.experience

Aus Ideen wird Wirkung: Requirements Engineering

Ein Magazin von ERNI seit 1999.

Editorial

Liebe Leserinnen und Leser

In einer Zeit, in der Systeme immer komplexer werden, steht das Requirements Engineering im Zentrum erfolgreicher Projekte. Doch was bedeutet das konkret? Es geht darum, die Bedürfnisse der Stakeholder zu verstehen, Prozesse klar zu gestalten, Daten sinnvoll zu strukturieren und neue Technologien wie KI effektiv einzusetzen – und das alles ohne unnötige Komplexität.

In dieser Ausgabe von .experience zeigen wir, wie Pragmatismus und Professionalität Hand in Hand gehen – von der Strategie über die Architektur bis hin zur Usability. Anhand konkreter Beispiele demonstrieren wir, wie schlankes und effizientes Engineering gelingt, ohne dass Qualität oder Compliance leiden.

Wir laden Sie ein, sich inspirieren zu lassen, neue Ansätze kennenzulernen und wertvolle Impulse für Ihre eigenen Projekte mitzunehmen. Viel Freude beim Lesen!

Herzliche Grüsse

Pavo Kohler
CEO, ERNI Group

Inhalt

Editorial 02
Von Pavo Kohler, CEO, ERNI Group

Wie sich Requirements Engineering in der modernen Softwareentwicklung positioniert 04
Von Urs Koepfli, Expert Consultant in Requirements Engineering and Business Analysis, ERNI Schweiz

Der pragmatische Architekt: Skalieren ohne überdimensionieren 09
Von Mihaly Fodor, Principal Engineer, ERNI Rumänien

Warum die Zusammenarbeit von Design- und Entwicklungsteams zu besseren Ergebnissen führt 12
Von Nicola Reinhard, CEO Office, ERNI Schweiz

Von Life Sciences bis Diagnostik: Technische Anforderungen für eine regulierte Markteinführung 15
Von Ares Cabó Carrera, Expert MedTech Consultant and Product Owner und Carolina Lezama, MedTech Delivery Manager, ERNI Spanien

Echtes Usability Engineering für Medizinprodukte nahtlos integriert 19
Von Simon Brendel, Senior Consultant und Stefan Siegle, Principal Consultant, ERNI Deutschland

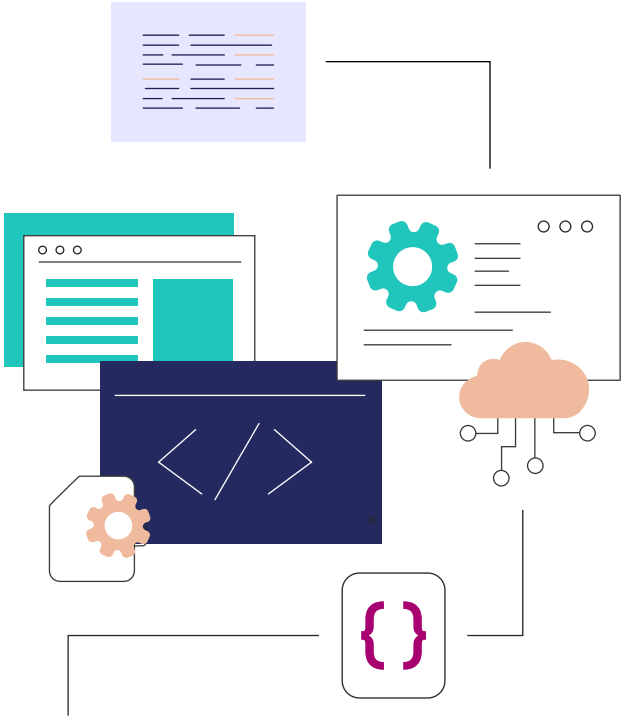
Wie man die Lücke zwischen Endnutzern und IT schliesst 23
Von Caroline Badoud, Senior IT Consultant, ERNI Schweiz

Anforderungsmanagement in komplexen Projekten mit einem digitalen Systemmodell 26
Von Thomas Ruckstuhl, Requirements Engineer, ERNI Schweiz

Wie sich Requirements Engineering in der modernen Softwareentwicklung positioniert

Im digitalen Zeitalter war es nie einfacher, Ideen zu entwickeln. Sie benutzerfreundlich umzusetzen, ist jedoch schwieriger. Requirements Engineering steht vor neuen Herausforderungen: komplexere Produkte, schnellere Zyklen – abgestimmte, flexible Anforderungen sind entscheidend.

Von Urs Koepfli, Expert Consultant in Requirements Engineering and Business Analysis, ERNI Schweiz



Was Requirements Engineering (RE) heute bedeutet

Ich habe in den letzten Jahren die Erfahrung gemacht, dass die Ära der herkömmlichen, umfangreichen Dokumentationen und starren RE-Prozesse vorbei ist. Modernes RE erfordert die Zusammenarbeit funktionsübergreifender Teams, die in der Lage sind, sich den laufend veränderten Anforderungen anzupassen, sowie die lückenlose Rückverfolgbarkeit über den gesamten Lebenszyklus der Anforderungen hinweg. All dies ist besonders wichtig in einem regulierten oder risikoreichen Umfeld, in dem Missverständnisse oder Unvollständigkeiten in den Anforderungen zu kostspieligen Rückschlägen führen können. Im Laufe meiner Arbeit ist mir eine wichtige Veränderung aufgefallen: RE muss heute skalierbar sein und sich nahtlos in agile Entwicklungsumgebungen integrieren lassen, ohne dass die erforderlichen Details oder Compliance-Anforderungen vernachlässigt werden. In diesem Leitartikel möchte ich aufzeigen, was wir in dieser

Ausgabe des .experience näher behandeln: Wie Unternehmen mit den heutigen Herausforderungen umgehen und RE-Praktiken entwickeln, um ihre operativen Ziele in immer komplexeren Systemen zu erreichen.

Für wen entwickeln wir Anforderungen?

In fast jedem Projekt, an dem ich gearbeitet habe, war klar, dass es nie nur einen «Kunden» gibt. Es gibt Nutzer, Käufer, Wartungsbeauftragte, Regulierungsbehörden und Entwickler – und sie alle haben ihre eigenen berechtigten Sichtweisen. Alle Perspektiven zu berücksichtigen führt zu Komplexität. Hier kommt RE ins Spiel und bringt Struktur in diese Komplexität. Stakeholder-Management geht meiner Meinung nach weit über das Abarbeiten von Checklisten und das Einholen von Unterschriften hinaus. Es geht darum, aktiv herauszufinden, was für jeden Stakeholder wichtig ist, und diese Erkenntnisse in eine gemeinsame Richtung zu übersetzen. Es geht darum, alle legitimen Stimmen im Prozess zu identifizieren, zu erfahren, was sie anstreben, und schliesslich eine gemeinsame Richtung zu finden, die die Entwicklung leitet. Vom Techniker vor Ort, der bei Regen (oder anderen Wetterbedingungen) mit Handschuhen arbeitet und versucht, einen ganzen Standort in einer App zusammenzufassen, bis hin zum Governance-, Risiko- und Compliance-Beauftragten, der für den Schutz von Informationen im Sinne der DSGVO verantwortlich ist – jeder Einzelne ist wichtig und liefert entscheidende Informationen, die über den zukünftigen Erfolg entscheiden.

In einem Projekt, an dem ich beteiligt war, wurde die veränderte Rolle des Requirements Engineering besonders deutlich. Ursprünglich erwartete das Team, dass RE lediglich die Anforderungen sammelt und dokumentiert. Im Laufe der Zeit wurde jedoch klar, dass unser eigentlicher Mehrwert ein anderer ist. Als Sprachrohr des Kunden im Lösungsteam halfen wir dabei, vage Erwartungen in konkrete Prioritäten umzusetzen. Mit Entwicklern, Designern und Stakeholdern aus dem Business erarbeiteten wir, was tatsächlich gebaut werden sollte – nicht nur, was technisch möglich war, sondern was einen echten Mehrwert bringen und im realen Kontext des Kunden nutzbar sein würde. Da wurde mir klar: Requirements Engineers sind nicht mehr nur Torwächter für Spezifikationen, sondern Vermittler für gemeinsames Verständnis und gemeinsame Ausrichtung.

«Es geht nicht darum, einfach das zu bauen, was der Kunde sagt, dass er will – sondern darum, seine wahren Ziele zu verstehen und dabei zu helfen, die beste Lösung zu finden und umzusetzen.»

RE ist die Brücke zwischen Geschäftszielen und technischer Umsetzung. Es übersetzt abstrakte Ideen in strukturierte Inputs für die Entwicklung. Besonders in agilen Teams stellt es sicher, dass bei schnellen Iterationen der Gesamtüberblick nicht verloren geht. Am wichtigsten ist vielleicht, dass es Teams hilft, frühzeitig bessere Entscheidungen zu treffen – in einer Phase, in der Änderungen noch kostengünstig sind und grosse Wirkung zeigen.

Das System muss funktionieren

RE ist nicht homogen, sondern besteht aus vier miteinander verbundenen Aufgaben:

1. Ermittlung

Durch Fokusgruppen, Interviews, Workshops und Beobachtungen werden die täglichen Bedürfnisse hinter den Anforderungen ermittelt.

2. Dokumentation

Anforderungen werden in einer lesbaren, testbaren und veränderbaren Form erfasst, hauptsächlich durch strukturierte Texte oder modellbasierte Ansätze.

3. Validierung

Durch die Validierung der Anforderungen stellen wir sicher, dass das Erfasste das richtige Problem adressiert.

4. Management

Durch kontinuierliche Nachverfolgung werden wechselnde Prioritäten und die Rückverfolgbarkeit während des gesamten Lebenszyklus berücksichtigt.

In agileren Situationen wandelt sich RE von einer sehr aufwändigen Aufgabe zu Beginn zu einer kontinuierlichen Zusammenarbeit, während die Dokumentation zu einer allgemeinen, kontinuierlichen Interaktion und zum Informationsaustausch mit Stakeholdern wird. Auch die Fähigkeiten der Tools entwickeln sich weiter – von der KI-gestützten Analyse von Gesetzestexten zu einer kollaborativen Prototyping-Plattform, immer mit einem Ziel vor Augen: Sicherzustellen, dass das Team die richtige Lösung entwickelt.

Die ersten 100 Stunden – eine solide Grundlage schaffen

Bei ERNI habe ich immer wieder erlebt, wie die ersten Tage eines Projekts über den Gesamterfolg entscheiden. Besonders ein Fall ist mir dabei in Erinnerung geblieben: Wir arbeiteten mit einem Kunden aus dem Infrastrukturbereich zusammen. Die Aufgabe schien auf den ersten Blick einfach – eine mobile App für Aussendiensttechnikerinnen und -techniker zu entwickeln, mit der sie Wartungsarbeiten vor Ort dokumentieren können.

Der Kunde hatte zumindest auf den ersten Blick klare Vorstellungen davon, was er wollte: eine elegante Benutzeroberfläche, schnelle Performance und nahtlose Backend-Integration. Doch während der ersten Gespräche ging unser Requirements-Engineering-Team einen Schritt weiter. Wir führten Vor-Ort-Besuche durch, sprachen mit den Technikerinnen und Technikern und versuchten zu verstehen, wie und wo die App eingesetzt werden sollte. Was wir dabei herausfanden, veränderte fast alles.

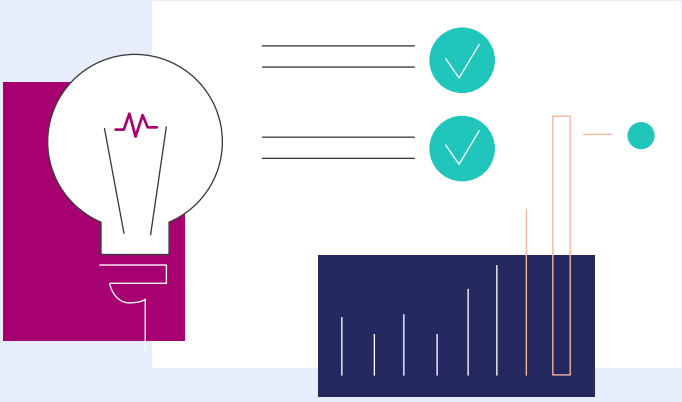
Die meisten Technikerinnen und Techniker arbeiteten das ganze Jahr über im Freien, oft war es kalt und nass. Sie trugen Handschuhe, arbeiteten bei schlechten Lichtverhältnissen und hatten nur wenig Zeit, sich in komplexen Menüs zurechtzufinden. Dies war in der ursprünglichen Einweisung nicht berücksichtigt worden. Hätten wir die Entwicklung lediglich auf Grundlage der ursprünglichen Spezifikationen vorangetrieben, hätten wir zwar ein technisch einwandfreies, aber praktisch unbrauchbares Produkt entwickelt. Tatsächlich war ein früherer Versuch eines anderen Anbieters genau daran gescheitert. Die Benutzeroberfläche funktionierte unter realen Bedingungen schlichtweg nicht.

Diese Erfahrung hat uns in einem Grundsatz bestärkt, den wir bei jedem Projekt befolgen: Man muss frühzeitig das richtige Fundament legen. Oder, wie ich oft sage:

«Wenn man ein vierstöckiges Haus baut, braucht man ein anderes Fundament als für ein Gartenhaus.»

