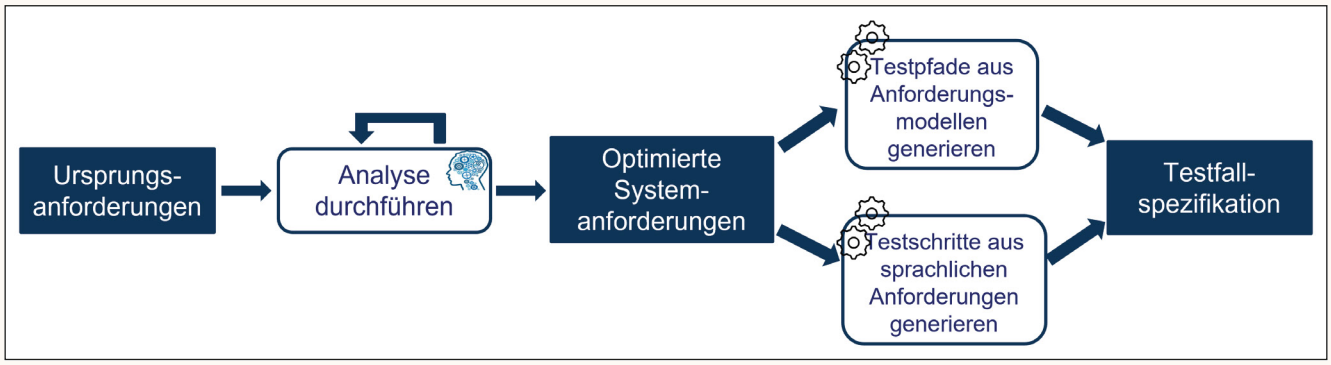

Abb. 1: Zusammenhang Analysetätigkeiten und Testfallgenerierung

Wir erwarten eine vollständige Beschreibung der funktionalen und nicht-funktionalen Aspekte des Systems oder Produkts gegebenenfalls inkl. weiterer Lieferbestandteile. Dies, weil wir möglichst flächendeckend testen möchten, ohne wichtige Systembestandteile unberücksichtigt zu lassen.

Die Anforderungen sollen auch derart detailliert sein, dass alle Vorbedingungen, Testereignisse und erwarteten Ergebnisse für die Testfallgenerierung in einer ableitbaren Form vorliegen. Für Blackbox-Tests möchten wir möglichst keine technischen Lösungen in den Anforderungen sehen, da wir diese in der Blackbox-Betrachtung nicht erkennen werden.

Dann sollten Anforderungen auch in „handhabbaren Häppchen" erscheinen, sodass wir eine brauchbare Testbewertung und statistische Auswertung durchführen können. Soweit zu den grundlegenden Aspekten. Aber wie kommen wir nun dahin?

Die Methode für den Weg zu testbaren Anforderungen

Um all diese Erwartungen an Anforderungen zu erfüllen, benötigt ein Requirements-Engineer mehr als nur einfach Schreibfertigkeiten. Auf Basis langjähriger Reviews von Anforderungsspezifikationen wurden Erkenntnisse für hochwertige Anforderungen gewonnen. Feedback bei der abteilungsübergreifenden Methoden-Einführung zwischen Anforderungs- und Testabteilungen hat zudem gezeigt, unter welchen Bedingungen diese Parteien reibungsfreier zusammenarbeiten **(siehe Abbildung 1)**. Daraus entstand unter anderem ein Set aus Analysetätigkeiten, das von jedem Requirements-Engineer und seitens Testern/Testfallerstellern verwendet werden kann [Rupp21].

Als ein Beispiel aus dem Fundus der Analysetätigkeiten soll eine näher vorgestellt werden:

Tätigkeit: Anforderungen separieren

Schreiben Sie eine separierte Anforderung oder einen separierten Anforderungssatz für:

- jede Funktionalität des Betrachtungsgegenstands,
- jede Qualitätsanforderung.

Das verfolgte Ziel dieser Analysetätigkeit ist es, verschwommene Anforderungen aus unübersichtlichem Fließtext in strukturierte und damit unter anderem testbarere „Häppchen" zu separieren. Um dies zu erreichen, wenden Sie folgende drei Schritte im Rahmen der Analysetätigkeit „Anforderungen separieren" an:

- 1) Einzelne Blackbox-Anforderungen identifizieren.
- 2) Untersuchung der Bedingungen von Funktionen.
- 3) Übergreifende Qualitätsanforderungen identifizieren.