In unserer RE-Praxis betrachten wir die ersten 100 Stunden jedes Projekts als kritisches Zeitfenster, in dem es gilt, zu erkunden, zu verstehen und abzustimmen. Die Schritte, die wir in dieser Phase machen, umfassen in der Regel:

1. Geschäftlichen Kontext und Ziele verstehen	Ziele, Einschränkungen und zugrunde liegende Treiber des Projekts abklären.
2. Alle relevanten Stakeholder einbeziehen	Endnutzer bis zu Compliance-Beauftragten fragen, um sicherzustellen, dass keine kritischen Stimmen überhört werden.
3. Problemfeld definieren und ermitteln	Schwachstellen, Systemgrenzen und Umgebungsbedingungen erfassen.
4. Übergeordnete Anforderungen ermitteln	Nicht nur auf Funktionen fokussieren, sondern auch auf Benutzerfreundlichkeit, Leistung und Einschränkungen.
5. Erste Inputs strukturieren und priorisieren	Bewährte Methoden verwenden, um Must-haves von Nice-to-haves zu trennen.
6. Umfang der ersten Lösung festlegen	Definieren, was geliefert wird – und ebenso wichtig, was nicht geliefert wird.
7. Klar mit dem Team kommunizieren	Eine gemeinsamen Sprache und gemeinsame Tools festlegen und Erwartungen abstimmen.
8. Risiken und Machbarkeit bewerten	Technische, organisatorische oder regulatorische Risiken so früh wie möglich identifizieren.



Dieses Projekt war erfolgreich, weil wir nicht nur eine Checkliste abgearbeitet haben, sondern weil wir uns vor dem Aufbau die Zeit genommen haben, zuzuhören, zu beobachten und Fragen zu stellen. Für mich ist das die Essenz eines gut durchgeführten Requirements Engineerings.

Die Zukunft des Requirements Engineerings

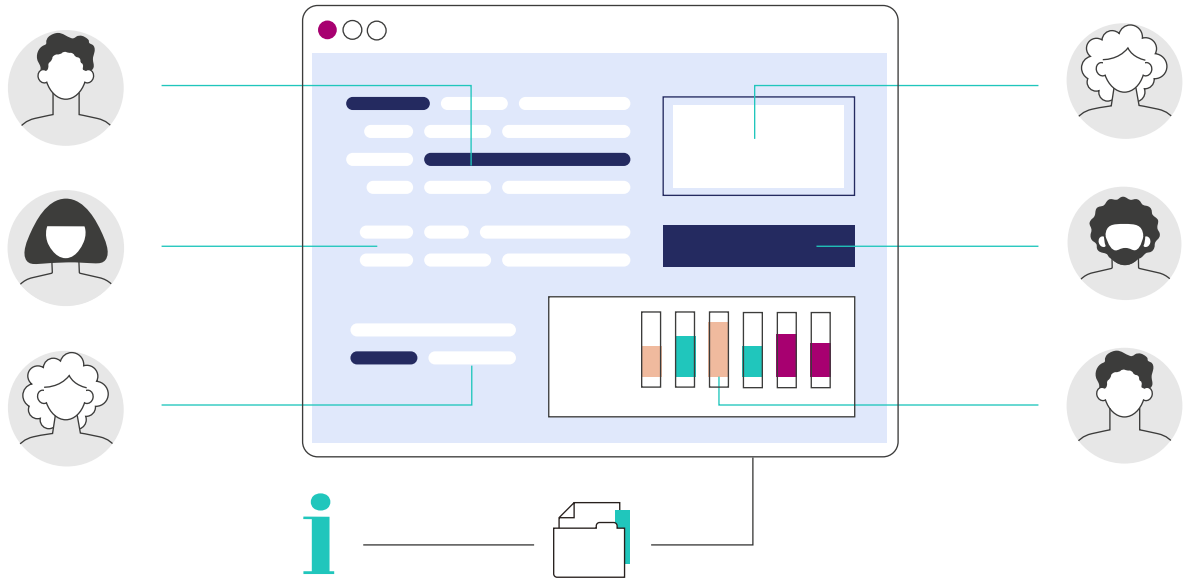
RE als Disziplin hat sich in seinem Kern nicht verändert – aber sein Kontext hat sich sicherlich gewandelt. Es wird keine einzelne Phase mehr sein, sondern ein kontinuierlicher, iterativer Prozess, der mit der Produktentwicklung Schritt hält. In agilen Umgebungen und unter Nutzung von DevOps-Pipelines entwickeln sich Anforderungen in Echtzeit weiter und erfordern daher Tools und Praktiken, um kontinuierliche Verfeinerungen, die Zustimmung der Stakeholder und die Rückverfolgbarkeit effektiv zu gewährleisten. Mit zunehmender Komplexität der Systeme, insbesondere in regulierten Bereichen wie der Medizintechnik oder Automobilindustrie, verlagert sich RE weiter in Richtung formalisierter Modelle und modellbasierter Systementwicklung (MBSE), um die gegenseitigen Abhängigkeiten zu meistern. Visuelle Modelle reduzieren den Kommunikationsaufwand zwischen Geschäfts-, Technik- und Compliance-Teams.

In Zukunft wird RE nicht mehr nur technische Spezifikationen und den Funktionsumfang abdecken, sondern auch ein umfassenderes Verständnis der Nutzerbedürfnisse, die Definition von Geschäftszielen und die Schaffung von Mehrwert ermöglichen.

Agile Arbeitsweisen, die Verbreitung von Tools und insbesondere KI verändern, wie Requirements Engineering durchgeführt wird. Das Warum bleibt aber unverändert: Es geht darum, ein gemeinsames Verständnis zu schaffen. Eine häufig gestellte Frage lautet, ob künstliche Intelligenz RE-Fachleute ersetzen wird. «Ersetzen» ist hier nicht der richtige Begriff, passender wäre «ergänzen». Wenn KI lange Compliance-Dokumente zusammenfassen oder schneller Entwürfe für User Stories erstellen kann, warum sollte man sie nicht einsetzen? Eines kann KI allerdings nicht ersetzen: das menschliche Urteilsvermögen bei der Abwägung von Geschäftszielen, Nutzerbedürfnissen und technischen Gegebenheiten. Dies bleibt auch in Zukunft die zentrale Aufgabe von Requirements Engineers.

Fazit

Bei Requirements Engineering geht es heute nicht mehr nur um Spezifikationen, sondern um das Schaffen einer gemeinsamen Vision. Es bildet die Grundlage für agile, skalierbare und menschenzentrierte Software. RE ist weniger eine klar definierte Rollenverteilung, sondern vielmehr eine flexible Zusammenarbeit. Immer mehr wird es zur Funktion, die nicht nur einen Prozess, sondern die Denkweise in Produktteams verankert.

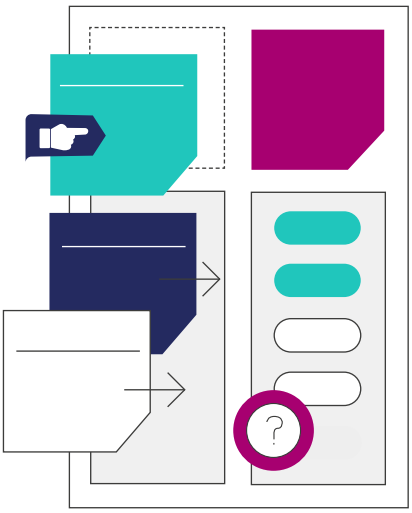


Der pragmatische Architekt: Skalieren ohne überdimensionieren

Komplexität ist verlockend, besonders zu Projektbeginn, wenn die Ideen sprudeln und die Architekturträume gross sind. Doch ohne klare Prioritäten und präzise Anforderungen werden Systeme für hypothetische Szenarien statt für echten Nutzen gebaut. Ein pragmatischer, YAGNI-inspirierter Ansatz hilft, schlank zu starten und trotzdem skalierfähig zu bleiben.

Von Mihaly Fodor, Principal Engineer, ERNI Rumänien

Stellen Sie sich eine allzu vertraute Situation vor. Ihr Software-Engineering-Partner hat gerade Ihren neuen Projektauftrag erhalten und alle im Kick-off-Meeting sind voller Enthusiasmus. Als Kunde bringen Sie eine mutige Idee, einen ehrgeizigen Zeitplan und die Hoffnung mit, dass intelligente Technologien Ihre Herausforderungen lösen. Innerhalb weniger Minuten verschwindet das Whiteboard unter Haftnotizen und Kästchen, Pfeilen, Akronymen, Toolnamen, Bereitstellungsdiagrammen. Alle diskutieren über Kubernetes-Cluster, serverlose Funktionen und die Vorzüge von Microservices gegenüber einem Monolithen. Die Leidenschaft wächst – und in weniger als einer Stunde ist das Team in eine intensive Debatte über die Architektur vertieft, während die User Stories noch immer leer sind. Dann endlich fragt jemand «Was genau soll der Benutzer in Version 1 erhalten?». Es wird still im Raum und diese Stille offenbart ein bekanntes Risiko: Wenn die Begeisterung die Konzentration übersteigt, läuft das Team Gefahr, für eine imaginäre Zukunft zu entwickeln statt sich auf die aktuellen Anforderungen zu konzentrieren. Wenn Anforderungen vage und ungeordnet bleiben, wird die Architektur zur Spekulation und die Lösung letztlich für Annahmen statt für echten Wert optimiert.



Verlockende Komplexität

Requirements Engineering ist mehr als nur eine Checkliste. Es ist das Fundament, auf dem eine zuverlässige Architektur aufbaut. Teams benötigen einen praktischen, klar definierten Rahmen, der unterscheidet, was heute geliefert werden muss und was bis morgen warten kann. Diese Klarheit schützt das Projekt vor unnötiger Komplexität. Eine solide Architektur beginnt mit einem klaren Ziel, nicht mit einem Tool oder Muster.

Dennoch beginnen viele Projekte immer noch mit einer Wunschliste: «Wir brauchen den kompletten Stack.». Microservices, serverlose Funktionen und Multi Cloud Setups tauchen auf, bevor auch nur eine einzige User Story geschrieben wurde. Ich habe einmal an einem Projekt gearbeitet, bei dem der Kunde zunächst davon überzeugt war, dass ein ausgeklügeltes Cloud Design den B2B-Umsatz steigern würde. Gemeinsam haben

wir dann herausgerabeitet, dass ein zuverlässiges Bestellsystem die Lösung für sein Problem war. Die Kluft zwischen wahrgenommenen und tatsächlichen Bedürfnissen ist der Ort, an dem Komplexität entsteht. Hier bewährt sich das YAGNI-Prinzip (You aren't gonna need it, du wirst es nicht brauchen). YAGNI sorgt dafür, dass sich das Team darauf konzentriert, nur das zu liefern, was notwendig ist, und zwar dann, wenn es notwendig ist, und es gründet jede architektonische Entscheidung auf reale Anforderungen statt auf Annahmen.

So kommt es zur Überdimensionierung

Überdimensionierung beginnt oft mit guten Absichten, aber falschen Annahmen. Vier Muster tauchen immer wieder auf:

Vier typische Muster, die Komplexität verursachen:

1. Unrealistische Prognosen

Entwürfe für «Millionen von Nutzern», obwohl die Schätzung für das erste Jahr nur wenige Hundert beträgt.

2. Nachahmung ohne Kontext

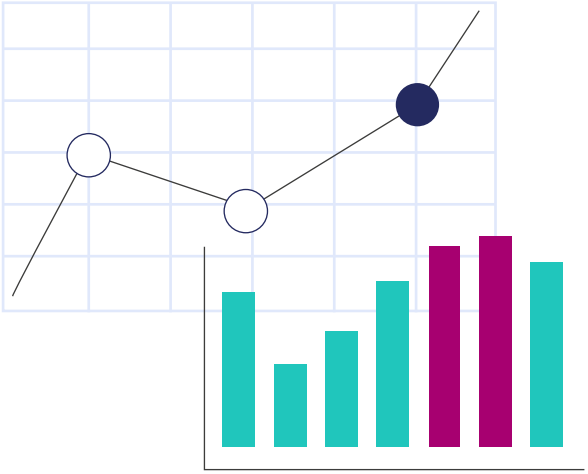
Übernahme der Architektur von Unternehmen wie Netflix, ohne deren Grösse oder Budget zu erreichen.

3. Versteckte Kostenfallen

Auswahl eines serverlosen Pay-per-Request-Modells, obwohl die Arbeitslast konstant ist und ständig läuft.

4. Scope Drift

Hinzufügen von Nice-to-have-Funktionen, die stillschweigend zu obligatorischen Funktionen werden und die Komplexität erhöhen.



Solche Entscheidungen werden in der Regel durch die Angst vor unbekannten Zukunftsfaktoren, unklare nicht-funktionale Ziele oder eine Kultur getrieben, die Komplexität belohnt. Die Folgen sind klar: langsamere Bereitstellung, fragile und komplizierte Systeme, höhere Betriebskosten und Entscheidungsmüdigkeit, die das Team erschöpft. Wenn ein Projektteam erkennt, dass es zu viel gewollt hat, sind die Kosten für Änderungen bereits hoch.

Der pragmatische Prozess

Mein Ansatz beim Requirements Engineering ist praxisbezogen und orientiert sich an realen Grenzen und dem geschäftlichen Nutzen. Er beginnt mit einer detaillierten Überprüfung aller Dokumente, bei der nicht relevante Aussagen entfernt werden, um die wenigen wirklich wichtigen Anforderungen herauszuarbeiten. Sobald diese klar sind, gehe ich zur Dimensionierung über und überprüfe Kernkennzahlen wie die erwartete Anzahl Nutzerinnen und Nutzer, das Datenvolumen, die Service Levels oder das Release-Tempo. Jede unklare Angabe wird sofort mit dem Kunden thematisiert. Ich bewerte den Funktionsumfang mit T-Shirt-Grössen (XS bis XL), einer schnellen Methode, mit der überdimensionierte oder vage Anforderungen frühzeitig erkannt werden.

Auf dieser Grundlage skizziere ich eine minimal funktionsfähige Architektur – oft nur drei Kästchen und zwei Pfeile. Alles, was nicht auf das Whiteboard passt, wird hinterfragt – wenn es nicht passt, gehört es wahrscheinlich nicht in den Plan für den ersten Tag. Vor der Schätzung überprüft ein Kollege den Entwurf und hinterfragt alle Annahmen. Der letzte Kontrollpunkt ist die Rückkehr zum Geschäftsziel. Wenn ein Designelement die wichtigsten KPIs nicht unterstützt, wird es verworfen. Das Ergebnis ist eine schlanke, validierte Grundlage, die auf Fakten statt auf Spekulationen basiert.

Ein von YAGNI inspirierter Referenzfall

Ein mittelständischer Hersteller von Sonnenschutzsystemen beauftragte uns mit der Entwicklung einer AWS-Lösung auf Basis von Microservices, automatischer Skalierung und separaten Teams für jeden Dienst. Ziel war es, eine neue Bestellplattform «zukunftssicher» zu machen und für weiteres Wachstum vorzubereiten. Auf dem Papier sah der Plan modern und solide aus.

Wir stellten schnell fest, dass die erwartete Spitzenauslastung unter 50 gleichzeitigen Nutzerinnen und Nutzern lag, weit unter dem Niveau, das eine verteilte Konfiguration rechtfertigt. Selbst unser erster Entwurf enthielt noch Lastverteiler (Load Balancer) und zustandslose Anwendungen, da wir von einem Wachstum ausgingen, das noch Jahre entfernt war. Die ersten AWS-Rechnungen zeigten den Fehler: Das System befasste sich mit Problemen, die noch gar nicht existierten.

Mit einer YAGNI-Denkweise haben wir Überflüssiges entfernt. Wir haben die Verbreitung von Microservices durch einen modularen Monolithen ersetzt und klare interne Grenzen beibehalten, damit Teile bei steigender Nachfrage später aufgeteilt werden können. Load Balancer wurden abgeschafft und die Bereitstellung wurde vereinfacht.

Das Unternehmen erreichte alle Geschäftsziele mit etwa einem Fünftel der bisherigen Infrastrukturkosten. Die Markteinführungszeit verkürzte sich, die Wartung wurde einfacher und das Team konnte sich auf die Kernfunktionen konzentrieren, anstatt sich mit Cloud-Overhead zu beschäftigen. Ebenso wichtig ist, dass die Plattform bei steigender Nachfrage weiterwachsen kann. Dieser Fall zeigt, dass eine solide Architektur mit dem beginnt, was notwendig ist, und nicht mit dem, was gerade im Trend liegt.

Was wir gelernt haben

Der Lean-Ansatz funktioniert, weil er verhindert, dass der Umfang zu früh zu gross wird. Klare Grenzen führen zu einfacheren Designs und genaueren Schätzungen. Der grösste Widerstand kommt aus der Angst vor einer festen Bindung. Ich begegne dem, indem ich die definierten Schnittstellen zeige, an denen Teile später ausgetauscht werden können, wenn ein messbarer KPI dies erfordert. In stark regulierten oder sicherheitskritischen Bereichen wie dem Echtzeit-Handel oder bei medizinischen Geräten kann von Anfang an zusätzliche Stabilität erforderlich sein. Selbst dann sollte jede hinzugefügte Ebene einen schriftlichen, überprüfbaren Grund für ihre Existenz haben.

Warum die Zusammenarbeit von Design- und Entwicklungsteams zu besseren Ergebnissen führt

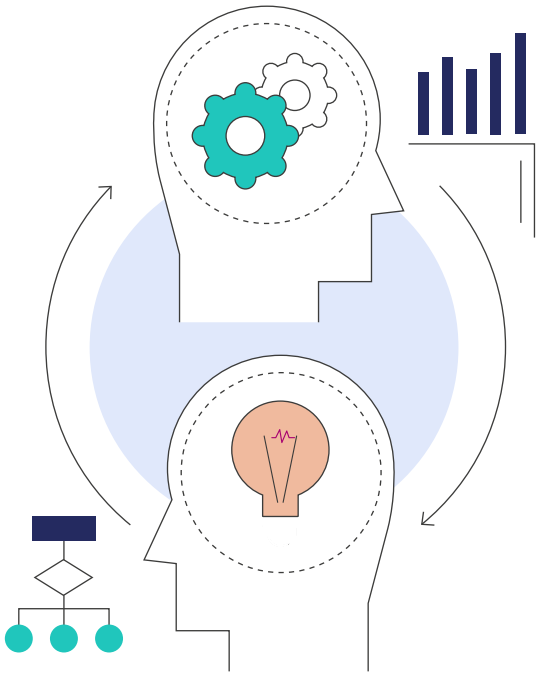
In einem Projekt leitete ich die Modernisierung einer zentralen Anwendung. Ziel war es, ein veraltetes Tech-Stack zu erneuern, um Skalierbarkeit und Wartbarkeit zu sichern. Aus dem Upgrade wurde eine Lektion in bereichsübergreifender Zusammenarbeit, vor allem zwischen Software-Entwicklung und Design.

Von Nicola Reinhard, CEO Office, ERNI Schweiz

Ein dualer Fokus mit einer Timeline

Zu Projektbeginn haben wir einen ehrgeizigen, jedoch realistisch umsetzbaren Zeitplan festgelegt. Wir haben ihn sorgfältig auf die Entwicklungskapazitäten abgestimmt und mit den Stakeholdern aus dem Unternehmen validiert. Unser Hauptaugenmerk lag dabei auf der Modernisierung der veralteten Technologien, ohne die Funktionen zu verändern. Die Entwicklerinnen und Entwickler haben sich voll und ganz darauf konzentriert, da sie wussten, dass der Erfolg davon abhing, eine hochmoderne Anwendung fristgerecht bereitzustellen.

Neben der technischen Aufrüstung wollten wir jedoch auch die Benutzeroberfläche und das Gesamterlebnis verbessern, um ein modernes Erscheinungsbild zu schaffen, das die Endnutzerinnen und Endnutzer begeistern würde. Daher wurde zunächst ein UX-Designer in das Team integriert. Dies war zwar eine strategische Entscheidung, führte jedoch zu zusätzlicher Komplexität. Was war das Risiko? Ein potenzieller Interessenkonflikt und ein zeitlicher Konflikt.



Die Designerinnen und Designer wollten Interaktionen, Layouts und Abläufe überdenken. Die Entwicklerinnen und Entwickler hatten jedoch bereits damit begonnen, die Benutzeroberfläche auf dem neuen Stack neu zu gestalten, und hatten ihre eigenen Vorstellungen vom Design. Die Berücksichtigung der Designänderungen bedeutete zusätzlichen Aufwand und Anpassungen, die in unserem ursprünglichen Zeitplan nicht vorgesehen waren. Der uneinheitliche Ansatz drohte zu einer fragmentierten Benutzererfahrung und möglichen Reibungsverlusten innerhalb des Teams zu führen.

Inspiriert vom Denken in der Produktion

Ein Wendepunkt kam unerwartet, als ich Walter Isaacsons Biografie über Elon Musk las. Musk versetzte sein Designteam in die Fabrikhalle und forderte es auf, gemeinsam mit den Ingenieurinnen und Ingenieuren auf iterative Weise Störungen zu beheben. Diese Integration der Problemlösung über die Grenzen von Design und Technik hinweg hat mich inspiriert. Ich wusste, dass wir eine ähnliche Zusammenarbeit brauchten. Design und Entwicklung konnten nicht isoliert arbeiten. Wir mussten die Mentalität der Übergabe ablegen und ein Modell der gemeinsamen Verantwortung fördern.

Echte Integration statt nur Zusammenarbeit

Sobald das Designteam an Bord war, haben wir dafür gesorgt, dass es in alle Entwicklungsmeetings eingebunden ist. Wir haben ein wöchentliches Design-Meeting eingeführt, um Prioritäten und Entscheidungen abzustimmen. Alle Designentscheidungen haben wir auf einem gemeinsamen Miro-Board festgehalten, sodass sie für alle sichtbar, gut organisiert und gemeinsam gepflegt werden konnten. Dabei haben sich drei Erfolgsfaktoren herauskristallisiert:

Verantwortung durch Designkompetenz

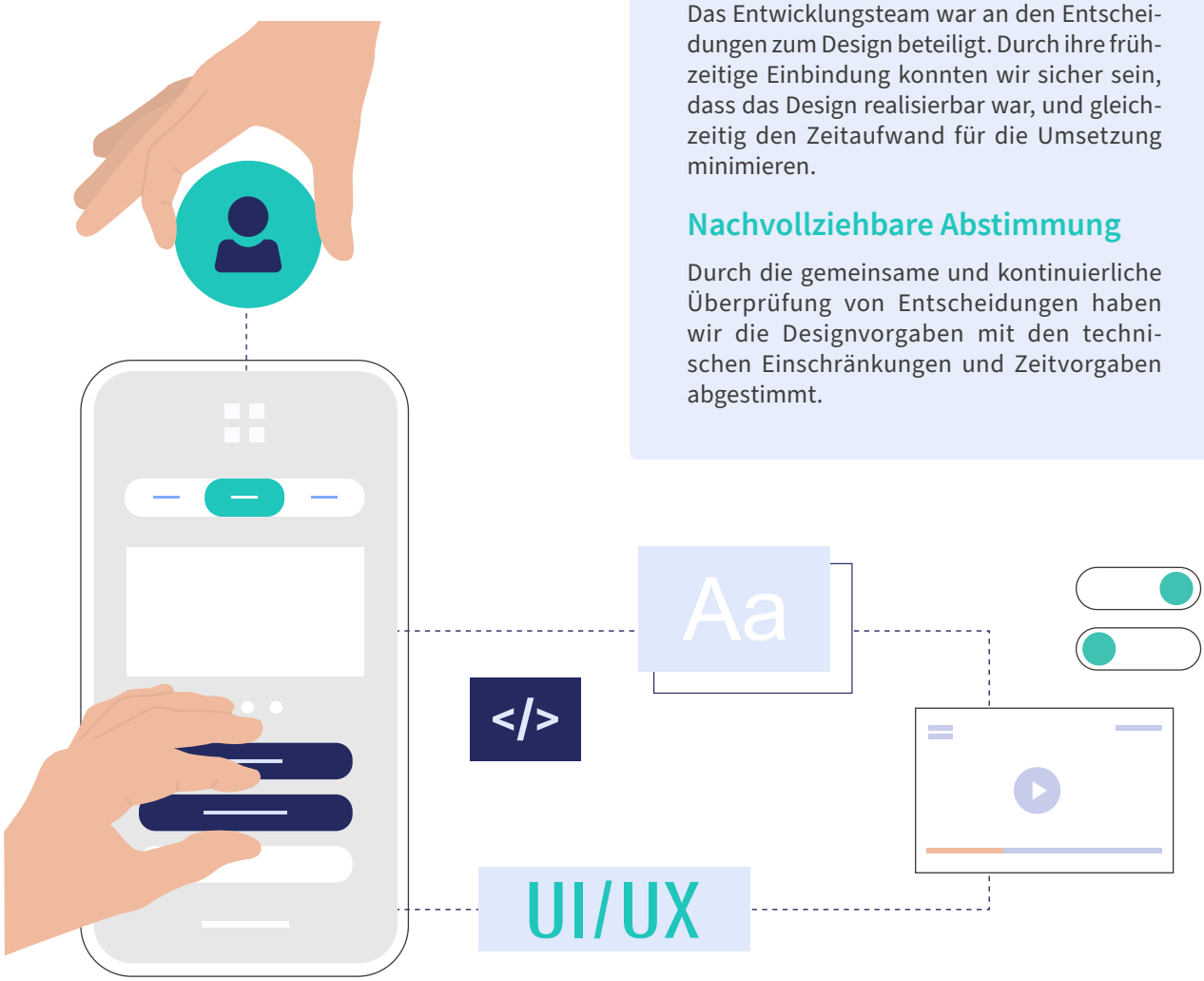
Alle wichtigen Designentscheidungen wurden übers Designteam kanalisiert. Das bedeutete nicht, dass diese blind umgesetzt wurden, sondern dass sie zuvor auf der Grundlage von Argumenten und Prinzipien sorgfältig diskutiert wurden.

Gleiches Verständnis

Das Entwicklungsteam war an den Entscheidungen zum Design beteiligt. Durch ihre frühzeitige Einbindung konnten wir sicher sein, dass das Design realisierbar war, und gleichzeitig den Zeitaufwand für die Umsetzung minimieren.

Nachvollziehbare Abstimmung

Durch die gemeinsame und kontinuierliche Überprüfung von Entscheidungen haben wir die Designvorgaben mit den technischen Einschränkungen und Zeitvorgaben abgestimmt.



Das Navigationsdilemma

Eine echte Bewährungsprobe für unsere Zusammenarbeit kam schon früh. Bevor das Designteam zu uns kam, hatten wir das neue Navigationssystem der Anwendung fertiggestellt. Bei ihrer Ankunft wiesen die Designerinnen und Designer auf mehrere Probleme hin: Es war nicht intuitiv, auf kleinen Bildschirmen ineffizient und berücksichtigte wichtige Nutzerpfade nicht. Worin lag nun die Herausforderung? Das System war bereits implementiert und unser Backlog voll.

Anstatt das Projekt zu gefährden oder einen Kompromiss zu erzwingen, haben wir uns darauf geeinigt, die Änderungen an der Navigation zu verschieben. Wir beschlossen, uns zunächst auf die Technologieumstellung zu konzentrieren und die Navigation zu überarbeiten, sobald die Kapazitäten dies zulassen. Diese bewusste Verschiebung hat sich ausgezahlt. Wir haben nicht nur unseren Termin eingehalten, sondern auch die Navigation kurz vor dem Go-Live erfolgreich überarbeitet.