Anschauungsbeispiel ist folgende Anforderung an ein Smarthome-System, das den Zugang zu einem Haus steuert:

„Das Smarthome-System muss die Erlaubnis zum Öffnen der Tür überprüfen und innerhalb von 2 Sekunden die Tür entriegeln."

1) Einzelne Blackbox-Anforderungen identifizieren

- Fragestellung: Können Sie die Funktion beziehungsweise Teile davon für den Betrachtungsgegenstand von außen (d. h. Blackbox-Sicht auf Basis der in der Architektur definierten Abgrenzung) einzeln wahrnehmen/erkennen?
- Aktivität: Die obige Anforderung enthält zwei unterschiedliche funktionale Anteile: Überprüfen der Erlaubnis und Entriegeln der Tür. Diese beiden funktionalen Anteile spezifizieren Sie nun als je eigenständige Anforderungen.
- Auswirkung: Durch die Aufteilung von Funktionen als eigenständige Anforderungen erhalten Sie eine klar strukturierte Basis von einzeln im automatisierten Testdurchlauf bewertbaren Anforderungen.

2) Untersuchung der Bedingungen von Funktionen

- Fragestellung: Sind in einem Anforderungssatz mehrere einzelne Funktionen enthalten und falls ja, gelten für diese dieselben Bedingungen?
- Aktivität: Untersuchen Sie, ob in unserer Anforderung für das Prüfen und für das Entriegeln unterschiedliche Bedingungen gelten. Das Entriegeln erfolgt in zeitlicher Abhängigkeit nach dem Prüfen und nur, falls die Erlaubnis bestätigt ist. Damit ist eine implizite Bedingung identifiziert, die formuliert werden muss. Dies geschieht aus Gründen der Übersichtlichkeit am besten mindestens als eigener Anforderungssatz oder als separate Einzel-Anforderung.
- Auswirkung: Wenn Sie exakt zum Anforderungsinhalt passende Bedingungen beschreiben, werden Testvorbedingungen oder Testereignisse zielgerichtet generierbar.

Referenzen

- [IEEE18] IEEE Standards Board: IEEE Std 29148-2018. IEEE Systems and Software Engineering - Life Cycle Process - Requirements Engineering, 1. Auflage, 2018
- [Rupp21] C. Rupp u. a., Requirements-Engineering und -Management, Carl Hanser Verlag, 2021
- [Sophist] https://www.sophist.de/publikationen/wissen-for-free/performing-the-analysis-of-requirements/

3) Übergreifende Qualitätsanforderungen identifizieren

- Fragestellung: Sind in einem Anforderungssatz Qualitätsaspekte enthalten und falls ja, gelten diese für mehrere Anforderungen übergreifend?

- Aktivität: Prüfen Sie jetzt, ob der Performanzaspekt in der obigen Anforderung („innerhalb von zwei Sekunden") auch für andere Funktionen gilt. Ist dies der Fall, dann spezifizieren Sie eine übergreifende Anforderung an die Performanz. Ansonsten belassen Sie die Qualitätsanforderung in ihrer Anforderungsstruktur so nahe wie möglich bei der funktionalen Anforderung.

- Auswirkung: Die leider oft „stiefmütterlich" behandelten nicht-funktionalen Anforderungen – hier im Beispiel eine Qualitätsanforderung – werden mit diesem Vorgehen ergänzt und im Hinblick auf verschiedenste Funktionen hinterfragt und spezifiziert. Wir erreichen dadurch eine vollständige Testfallgenerierung.

Auf die Frage nach dem richtigen Betrachtungsgegenstand sind wir zu Beginn des Artikels bereits eingegangen. An dieser Stelle haben wir noch einen Tipp: Unsere Erfahrung im Bereich Systems-Engineering – vor allem bei technisch komplexen Systemen – hat gezeigt, dass oft eine verwirrende Vermischung von Anforderungen und technischer Lösung zu sehen ist. Dies erschwert es, sich beim Testen auf einer Teststufe zu halten oder die Anforderungen, mit zusätzlichem administrativem Aufwand, in eine tiefere Teststufe zu verschieben. Bedenken Sie dabei, dass sich die Teststufen nach der gewählten Architektur Ihres Systems richten. Nur so passt eine anforderungsbasierte Testfallgenerierung auch zu den Testgegenständen.