Leistung, Geschwindigkeit und Qualität

Anfangs war es eine Herausforderung, die Designerinnen und Designer in Zeitverzug ins Team zu holen. Aber die Art und Weise, wie wir sie integriert haben, machte den Unterschied. Die Entwicklerinnen und Entwickler respektierten ihre Entscheidungen, weil sie fundiert waren und ihre Bedenken berücksichtigten. Die Designerinnen und Designer entwickelten sich wiederum gemeinsam mit dem Team weiter und leisteten schliesslich sogar einen direkten Beitrag zum Code. Diese enge Zusammenarbeit führte zu einer schnelleren Umsetzung und einheitlicheren Benutzererfahrung. Das Team fühlte sich sowohl für das Design als auch für den Code verantwortlich, was sich auch im Endprodukt widerspiegelte.

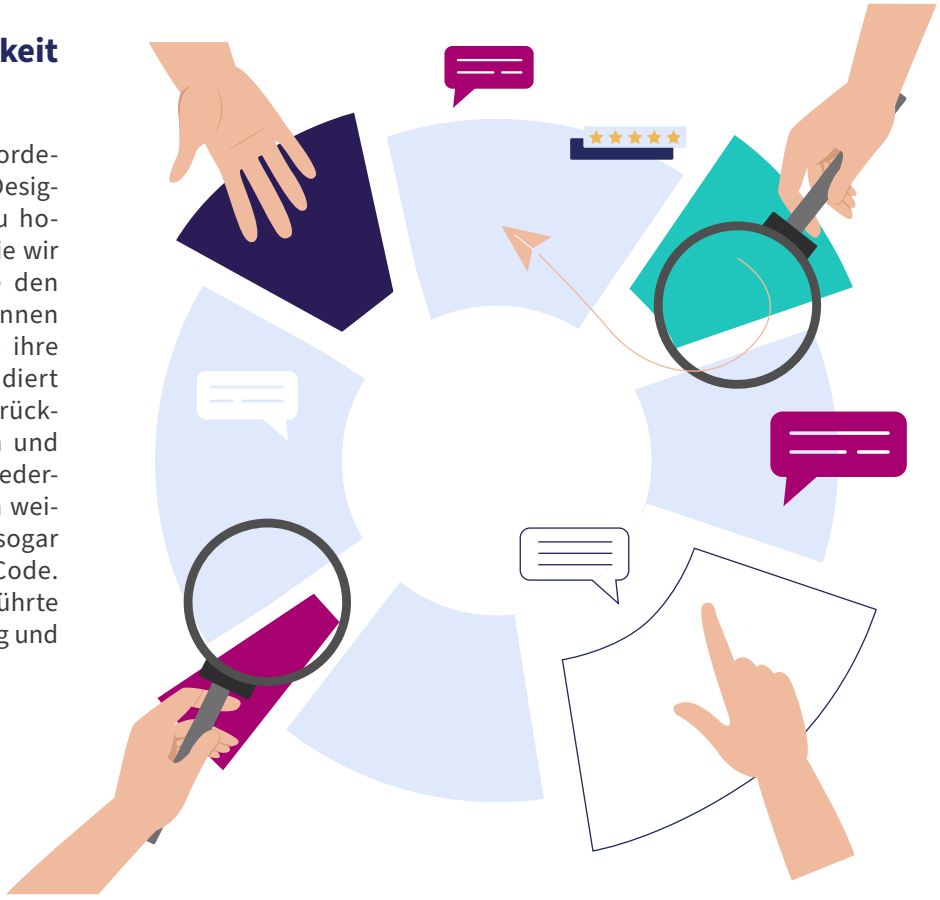
Ziel: Digitale Erlebnisse schaffen

Design und Entwicklung müssen nicht getrennte Bereiche sein – sie funktionieren am besten, wenn sie eng miteinander verzahnt sind.

Integration bedeutet nicht, dass alle Designerinnen und Designer programmieren müssen. Es bedeutet, von Anfang an den Dialog, das Vertrauen und den gegenseitigen Respekt zu fördern.

Entwicklerinnen und Entwickler bringen fundierte Kenntnisse über die Machbarkeit und die Erwartungen der Nutzerinnen und Nutzer mit. Designerinnen und Designer bringen die Vision von Benutzerfreundlichkeit und Ästhetik ein. Gemeinsam können sie mehr als nur funktionale Software entwickeln; sie können aussergewöhnliche digitale Erlebnisse schaffen.

Mein Rat an andere Projektmanager und Technologieführerinnen: Behandeln Sie Design nicht als Phase oder Zusatz. Integrieren Sie es. Machen Sie es sich zu eigen. Und beobachten Sie, wie Ihre Teams und Ihre Anwendungen aufblühen.



Von Life Sciences bis Diagnostik: Technische Anforderungen für eine regulierte Markteinführung

Die Umwandlung eines Life-Science- oder Research-Use-Only-Geräts in ein Medizinprodukt ist voller Chancen und Herausforderungen. Wir unterstützten ein Unternehmen bei der Konvertierung eines Labor-Diagnosegeräts für die klinische Nutzung. Der Artikel zeigt, wie Expertise im regulierten Umfeld den Erfolg sichert.

Von Ares Cabó Carrera, Expert MedTech Consultant and Product Owner und Carolina Lezama, MedTech Delivery Manager, ERNI Spanien

Herausforderungen

Obwohl die Kerntechnologie dieselbe zu sein schien, veränderten Patientensicherheit, gesetzliche Haftung, Rückverfolgbarkeit und andere Dokumentationsanforderungen fast alles. Die regulatorischen Anforderungen, Erwartungen und der Umfang der kontrollierten Prozesse wurden erheblich erweitert – und damit auch die Erwartungen an die Produktentwicklerinnen und - entwickler, die sich vom Tempo oder der Flexibilität der Entwicklung hin zu vollständiger Verantwortlichkeit und validierter Rückverfolgbarkeit verlagerten.

In diesem Fall wurden die Herausforderungen deutlich und waren vielfältig: fragmentierte Anforderungen, sich ständig ändernde Bedürfnisse der Stakeholder, Fristen, ISO-, FDA- und EU-MDR-Konformität sowie das Bewusstsein für die Notwendigkeit, kostspielige Nacharbeiten und Compliance-Verstösse zu vermeiden.

Unterschied zwischen einem Life-Science-Gerät und einem Medizingerät

Life-Science-Gerät

Ein Life-Science-Gerät ist ein Apparat, Werkzeug oder Instrument, das in der biologischen, biochemischen, pharmazeutischen und Laborforschung verwendet wird. Es unterstützt wissenschaftliche Entdeckungen, die Entwicklung von Medikamenten, die Diagnostik und die biotechnologische Fertigung. Es kommt nicht immer in direkten Kontakt mit Patientinnen und Patienten und wird häufig in Laboren, Biotech- oder Pharmaunternehmen und Forschungsinstituten verwendet.

Medizinprodukt

Als Medizinprodukt werden unter anderem Apparate, Gegenstände, Stoffe und auch Software bezeichnet, die zu therapeutischen oder diagnostischen Zwecken für Menschen verwendet werden. Es ist für den klinischen Gebrauch bestimmt, das heisst zur Anwendung am oder im Körper von Patientinnen und Patienten, wird von Gesundheitsbehörden wie der FDA (USA), der EMA oder MDR (EU) reguliert und nach Risikostufen wie Klasse I, II oder III klassifiziert.

Prozess: Der Weg zur Regulierung

Unser Ansatz für diese Herausforderungen war Requirements Engineering. Es bot von Anfang an eine nachvollziehbare und strukturierte Grundlage, damit wir die technischen Aspekte der Entwicklung an die regulatorischen Anforderungen anpassen, Änderungen angemessen und effektiv verwalten und sicherstellen konnten, dass keine kritischen Details übersehen oder vergessen wurden. RE diente als Hebel für den Übergang von der wissenschaftlichen Erfindung zur klinischen Konformität.

Die Umwandlung von Software für ein Life-Sciences-Forschungsinstrument in ein reguliertes medizinisches Diagnosegerät ist keine lineare Aufrüstung, sondern eine grundlegende Veränderung in der Denkweise, Methodik und Verantwortung. Diese Transformation führte unsere Teams aus dem relativ flexiblen Bereich der wissenschaftlichen Software in die stringente Welt der regulierten Diagnostik, in der jede Funktion gerechtfertigt, jedes Risiko minimiert und jede Zeile Code nachvollziehbar sein muss.

In Bezug auf Research-Use-Only-Produkte optimieren Teams häufig hinsichtlich Schnelligkeit, Funktionalität und Erkundung.

Die Dokumentation ist weniger umfangreich, das Testen pragmatisch und das Risikomanagement eher informell. Im Gegensatz dazu erfordert ein Medizinprodukt, das die Anforderungen der FDA und IVDR erfüllen muss, Prozessreife, systematische Rückverfolgbarkeit und verifizierte Qualität in jeder Phase.

Das bedeutete, dass wir unsere Arbeitsweise überdenken mussten, angefangen bei der Dokumentation und Validierung der Benutzeranforderungen bis hin zur Überprüfung des Softwareverhaltens unter realen klinischen Bedingungen. Das Risikomanagement war nicht mehr nur eine Hintergrundaktivität, sondern wurde in jeden Backlog-Refinement- und Entscheidungsprozess integriert.

Agilität in einer regulierten Welt neu definieren

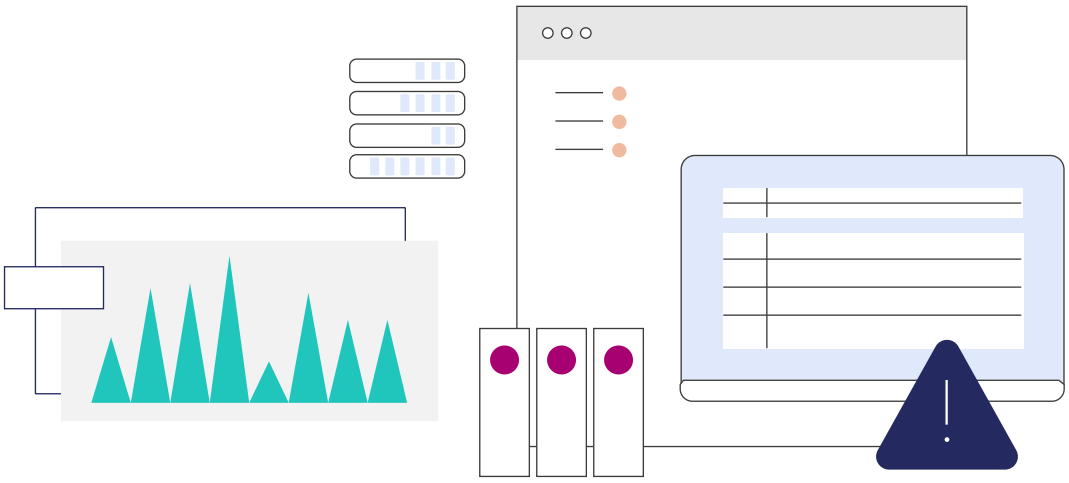
Agile Prinzipien blieben im Kern unserer Herangehensweise, aber wir mussten sie an ein Compliance-orientiertes Umfeld anpassen. Statt schnellen Iterationen auf Kosten der Nachvollziehbarkeit haben wir ein hybrides agiles Modell entwickelt, das die Anpassungsfähigkeit bewahrt und gleichzeitig vollständige Nachvollziehbarkeit, Validierung und Compliance gewährleistet.

- **Erstellen von sukzessiven Release-Kandidaten statt einer einzigen monolithischen Einführung. Die Modularität der Architektur diente der Optimierung des Codes und der Vereinfachung der zukünftigen Wartung.**
- **Definition von minimal existenzfähigen Produkten (Minimum Viable Products, MVP), die nicht nur funktional einsetzbar waren, sondern auch konform mit den gesetzlichen Vorschriften. So konnten wir dem Kunden bereits in einer frühen Phase realistische Ergebnisse liefern.**
- **Die Definition of Done (Fertigstellungsdefinition) ging weit über «funktionierende Software» hinaus. Jede User Story musste mit validierten Anforderungen, Testfällen, Akzeptanzkriterien und dokumentierten Verifizierungsergebnissen unterlegt werden.**

Rückblickend würden wir bei einem so grossen Projekt mit 100 Teammitgliedern, die über mehrere Standorte in Europa verteilt sind, ein anderes agiles Framework wählen – höchstwahrscheinlich SAFe, da zu viele unterschiedliche Teams beteiligt waren und nicht alle Teams denselben Iterationsprozess befolgten.

Risiko im Fokus

Wir haben das Risikomanagement direkt in das Backlog Refinement integriert. Stories wurden nicht nur nach ihrem geschäftlichen Nutzen oder ihrer Komplexität, sondern auch nach ihrem Risiko und ihrer regulatorischen Relevanz unterteilt. So wurden beispielsweise Funktionen zur Datenintegrität wie Audit-Protokolle oder die Rückverfolgbarkeit von Ergebnissen wegen ihrer Auswirkungen auf die Patientensicherheit und die Compliance ganz oben in die Roadmap gesetzt. Dies half dem Team, unter Druck Prioritäten zu setzen, insbesondere wenn der Umfang wegen enger Zeitvorgaben neu ausgehandelt werden musste. Das MVP war nicht das absolute Minimum, sondern das minimal zertifizierbare Produkt.



Neue Definition von Product Ownership

In diesem Umfeld waren Product Owner (PO) mehr als nur Feature-Gatekeeper. Sie fungierten als Übersetzer zwischen Stakeholdern, Compliance-Teams und Entwicklerinnen und Entwicklern und stellten sicher, dass sich wandelnde Kundenanforderungen richtig interpretiert und im Einklang mit den regulatorischen Erwartungen umgesetzt wurden. Product Owner und Requirements Engineers arbeiteten Seite an Seite, um klinische Anforderungen mit Entwicklungsbeschränkungen in Einklang zu bringen und ständig zu überprüfen, ob jedes Ergebnis den regulatorischen Anforderungen standhalten konnte – bevor es das Ende eines Sprints erreichte, geschweige denn den Markt.

Dokumentation als «Lebewesen»

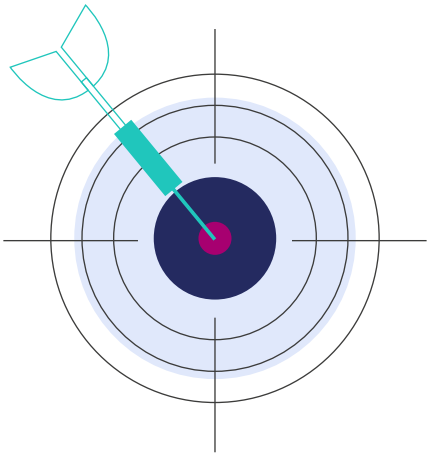
Während des gesamten Projekts haben wir die Dokumentation nicht als Mehraufwand betrachtet, sondern als lebendiges Kapital – als dynamisches, sich weiterentwickelndes Wissensreservoir, das sichere Entscheidungen ermöglichte, die Einhaltung gesetzlicher Vorschriften gewährleistete und Skalierbarkeit ermöglichte. Jeder Release Candidate wurde als eigenständiger Meilenstein behandelt. Anstatt die gesamte Dokumentation vorab zu erstellen oder bis zur endgültigen Veröffentlichung aufzuschieben, haben wir sie schrittweise und konsistent aufgebaut. Jede User Story war Teil einer umfassenderen User Journey. Auf diese Weise wurde die Dokumentation zu einem Instrument für die Abstimmung und Qualitätssicherung und nicht nur für die Einhaltung von Vorschriften. Sie ermöglichte es allen Beteiligten, von den Entwicklerinnen und Entwicklern bis hin zu den Prüfstellen, zu verstehen, was entwickelt wurde, warum es wichtig war und wie es überprüft wurde.

Lösung

Gemeinsam mit unserem Kunden haben wir nicht nur ein konformes und validiertes Gerät entwickelt, sondern auch ein modulares, skalierbares Produkt, das bereit für den globalen Markt war.

Wir freuen uns, dass das Produkt nun auf dem Markt ist – das Feedback, das wir erhalten haben, war positiv, nicht nur von internen Stakeholdern, sondern auch von Endnutzerinnen und Endnutzern in klinischen Umgebungen, die nun von dem Gerät profitieren.

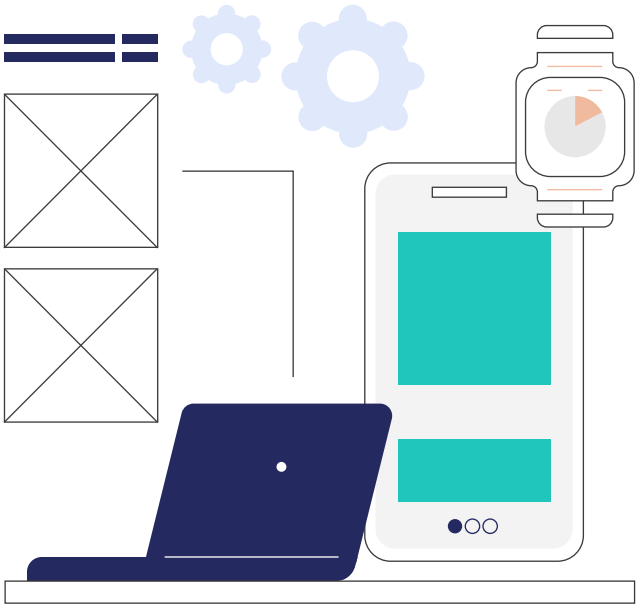
Zu den wichtigsten Erkenntnissen, die wir gewonnen haben, gehört, dass eine frühzeitige Abstimmung der Produktanforderungen von entscheidender Bedeutung ist, dass agile Praktiken auf regulierte Umgebungen zugeschnitten und nicht einfach übertragen werden müssen und dass vor allem das Requirements Engineering und die Dokumentation als strategische Wegbereiter und nicht als administrativer Aufwand betrachtet werden müssen.



Echtes Usability Engineering für Medizinprodukte nahtlos integriert

In der Medizintechnik spielt Usability Engineering oft nur eine Nebenrolle und wird schnell vor der Zulassung umgesetzt. Frühzeitig integriert schafft Usability sichere Produkte, die sich abheben. In diesem Artikel zeigen wir, wie sie Effizienz, Qualität und Markterfolg selbst in regulierten Umgebungen steigert.

Von Simon Brendel, Senior Consultant und Stefan Siegle, Principal Consultant, ERNI Deutschland



Usability – oft unterschätzt, selten richtig gemacht

Die regulatorischen Anforderungen – zum Beispiel DIN EN 62366, MDR (Medical Device Regulation), IVDR (In Vitro Diagnostic Medical Devices and Repealing Directive) oder die FDA Human Factors Guidance – fordern eine Mindest-Gebrauchstauglichkeit, die in erster Linie auf die Vermeidung von Gefährdungen abzielt. Diese Vorgaben genügen jedoch nicht, um ein gutes Produkt zu bauen. Denn: Die Normen verlangen zwar, dass Risiken adressiert werden – nicht aber, dass Produkte intuitiv, effizient oder gar gerne genutzt werden. Dieses Missverständnis führt in der Medizintechnologie dazu, dass viele Usability-Massnahmen zu spät oder halbherzig erfolgen – mit teuren Korrekturschleifen oder Fehlentwicklungen als Folge.

Früh starten, richtige Prioritäten setzen

Ob Chefärztin, Laborassistent oder Pflegekraft: Jedes System hat mehrere Nutzergruppen mit unterschiedlichen Aufgaben und Bedürfnissen. Oft wird an dieser Vielfalt vorbeientwickelt – und damit die Akzeptanz mancher Nutzergruppen aufs Spiel gesetzt.

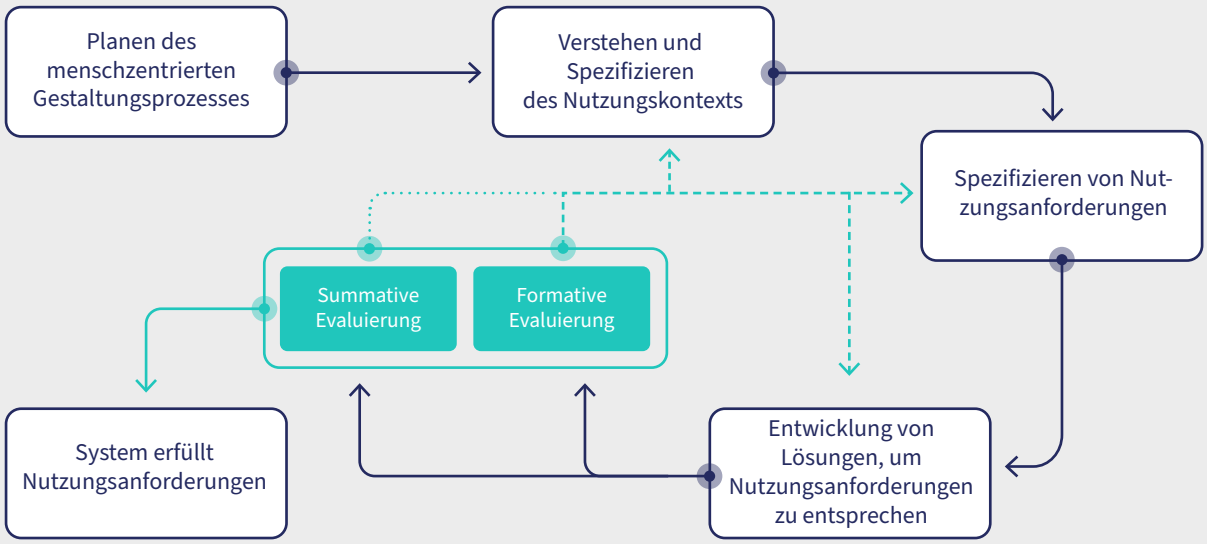
Dass häufig nicht die Endnutzerinnen und Endnutzer entscheiden, welches Produkt angeschafft wird, sondern Einkaufsabteilungen oder ärztliche Leitungen, die selbst nicht oder nur selten mit dem Gerät arbeiten, erschwert die flächendeckende Einführung echter Usability zusätzlich. So wird die Qualität der Gebrauchstauglichkeit zu selten zum ausschlaggebenden Kriterium bei Kaufentscheidungen – obwohl sie im klinischen Alltag entscheidend für Sicherheit, Effizienz und Akzeptanz ist. Besonders Effizienzsteigerungen, die durch gute Usability erzielt werden, wirken sich langfristig gesehen sehr positv auf die Wirtschaftlichkeit eines Unternehmens aus. Warum also wird dem Thema nicht mehr Gewicht gegeben?

Compliance als Nebenprodukt guter Prozesse

Usability Engineering sollte nicht nur als «Nice to have» angesehen werden, sondern als integraler Bestandteil guter Requirements-Engineering-Prozesse. Wer früh die Nutzung analysiert, Aufgabenmodelle erstellt, Personas entwickelt und iterativ testet, erhält automatisch die Artefakte, die später für die Zulassung benötigt werden – und das aus der Entwicklungslogik heraus und nicht aus Pflichtgefühl. Sie basieren auf realen Erkenntnissen und nicht nur auf Annahmen.

Vom Nutzer zum Artefakt – ein schlanker Prozess

Abbildung: Der ERNI Medical Usability Prozess ergänzt den menschzentrierten Gestaltungsprozess nach ISO 9241-210 um Aspekte, die für die Zulassung von Medizinprodukten erforderlich sind.



Wir haben einen Prozess entwickelt, der das Vernachlässigen von echtem Usability Engineering in MedTech-Projekten verhindert. Er basiert bewusst auf dem Vorgehen für menschzentrierte Gestaltung nach DIN EN ISO 9241-210, welches in Branchen gängig ist, in denen eine gute, echte Usability historisch einen viel höheren Stellenwert hat als in der Medizintechnik. Beispielsweise im E-Commerce, der Spiele- und Entertainment-Branche oder in der Entwicklung von B2C-Apps.

Unser Prozess besteht aus den klassischen Elementen der menschzentrierten Gestaltung: Planung, Kontextanalyse, Ableitung von Anforderungen und Lösungsentwicklung. Bei der iterativen Evaluierung von Prototypen und anderen Entwicklungsartefakten differenzieren wir zwischen formativen und summativen Evaluierungen. Unser Vorgehen funktioniert sowohl im V-Modell als auch agil oder hybrid.

Sofern alle Schritte unseres Prozesses sorgfältig und zum richtigen Zeitpunkt umgesetzt werden, entstehen automatisch alle für gute Usability gängigen und regulatorisch geforderten Artefakte:

1. Menschzentrierten Gestaltungsprozess planen

- **Aus DIN EN ISO 9241-210**
  Menschzentrierte Qualitätsziele, Ressourcenplan, Zeitplan, Zielgruppen- und Nutzergruppendefinition

2. Nutzungskontext verstehen und spezifizieren

- **Aus DIN EN ISO 9241-210**
  Nutzergruppenprofile, Aufgabenmodelle, Personas, Ist-Szenarien, User Journey Maps
- **Aus DIN EN 62366-1**
  Safety & Use Errors, Use Specification, gefährdungsbezogene Use-Szenarios, bekannte und vorhersehbare Gefährdungen und Gefährdungssituationen
- **Aus FDA Human Factors Guidance**
  Nutzungskontextbeschreibung (Summe aller anderen Artefakte dieses Schrittes)

3. Nutzungsanforderungen spezifizieren

- **Aus DIN EN ISO 9241-210**
  Erfordernisse, Nutzungsanforderungen
- **Aus DIN EN 62366-1**
  User-Interface Spezifikation (Requirements), User-Interface Evaluierungsplan, Beschreibung kritischer Aufgaben

4. Lösungen entwickeln

- **Aus DIN EN ISO 9241-210**
  Prototypen, Nutzungsszenarien, Storyboards, User Journey Maps, Aufgabenmodelle, Informationsarchitektur, Navigationsstruktur, Styleguide

5. Formative und summative Evaluierung

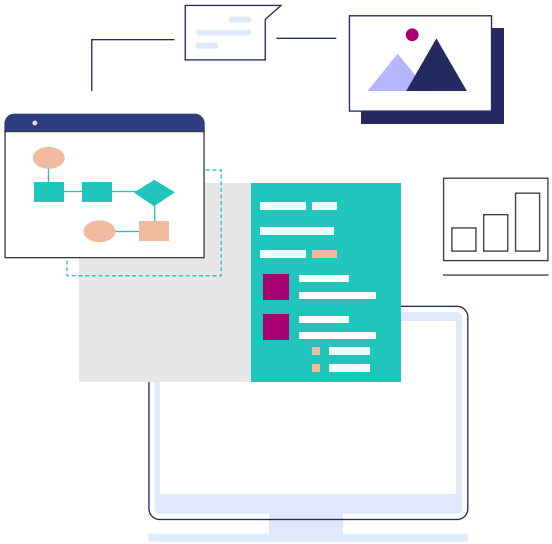
- **Aus DIN EN ISO 9241-210**
  Evaluierungsberichte
- **Aus DIN EN 62366-1**
  Keine spezifischen Artefakte gefordert, aber verpflichtende Dokumentation von Evaluierungsaktivitäten und -ergebnissen im Usability Engineering File, das alle Aktivitäten strukturiert nachweist.
- **Aus FDA Human Factors Guidance**
  HFE/UE Report, Description of User-Interface-Beschreibung, Prozess- und Interaktionsdokumentation

Usability Engineering ist Requirements Engineering

In der Praxis lassen sich Usability und Requirements Engineering kaum trennen. Alle Erkenntnisse aus der menschzentrierten Gestaltung – Interviews, Beobachtungen, Kontextanalysen – fliessen in die Anforderungen ein. Deshalb sollten beide Disziplinen eng verzahnt sein, idealerweise in einer Rolle oder in einem Tandemteam. Ein Projekt, das keinen Usability-Verantwortlichen hat, verschenkt Potenzial – und riskiert das Scheitern des Projekts.