Haben Sie den richtigen Betrachtungsgegenstand für die Anforderungen identifiziert, fällt es Ihnen zudem leichter, die Analysetätigkeiten anzuwenden. Mithilfe dieser Tätigkeiten können die Anforderungen entsprechend zur Teststufe justiert werden. Sie sind zugeschnitten auf den aus der Architektur abgeleiteten Betrachtungsgegenstand und zielgerichtet verwertbar für das Black-box-Testfallgenerieren.

Zusammenfassung

Der Artikel hat aufgezeigt, wie Sie mit einem geringen Aufwand und den richtigen Fragen während der Analyse Ihrer Anforderungen entscheidend die Qualität der Anforderungen erhöhen können, um damit die Testfallerstellung/-generierung und das Testen Ihres Systems zu erleichtern. Wir erreichen damit, auch die Testfallgenerierung zukunftsfähig auszugestalten und eine weitere Grundlage zur Automatisierung von Entwicklungsprozessen zu schaffen. Neben der Frage nach dem richtigen Betrachtungsgegenstand in den Anforderungen haben wir Ihnen eine Analysetätigkeit anhand eines Beispiels näher erläutert. Eine vollständige Übersicht über die Analysetätigkeiten erhalten Sie in [Rupp21].

In Ihrem konkreten Entwicklungsprojekt können die Fragestellungen an unterschiedlichen Stellen zum Einsatz kommen. Eine Möglichkeit ist deren Integration aus den Analysetätigen in Checklisten für gute Anforderungen. Mit dieser Hilfestellung unterstützen Sie diejenigen Personen, die die Anforderungen an Ihr zu entwickelndes System stellen, bereits während der Formulierung der Anforderungen. Eine weitere Möglichkeit ist die Einbindung der Tester in die Anforderungsreviews, die gezielt aus den Analysetätigkeiten abgeleitete Fragen stellen können.

Versuchen Sie doch mal, die oben genannte Analysetätigkeit als Qualitätscheck in der Praxis anzuwenden, und sorgen Sie dafür, dass die Qualitätssicherung und hohe Qualität bereits bei den Anforderungen beginnen. Einen Spickzettel zu den Analysetätigkeiten haben wir Ihnen unter [Sophist] zur Verfügung gestellt.

Andreas Günther

andreas.guenther@sophist.de
ist Berater und Trainer bei SOPHIST und spezialisiert auf sprachliche und modellbasierte Methoden im Requirements-Engineering. Sein Aufgabengebiet umfasst Methoden der linguistischen Analyse sowie der Anforderungsmodelle. Darüber hinaus ist er als Business Analyst in der Geschäftsprozessanalyse tätig. Andreas Günther arbeitet und berät mit allen Vorgehensmodellen, vom Wasserfall angefangen bis hin zu agilem Vorgehen wie zum Beispiel Scrum. Weiterhin beschäftigt er sich mit der Erstellung von Abnahmekriterien/Testfällen im Rahmen des Requirements-Engineerings.

Alexander Rauh

alexander.rauh@sophist.de
unterstützt als Berater und Trainer bei SOPHIST die Kunden bei der Verbesserung von Entwicklungsprozessen und -methoden mit den Schwerpunkten Requirements-Engineering und Architekturentwicklung und -dokumentation. Spezialisiert ist er auf die Entwicklung von komplexen technischen Systemen. Durch seine Kenntnisse und Fähigkeiten in der Anforderungsanalyse und der Modellierung von Anforderungen und Architekturen mittels UML/SysML erarbeitet er Lösungen für unterschiedliche Probleme und vermittelt sein Wissen durch Coaching an Anwender der Prozesse und Methoden. Neben seiner täglichen Arbeit beschäftigt sich Alexander Rauh im Rahmen seiner Promotion noch intensiver mit der Qualität von Anforderungen und Anforderungsmodellen.