Um beide Disziplinen reibungslos zu vereinen und trotz der vielen Regularien gute Produkte zu entwickeln, ist es notwendig, sich regelmässig diese Fragen zu stellen:

- Akzeptiere ich eine Person als Repräsentanten einer ganzen User-Gruppe?
- Habe ich die User und den Nutzungskontext mit eigenen Augen gesehen?
- Habe ich alle User-Gruppen berücksichtigt und beziehe sie von Beginn an in die Entwicklung ein?
- Vertraue ich vermeintlichen Experten, ohne mit tatsächlichen Usern gesprochen zu haben?
- Betreibe ich Usability Engineering vom Schreibtisch?
- Hinterfrage ich vermeintliche Gegebenheiten?
- Spiele ich irgendwo stille Post?
- Habe ich das tatsächliche Problem verstanden?
- Könnte ich ein detailliertes Bild des tatsächlichen Einsatzes meines Produkts zeichnen?
- Kann ich die Aufgaben von Nutzerinnen und Nutzern, die meine Lösung als Tool einsetzen, genau Schritt für Schritt beschreiben?



Fazit

Usability ist keine Zusatzaufgabe für die Zulassung, sondern ein strategischer Erfolgsfaktor – sofern man sie ernst nimmt. In unseren Projekten erleben wir immer wieder, welch grosser Unterschied zwischen formal erfüllten Anforderungen und echter Gebrauchstauglichkeit besteht. Mit dem ERNI-Medical-Usability-Prozess helfen wir unseren Kunden, von Beginn an die richtigen Fragen zu stellen, Nutzerinnen und Nutzer konsequent einzubeziehen und Usability nicht als Pflicht, sondern als Werkzeug zu verstehen. So entwickeln wir gemeinsam nicht nur sichere, sondern auch überzeugende und wirtschaftlich erfolgreiche Produkte.

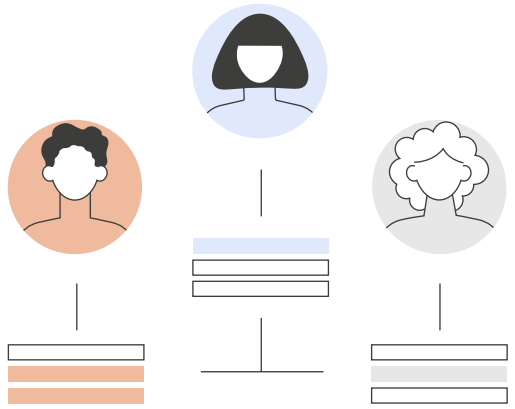
Wie man die Lücke zwischen Endnutzern und IT schliesst

Der Erfolg von Digitalisierungsprojekten hängt nicht nur von der Technologie ab, sondern auch von der Akzeptanz der Endnutzerinnen und Endnutzer, besonders wenn diese wenig technikaffin sind. Fühlen sie sich nicht gehört, entsteht Frust. Doch es ist nie zu spät, die Lücke zwischen Nutzern und IT auch in späteren Entwicklungsphasen zu schliessen.

Von Caroline Badoud, Senior IT Consultant, ERNI Schweiz

Wenn IT ausser Reichweite scheint

Technisch gesehen haben die Entwicklungsteams vielleicht eine solide IT-Anwendung erstellt. Doch eine Herausforderung bleibt: Wenn die Endnutzerinnen und Endnutzer das neue System nicht akzeptieren, hinterlässt die Fertigstellung des Projekts bei den Entwicklerinnen und Entwicklern einen bitteren Nachgeschmack. Die Kluft zwischen IT und Endnutzerinnen und Endnutzern zu überbrücken ist entscheidend für den Erfolg – und einer der Schwerpunkte in der Arbeit eines Requirements Engineers.



In diesem Artikel erklären wir, wie wir IT-Teams und Endnutzerinnen und Endnutzer mit minimalem Aufwand, geringen Kosten und wenig Widerstand zusammenbringen, insbesondere in der kritischen Phase der Einführung eines Digitalisierungssystems. In dieser Phase beginnen die Nutzerinnen und Nutzer, das neue System in ihren Arbeitsalltag zu integrieren – sie fangen an, das System zu lieben oder zu hassen. Ich bin überzeugt: Es ist nie zu spät, Lücken zwischen einzelnen Nutzerinnen und Nutzern und der IT zu schliessen, selbst in späteren Entwicklungsphasen. Die Akzeptanz eines neuen Systems erfordert jedoch einen strukturierten Ansatz, der alle Beteiligten einbezieht und die IT verständlich macht. Erforderlich sind eine kooperative Denkweise des Entwicklungsteams und eine sorgfältige Planung, um die reibungslose Einführung für alle zu gewährleisten.

Akzeptanz der Endnutzerinnen und Endnutzer in späteren Entwicklungsphasen erreichen

Es geht darum, eine Kultur des gegenseitigen Respekts und des kontinuierlichen Dialogs zu schaffen. Ich gehe Projekte persönlich mit dem Fokus auf den Aufbau von Vertrauen, einem möglichst tiefen Verständnis der Geschäftsanforderungen und der gemeinsamen Entwicklung von Lösungen mit den Anwenderinnen und Anwendern, nicht nur für sie, an.

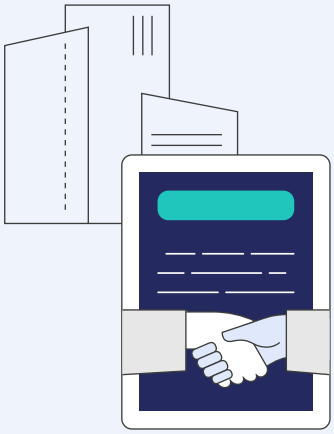
Organisieren Sie Fokusgruppendialoge, um mögliche Probleme frühzeitig zu erkennen. Auf Basis dieser Erkenntnisse lässt sich ein massgeschneiderter Massnahmenkatalog erstellen. Ich bin überzeugt, dass die folgenden sechs Schritte entscheidend dazu beitragen können, die Kluft zwischen Endnutzerinnen und Endnutzern und der IT zu überbrücken.

1.

Engagement vor Ort steigern und Gruppen überschaubar halten

Vertrauen entsteht in dem Moment, in dem Sie persönlich dort erscheinen, wo die Nutzerinnen und Nutzer arbeiten. Ob in der Produktion, in der Logistik oder anderswo – IT-Teams sollten die Nutzerinnen und Nutzer an ihrem Arbeitsplatz treffen. Das kann auch bedeuten, sich entsprechend auszurüsten und dorthin zu gehen, wo das IT-System später eingesetzt wird. Physische Präsenz signalisiert echtes Interesse an der täglichen Realität der Nutzerinnen und Nutzer.

Beziehen Sie die Nutzerinnen und Nutzer in Fokusgruppengespräche ein und halten Sie die Teilnehmerzahl möglichst gering. In kleinen Gruppen sind die Nutzerinnen und Nutzer eher bereit, sich aktiv zu beteiligen, Dialoge zu führen, Fragen zu stellen und sich einzubringen. So lassen sich Arbeitsabläufe, Bedenken und Schwachstellen aus erster Hand erfassen – weit über Remote-Anrufe oder Ticketingsysteme hinaus.



2.

Sprechen Sie die Sprache der Nutzerinnen und Nutzer

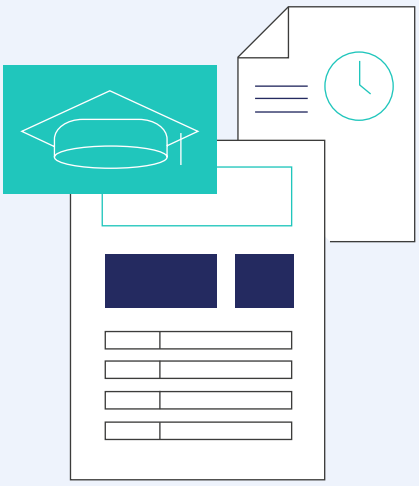
Gerade bei mehrsprachigen Anwendungen ist es eine besondere Herausforderung, alle Endnutzerinnen und Endnutzer einzubeziehen. Keine Nutzerin oder Nutzer sollte sich ausgeschlossen fühlen, nur weil er oder sie eine andere Sprache spricht als das Entwicklungsteam. Solche Ausgrenzung führt zu Frustration und kann Widerstand gegen das neue System hervorrufen. Wann immer möglich, empfehlen wir den Einsatz von mehrsprachigen IT-Spezialistinnen und -Spezialisten als Beratung – das steigert Zufriedenheit und Motivation der Nutzerinnen und Nutzer erheblich.

Es geht nicht nur um die gesprochene Sprache, sondern auch um den verwendeten Wortschatz. Technische Teams unterschätzen häufig, welche Wirkung ihr Vokabular haben kann. IT-Fachjargon kann für Endnutzerinnen und Endnutzer, die nicht technisch veranlagt sind, überwältigend sein. Mein Motto lautet: Klarheit verbindet. Komplexität hingegen wirkt oft eher abschreckend – sie soll beeindrucken, erreicht aber das Gegenteil. Gute Kommunikation hat genau das nicht zum Ziel. Begriffe und Konzepte zu verwenden, die den Nutzerinnen und Nutzern vertraut sind, fördert ein gemeinsames Verständnis. Es signalisiert, dass man auf derselben Ebene spricht und nicht aus der Distanz. Dabei geht es nicht darum, Inhalte zu vereinfachen, sondern Empathie über das eigene Ego zu stellen.

3.

Schulungen auf die Bedürfnisse der Nutzerinnen und Nutzer anpassen

Es liegt auf der Hand, dass der effektivste Schulungsansatz praktische Hands-on-Sitzungen sind, in denen die Nutzerinnen und Nutzer frühzeitig mit dem Live-System interagieren. Dieser Ansatz steigert Motivation und Verständnis, da er sich auf reale Anwendungsfälle konzentriert. Der Erfolg hängt nicht nur vom Schulungsinhalt ab, sondern auch vom Zeitplan. Häufig legt die IT-Abteilung Schulungs- und Rollout-Termine fest, ohne die betrieblichen Einschränkungen der Anwenderinnen und Anwender vollständig zu berücksichtigen. Die Einbeziehung der Anwendergruppen in die Rollout-Planung – abgestimmt auf ihre Verfügbarkeit – fördert die Akzeptanz und beschleunigt die Einführung, wodurch ein anwenderorientierter und flexibler Implementierungsprozess entsteht.



4.

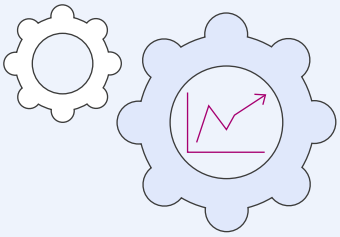
Reibungsloser Rollout statt Big Bang

Leistungs- und Belastungstests eines IT-Systems sind in einer Entwicklungsumgebung oft nur schwer realistisch zu simulieren – und manchmal weicht die Erfahrung im Live-Betrieb deutlich von der Simulation ab. Keine Entwicklerin oder kein Entwickler möchte eine Systemüberlastung in der Produktion riskieren. Daher empfiehlt sich eine schrittweise Einführung, bei der Woche für Woche weitere Benutzerinnen und Benutzer hinzukommen. So lässt sich die Systemleistung genau überwachen und Probleme können proaktiv behoben werden. Early Adopters helfen, Fehler frühzeitig zu identifizieren, damit das System reibungslos läuft, wenn zögerlichere Benutzerinnen und Benutzer eingebunden werden – und die Einführung mit minimalen Unterbrechungen fortgesetzt werden kann.

5.

Binden Sie den Betrieb fest ins Entwicklungsteam ein

Entwicklerinnen und Entwickler arbeiten oft lieber an neuen Funktionen, als ein bestehendes System zu warten. Als Moderatorin oder Moderator sollten Sie Ihr Entwicklungsteam dazu ermutigen, Nutzer-Feedback aus dem Betrieb wertzuschätzen. Machen Sie dieses Feedback sichtbar, organisieren Sie regelmässige Betriebsbesprechungen im Team, priorisieren Sie die Rückmeldungen und integrieren Sie sie in den Entwicklungsplan. Achten Sie darauf, dass jedes Entwicklungsticket auch Schritte für Tests, Benutzerdokumentation oder Kommunikation enthält. Mit der Zeit verbessert dieser Ansatz nicht nur die Systemstabilität, sondern zeigt den Teams auch den Wert von Benutzerbeiträgen, und fördert so eine kooperative Denkweise.



6.

Enge Zusammenarbeit auch nach dem Rollout sichern

Die Arbeit endet nicht mit der Inbetriebnahme des Systems – oft beginnt dann erst der wertvollste Dialog. Fördern Sie eine kontinuierliche Zusammenarbeit, indem Sie regelmässig Anwendertreffen organisieren, bei denen Erfahrungen offen ausgetauscht werden können. Anwenderinnen und Anwender berichten, was gut funktioniert und wo es Herausforderungen gibt, das Entwicklungsteam stellt aktuelle und geplante Funktionen vor und bittet gezielt um Feedback.

Besonders wirkungsvoll ist es, engagierte Nutzerinnen und Nutzer als Pioniere einzubinden, ihnen Anerkennung zu geben und sie an künftigen Verbesserungen zu beteiligen. Dieses kontinuierliche Engagement fördert ein Gefühl der Eigenverantwortung, hilft Entwicklerinnen und Entwicklern, die Hintergründe von Anfragen zu verstehen, und macht die Nutzerinnen und Nutzer zu Botschafterinnen und Botschaftern des Systems. So werden Pionier-Nutzerinnen und -Nutzer zu Motivatoren für zögerlichere Kolleginnen und Kollegen in ihrem Team.

Fazit

Vor allem mit diesen sechs einfachen Massnahmen lässt sich die Kluft zwischen IT-Teams, Endnutzerinnen und Endnutzern verringern – was zu höherer Zufriedenheit und stärkerer Akzeptanz des Systems führt. Meine Erfahrung zeigt: Vertrauen und offene Kommunikation sind die Grundlage jedes erfolgreichen Projekts. Wenn Nutzerinnen und Nutzer sich gesehen, gehört und wirklich einbezogen fühlen, sind sie weit eher bereit, das Endergebnis anzunehmen und zu unterstützen. Das mag selbstverständlich klingen, erfordert in der Praxis jedoch konsequente Anstrengungen, sorgfältige Planung und das Engagement aller Beteiligten. Wer ein solches Umfeld fördert, macht Nutzerinnen und Nutzer zu motivierten Partnerinnen und Partnern der digitalen Transformation und legt den Grundstein für zukünftige Innovationen im Unternehmen.

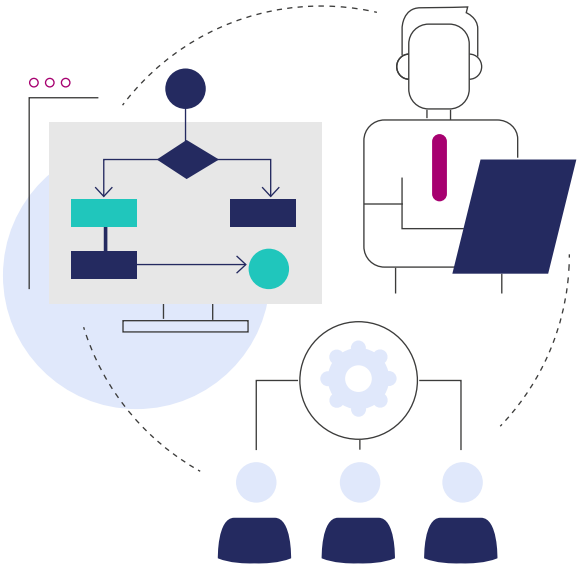
Anforderungsmanagement in komplexen Projekten mit einem digitalen Systemmodell

Bei der Entwicklung komplexer Medizinsysteme wie automatisierter Diagnostikgeräte oder Patientenmonitoring-Plattformen ist die präzise Anforderungserfassung zentral. Dokumentenbasierte Ansätze reichen nicht mehr. Wir zeigen, wie Requirements Engineering als strategische Modellierungsdisziplin Basis für digitale Systemmodelle schafft.

Von Thomas Ruckstuhl, Requirements Engineer, ERNI Schweiz

Die Folgen der Komplexität

Jede Domäne hat ihre eigenen spezifischen Anforderungen, die ein hohes Mass an Detailgenauigkeit erfordern. Besonders in stark regulierten Bereichen – aber nicht nur dort – ist es in der Regel zwingend erforderlich, die Arbeiten, die Funktionsweise des Systems, seine Möglichkeiten und Grenzen sowie die erforderlichen Tests sorgfältig zu dokumentieren. Erfahrungen aus komplexen Projekten in regulierten Bereichen können auch anderswo von grossem Nutzen sein.



Luftfahrt und Raumfahrt sind nur zwei komplexe Branche, in denen die Entwicklungsumgebung durch strenge regulatorische Anforderungen, einen Fokus auf Qualität und Risikomanagement, interdisziplinäre Zusammenarbeit und die Integration fortschrittlicher Technologien gekennzeichnet ist. Unternehmen müssen agil und vorausschauend bleiben, um diese Komplexitäten zu bewältigen und zum Beispiel sichere sowie wirksame Medizinprodukte erfolgreich auf den Markt zu bringen.

5 Ursachen für Projektkomplexität (auch in nicht regulierten Branchen)

Vernetzung

Systeme sind nicht mehr isoliert – sie sind mit Apps, Cloud-Plattformen und anderen Geräten verbunden, was die Komplexität von Schnittstellen und Integrationen erhöht.

Mehr Funktionalität

Zusätzliche Funktionen steigern die Komplexität, da sie neue Interaktionen, Abhängigkeiten und potenzielle Fehlerquellen schaffen. Dadurch wird das System schwieriger zu verstehen, zu warten und zu verwalten.

Kürzere Entwicklungszyklen

Schnellere Entwicklungszyklen erhöhen den Druck, in kurzer Zeit hochwertige Projektergebnisse zu liefern.

Cybersicherheit

In vernetzten Umgebungen sind Systeme ständigen Bedrohungen ausgesetzt.

Sich weiterentwickelnde Vorschriften

Projekte müssen an sich verändernde gesetzliche Vorgaben und Compliance-Anforderungen angepasst werden.

Requirements Engineering als Teil von Systems Engineering

Ein Ansatz für das Systems Engineering, wie er vom INCOSE (International Council on Systems Engineering) definiert wird, gewinnt immer mehr an Bedeutung, weil die Industrie immer häufiger mit komplexen Projekten konfrontiert ist. Für Unternehmen, die in stark regulierten Umgebungen tätig sind und komplexe, intelligente Geräte entwickeln, stellt Systems Engineering einen wirksamen Ansatz dar, um diese Herausforderungen zu bewältigen. Als transdisziplinärer und integrativer Ansatz für die erfolgreiche Entwicklung, Nutzung und Ausserbetriebnahme technischer Systeme nutzt er Systemprinzipien und -konzepte sowie wissenschaftliche, technologische und managementbezogene Methoden.

Das V-Modell: Darstellung der Umwandlung von Stakeholder-Bedürfnissen in Systemfähigkeiten

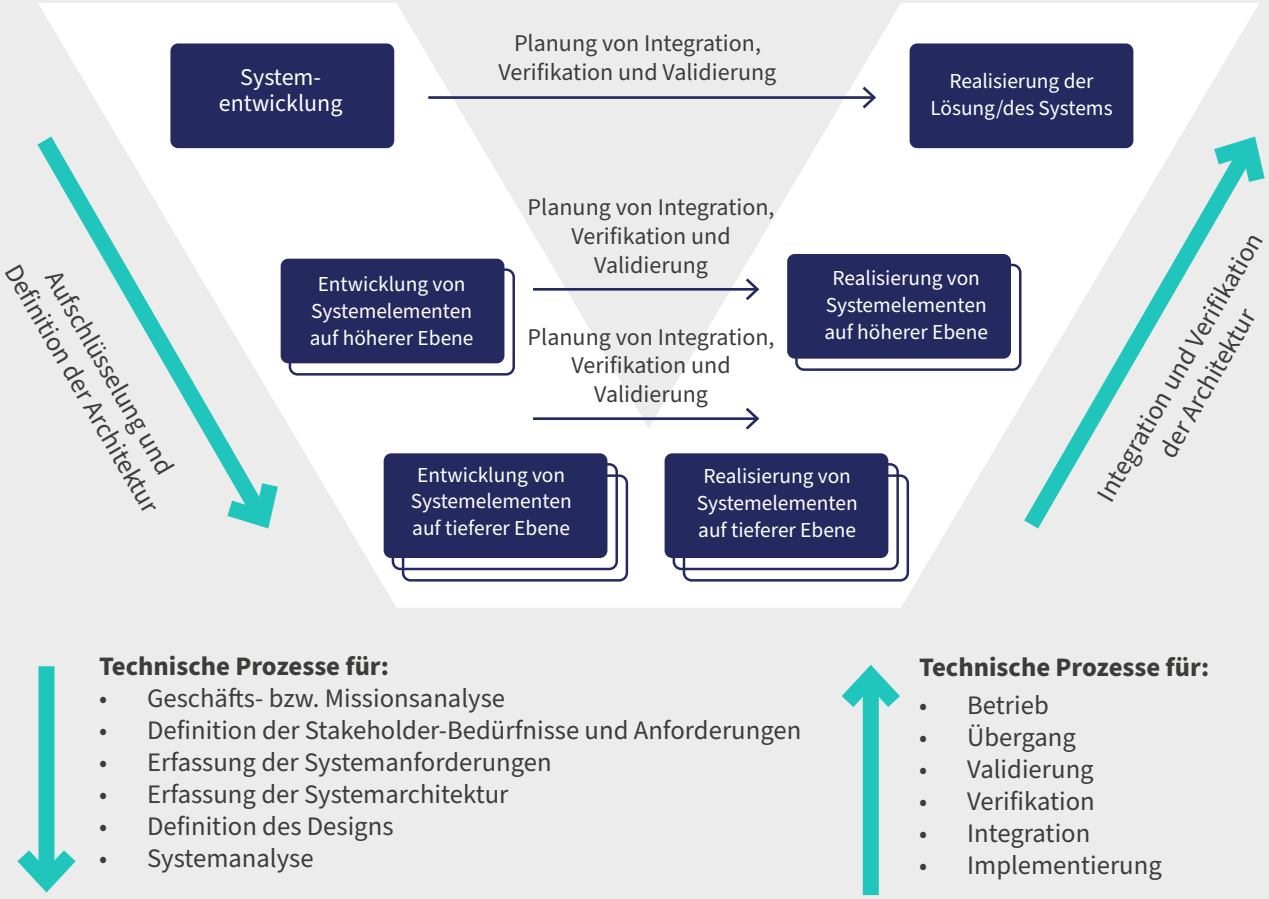
Das V-Modell oder Vee Model ist im Systems Engineering ein visuelles Framework, das den Lebenszyklus der Systementwicklung auf strukturierte und sequenzielle Weise veranschaulicht. Es visualisiert die Beziehung zwischen jeder Entwicklungsphase auf der linken Seite des «V» und der entsprechenden Validierungs- beziehungsweise Verifizierungsaktivität auf der rechten Seite.

Dabei gibt es drei spezifische technische SE-Prozesse, an denen Requirements Engineering massgeblich beteiligt ist:

- Geschäfts- oder Aufgabenanalyse
- Definition der Bedürfnisse und Anforderungen der Stakeholder
- Definition der Systemanforderungen

Diese Schritte definieren die Bedürfnisse und Anforderungen der Stakeholder und wandeln die stakeholder- und nutzerorientierte Sichtweise der gewünschten Fähigkeiten in eine technische Sichtweise einer Lösung um, die den betrieblichen Anforderungen der Nutzerinnen und Nutzer entspricht. Die Systemanforderungen bilden die Grundlage für Architektur, Design, Implementierung und Verifizierung. Mit der Etablierung des Requirements Engineering in diesem Abschnitt des SE-Lebenszyklus bleiben die Anforderungen während des gesamten Lebenszyklus sichtbar, überprüfbar und konsistent.

Infografik 1: Das V-Modell



Adaptiert aus INCOSE SEH, 2023 (Forsberg et al., 2005)

Komplexität erfordert mehr als nur handwerkliches Können

Erfahrene Requirements Engineers beherrschen ihr Handwerk und kennen die Grundlagen – alle Kontext-, Use-Case- und Aktivitätsdiagramme – und wissen, wie man auf deren Basis Anforderungen ausarbeitet. Die eigentliche Herausforderung beginnt jedoch erst, wenn Anforderungen von statischen Artefakten zu lebendigen, miteinander vernetzten Assets werden, die über Tools, Teams und Zeit hinweg verwaltet werden müssen. Dies erfordert systemisches Denken, Zusammenarbeit und Anpassungsfähigkeit.

ALM-Tools: Ein guter Anfang, aber bei Weitem nicht das Ziel

ALM-Tools für das Application Lifecycle Management sind in regulierten Bereichen wie der Medizin, dem Finanzwesen oder der Luft- und Raumfahrt von entscheidender Bedeutung, weil sie die Softwareentwicklung von Anfang bis Ende verwalten und so Compliance und Qualität sicherstellen.

Anforderungen, die aus den Artefakten des Requirements Engineering abgeleitet und ausgearbeitet wurden, werden in einem ALM-Tool in Textform gespeichert und verwaltet – was selbstverständlich bereits ein guter Anfang ist. Die Zeiten, in denen Anforderungen in Tabellen gepflegt wurden, sind vorbei.

Oft werden Requirements-Engineering-Artefakte in verschiedenen Dokumenten oder Tools gespeichert. Es ist eine Herausforderung, diese Diagramme während des Projekts – und insbesondere in der Betriebs- und Wartungsphase – aktuell zu halten und miteinander abzugleichen. Mitunter kommt es vor, dass Tools wie Microsoft Visio bereits vor Projektabschluss ausser Betrieb genommen werden und die Diagramme dadurch verloren gehen oder nicht mehr gepflegt werden können. Und in welchem Projekt wünscht man sich nicht eine einzige Quelle der Wahrheit?

Design-Rückverfolgbarkeit und Bewusstsein für Auswirkungen – eine Verbindung, die fehlt

Eine weitere grosse Herausforderung bei komplexen Projekten entsteht, wenn die Requirements Engineers nicht wissen, welche Systemfunktionen oder -komponenten von ihren Anforderungen betroffen sind. Oder sie wissen es zwar, aber es ist nicht ordnungsgemäss dokumentiert und damit im Projekt nicht transparent. Jede und jeder Requirements Engineer sollte in der Lage sein, die wichtigsten Anforderungen im Design nachzuverfolgen und Abweichungen zu erkennen. Daher müssen nicht nur die Verknüpfungen zwischen den Anforderungen und Testfällen gepflegt werden, sondern auch die Verbindung zwischen den Anforderungen und entsprechenden Design- und Architekturelementen.

Unsere Erfahrung hat gezeigt, dass Requirements Engineering nicht nur eine Schlüsselkomponente im Systems-Engineering-Prozess ist, sondern vom Systems-Engineering-Ansatz profitiert – insbesondere im Kontext des modellbasierten Systems Engineerings (MBSE).

MBSE betritt die Bühne

Model-Based Systems Engineering ist eine standardisierte Methodik zur Vereinfachung von Anforderungen, Design, Analyse, Verifizierung und Validierung bei der Entwicklung komplexer Systeme. Im Gegensatz zum dokumentenzentrierten Engineering steht die Modellierung des Systemdesigns im Mittelpunkt. Die zunehmende Verbreitung digitaler Modellierungsumgebungen in der Industrie in den letzten Jahren hat die Anwendung von MBSE massgeblich gefördert. Bereits im Januar 2020 stellte die NASA fest, dass MBSE «sowohl in der Industrie als auch in Regierungsorganisationen zunehmend als Mittel zur Beherrschung von Systemkomplexität eingesetzt wird». Methodisch betrachtet ist MBSE eine Sammlung miteinander verbundener Prozesse, Methoden und Werkzeuge.

Die INCOSE Systems Engineering Vision 2020 (2007) definiert MBSE (modellbasiertes Systems Engineering) wie folgt:

Die formalisierte Anwendung von Modellen zur Unterstützung von Systemanforderungen, Design, Analyse, Verifizierung und Validierung, beginnend in der Konzeptphase und fortgesetzt während der gesamten Entwicklung und späteren Lebenszyklusphasen.

Aus Sicht des Requirements Engineering ist es wichtig, zum Modell beizutragen, indem Artefakte wie Aktivitäts- und Sequenzdiagramme sowie ausgearbeitete Anforderungen erstellt werden. Sobald das Systemdesign und die Architektur vorliegen, können die Anforderungen den entsprechenden Elementen des Modells zugeordnet und zu ihnen zurückverfolgt werden. Auf diese Weise wird eine Beziehung zwischen den textbasierten Anforderungen und den Modellelementen hergestellt und aufrechterhalten.

Modellbasiertes Systems Engineering spielt eine wichtige Rolle, da es durch visuelle Darstellungen die Klarheit und Kommunikation zwischen allen Beteiligten verbessert, die frühzeitige Validierung und Verifizierung von Anforderungen erleichtert und das Risikomanagement dank der Simulation und Analyse komplexer Systeme verbessert. MBSE fördert die Rückverfolgbarkeit, stellt sicher, dass alle Systemelemente mit den Anforderungen übereinstimmen, und ermöglicht eine iterative Entwicklung, die die Anpassung an Änderungen erleichtert. Darüber hinaus verbessert es die Zusammenarbeit zwischen funktionsübergreifenden Teams und bietet einen strukturierten Ansatz für das Management von Komplexität, was letztlich zu einer effizienteren und erfolgreicheren Systementwicklung führt.

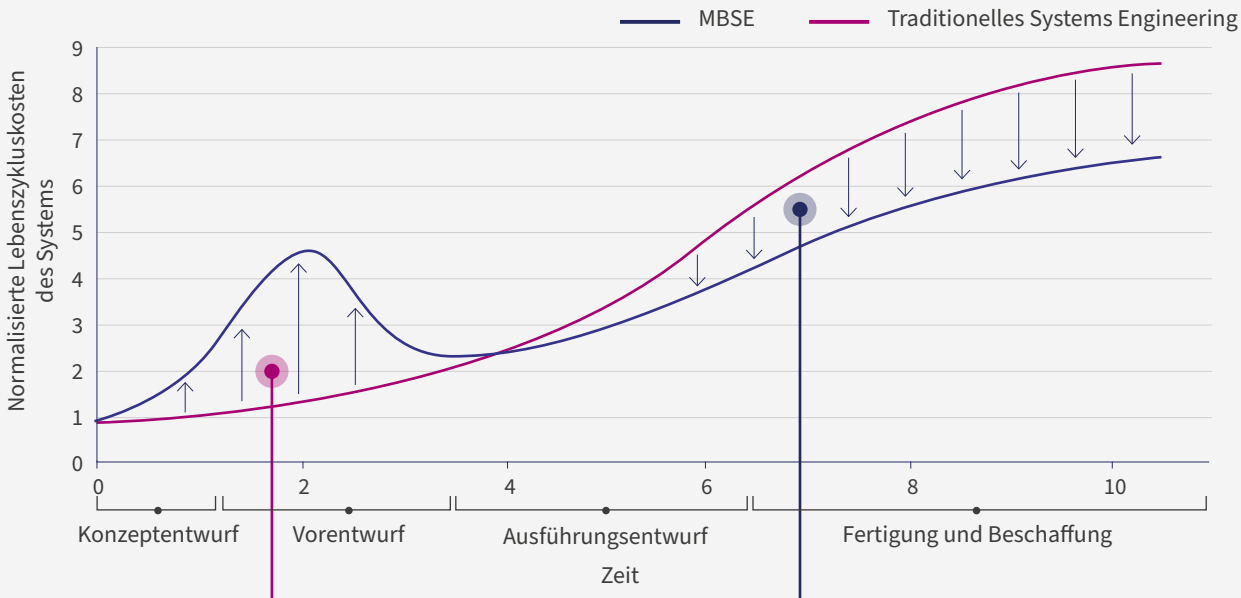
Kosten-Nutzen-Verhältnis

Die Implementierung von MBSE erfordert im Vergleich zu herkömmlichen Ansätzen höhere Vorabinvestitionen. Zeit und Ressourcen werden früh im Lebenszyklus für die Toolintegration, die Modellierung, Schulungen und die Anpassung von Arbeitsabläufen benötigt. Dieser Mehraufwand ist eine strategische Investition. Projekte, die MBSE einsetzen, erzielen in späteren Entwicklungsphasen erhebliche Erträge, insbesondere bei Verifizierung, Integration und Compliance. Das Verständnis dieser Zusammenhänge ist entscheidend. Die Analyse der Kostentreiber für die frühzeitige Einführung von MBSE und der Werthebel in späteren Phasen hilft, ein überzeugendes Geschäftsszenario für Veränderungen zu entwickeln. Zu den üblichen Anfangsinvestitionen gehören Prozessanpassung, Toolkonfiguration und Modellerstellung. Zu den nachgelagerten Vorteilen zählen eine schnellere Rückverfolgbarkeit, weniger Nacharbeit, ein einfacheres Änderungsmanagement und reibungslosere behördliche Audits. Über den gesamten Lebenszyklus betrachtet, spricht die Wirtschaftlichkeit von MBSE eindeutig für langfristige Effizienz und Produktqualität.

Sobald ein Change Management erforderlich ist und eine Funktion geändert werden muss, kann die oder der Requirements Engineer die entsprechenden Anforderungen leicht identifizieren. Umgekehrt lässt sich bei einer Änderung einer Anforderung nachvollziehen, welche Funktionen oder Teile des Modells davon betroffen sind. Alle Diagramme sind Teil des Modells und bilden eine einzige Quelle der Wahrheit. Dies gilt nicht nur für die Entwicklungsphase, sondern auch für Betrieb und Wartung, wenn die Dokumentation noch wichtiger wird.

MBSE bietet eine Reihe bedeutender Vorteile. Es stellt sicher, dass das Modell in sich konsistent ist und unterstützt durch semantische Beziehungen eine vollständige Rückverfolgbarkeit. So entstehen klare Verbindungen zwischen Anforderungen, Designelementen und Systemverhalten. MBSE fördert gleichzeitig ein hohes Mass an Wiederverwendbarkeit, steigert die Effizienz und senkt die Fehlerquote über den gesamten Entwicklungslebenszyklus. Ausserdem können Requirements Engineers Anforderungen systematisch direkt aus dem Modell ableiten – eine solide Grundlage für die automatisierte Dokumentenerstellung.

Infografik 2: Schlüsselfaktoren in Bezug zu MBSE-Investitionen und -Nutzen



Faktoren im Zusammenhang mit MBSE-Investitionen

- Kosten der Prozessdefinition
- Infrastrukturkosten
- Schulungskosten
- Modellentwicklung
- Modellverifizierung
- Modellpflege
- Konfigurationsmanagement

Vorteile im Zusammenhang mit MBSE-Nutzen

- Früherkennung von Fehlern
- Wiederverwendung
- Risikominderung
- Verbesserte Kommunikation
- Einsatz in der Lieferkette
- Produktlinien-Definition
- Normenkonformität und Rückverfolgbarkeit

*Quelle: Azad M. Madni * und Shatad Purohit, 2019, researchgate.net*

Praxisbeispiel: Ein komplexes Diagnosesystem

Dank unserer umfassenden Erfahrung im Bereich Medizintechnik haben wir die Entwicklung eines vollautomatischen Diagnosesystems begleitet, das verschiedene Parameter messen kann – alles in einer kompakten Form.

In einem solchen System spielen mehrere Faktoren eine entscheidende Rolle:

- **Ehrgeizige Produktvision und -umfang**
- **Wettbewerbsfähigkeit auf dem Markt und Kostensensibilität**
- **Fortgeschrittene technologische Entwicklung**
- **Umfassende Integration von Hardware und Software**
- **Strenge regulatorische und Compliance-Rahmenbedingungen**
- **Komplexes Projektmanagement**

Mit dem MBSE-Ansatz verknüpfte das Entwicklungsteam jede Anforderung der Stakeholder mit einer Systemfunktion, einer Komponente und einem Testfall. Änderungen in einem Bereich – beispielsweise eine neue gesetzliche Anforderung für die Bilirubinmessung – wurden automatisch im gesamten Modell übernommen, sodass die Auswirkungen auf die Architektur und Verifizierungspläne sofort sichtbar wurden. Diese Rückverfolgbarkeit schuf Vertrauen in die Konformität, minimierte Nacharbeiten in späteren Phasen und ermöglichte die schnellere Erstellung der Dokumentation für die Aufsichtsbehörden.

Fazit

Die Integration von Systems Engineering und MBSE bedeutet nicht nur den Einsatz neuer Tools oder Methoden. Sie erfordert eine unternehmensweite Veränderung der Denk- und Arbeitsweise über Disziplinen hinweg und verlangt sowohl technisches Fachwissen als auch Kompetenzen im Change Management. Ein erfahrener Partner an Ihrer Seite unterstützt Sie dabei, komplexe Projekte in regulierten und nicht regulierten Bereichen erfolgreich zu meistern. Der Unterschied liegt nicht nur in der Implementierung von Frameworks, sondern auch darin, diese auf den branchenspezifischen Kontext und die internen Fähigkeiten zuzuschneiden und sicherzustellen, dass jeder Schritt auf dem Weg mit den übergeordneten strategischen Zielen in Einklang steht.

Über ERNI

ERNI steht für Swiss Software Engineering. Woran sind wir wirklich interessiert? Wie können wir Sie und Ihre Mitarbeitenden besser als jedes andere Unternehmen bei der Entwicklung und Vermarktung softwarebasierter Produkte und Dienstleistungen unterstützen? Unsere globale Plattform für Softwareentwicklung in Kombination mit einem fundierten Marktverständnis bildet den Rahmen für den Erfolg unserer Kunden. Unser Team implementiert auch komplexe Projekte, befähigt Menschen und liefert herausragende Kundenlösungen in kürzester Zeit. Wir wenden die Schweizer Mentalität mit Verhaltensmustern wie Konsensbildung, Pragmatismus, Integration, Zuverlässigkeit und Transparenz auf globaler Ebene an – und dies seit unserer Gründung im Jahr 1994; zusammen mit unserem grossartigen Team, das die Basis für erfolgreiche Software-Projekte ist. Heute beschäftigt die ERNI Group weltweit mehr als 800 Mitarbeitende.

Über .experience

In diesem Magazin, das von ERNI mehrmals im Jahr veröffentlicht wird, informieren wir über wichtige Erfahrungen, die wir bei unserer täglichen Arbeit in den Bereichen Zusammenarbeit, Prozesse und Technologie machen.

Impressum

Ausgabe 1/2025

ERNI

Swiss Software Engineering
betterask.erni

Herausgeber

ERNI Management Services AG

ERNI Standorte

ERNI Schweiz AG
Bern
Zürich
Luzern
Lausanne
Basel

ERNI Consulting España S.L.U.
Barcelona
Madrid
Sant C. del Vallès

ERNI Development Center Spain, S.L.
Valencia

ERNI (Germany) GmbH
Frankfurt
München
Berlin
Stuttgart

ERNI Development Centre Philippines Inc.
Manila

ERNI Development Centre Romania S.R.L.
Cluj-Napoca

ERNI Singapore Pte Ltd.
Singapur

ERNI (Slovakia) s.r.o.
Bratislava

ERNI USA
New York

Kontakt

ERNI Management Services AG
Löwenstrasse 11 I 8001 Zürich
Email: marketing@betterask.erni
Phone: +41 58 268 12 00
Web: www.betterask.erni

ERNI in den Social Media Netzwerken

© 2025
von ERNI Management Services AG

ERNI